TOPICS IN HIGH-DIMENSIONAL INFERENCE WITH APPLICATIONS TO RAMAN SPECTROSCOPY

by

Amy S. Wagaman

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Statistics)
in The University of Michigan
2008

Doctoral Committee:

Assistant Professor Elizaveta Levina, Chair
Professor Michael D. Morris
Associate Professor Kerby A. Shedden
Assistant Professor Ji Zhu

To my family, especially my grandmother

ACKNOWLEDGEMENTS

I would like to thank those who made the completion of this dissertation possible. I would especially like to thank Liza Levina for providing excellent guidance and boundless patience during the development of the thesis. I would also like to thank George Michailidis for helpful discussions on multi-dimensional scaling and other dimension reduction techniques. I am also very grateful for the support of my dissertation committee, and I would like to thank Michael Morris, Kerby Shedden, and Ji Zhu for their guidance. I also owe a note of gratitude to Andrew Callender, Gurjit Mandair, Kurt Golcuk, and Kate Dooley from the Morris lab for the time they took to explain the chemistry setup to me, as well as data collection they did for our analyses. I would also like to thank my family for their support and encouragement. Many thanks to Kat Kentes, Herle McGowan and Brenda Gunderson for additional support and encouragement during my years at Michigan.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

TOPICS IN HIGH-DIMENSIONAL INFERENCE WITH APPLICATIONS TO RAMAN SPECTROSCOPY

by

Amy S. Wagaman

Chair: Elizaveta Levina

Recent advances in technology have led to a demand for statistical techniques for high-dimensional data. This thesis explores dimension estimation and reduction, covariance estimation and regularization, and improving nearest neighbor graphs with some examples in the context of Raman spectroscopy.

A new technique for estimating intrinsic dimension is proposed and used to estimate the number of pure components in a chemical mixture in Raman spectroscopy applications. We show how the new method improves over existing procedures, can be adapted via smoothing to deal with high noise levels, and has future applications in detecting mixture homogeneity.

Next, we consider covariance estimation and regularization in high dimensions. Regularized covariance estimators in high dimensions depend on the ordering of variables or are completely invariant to variable permutations. We propose a new method, Isoband, which uses the unordered data to discover a suitable order for the variables and then apply methods which depend on variable ordering to improve

covariance estimation for sparse covariance matrices. Our method has the additional advantage of being able to detect blocks within covariances and thus create additional sparsity and structure in the estimate. We show by simulations that when a suitable variable ordering exists, we do better by discovering it than by using a permutation-invariant method, and illustrate the new methodology on a real data example.

The Isoband methodology relies on a nearest neighbor graph, and in the last chapter, we address improving robustness of nearest neighbor graphs, which have widespread statistical applications. In our application, the nearest neighbor graph is based on the variables rather than the observations. Two new methods are proposed which improve upon the basic nearest neighbor graphs by removing spurious edges by either bootstrapping the data or smoothing. Both methods are competitive compared to existing graph perturbation methods in the literature.

CHAPTER I

Introduction

Many modern data sets are characterized by high dimensionality - that is, the number of variables observed (p) may be very large, especially when compared to the number of observations (n). Examples cover a wide range of fields including spectroscopic analysis, gene expression data, financial data, and many others. In this high-dimensional situation, many traditional statistical procedures fail, and developing alternatives specifically for high-dimensional data has become an important area of modern statistics. This thesis contributes to the following general areas of high-dimensional inference: dimension estimation and reduction, covariance estimation and regularization, as well as perturbation methods for improving robustness of nearest neighbor graphs, with some applications to Raman spectroscopy.

For high-dimensional data, dimension estimation and reduction can be a valuable data analysis tool. Estimating the true "intrinsic" dimension, s, of the data in p dimensions ($s \ll p$) can enable the use of dimension reduction methods and thus remove the "curse of dimensionality". Chapter II presents an interdisciplinary project on Raman spectroscopy to identify chemical components in objects scanned as images, where the number of distinct components can be viewed as the intrinsic dimension of the data. We developed a new procedure [38] based on the maximum

likelihood estimator (MLE) of intrinsic dimension [36] that proved to be superior to traditional estimators used in chemistry such as the Malinowski F-test [39] for estimating the number of components. Preliminary background information on Raman spectroscopy is given later in this Introduction (Section 1.1).

Another area of high-dimensional inference we consider is estimating the covariance matrix. This is an important inference problem because of the number of statistical techniques that require an estimate of the covariance matrix or its inverse. Recent results in random matrix theory have shown the sample covariance performs poorly when p is large relative to n. Thus, alternative ways of estimating covariance are needed in high-dimensional situations. Alternative methods for estimating the covariance can be divided into two classes depending on whether or not they require ordered variables (e.g., time series). Background on known results on covariance matrices in high dimensions and various alternative covariance estimators is given in Section 1.3. In Chapter III, we propose a methodology we call Isoband to uncover structure in covariance by applying dimension reduction methods to discover an ordering of variables. Chapter III also contains a discussion of eigenspace comparison measures needed to study the impact of different covariance estimators on the resulting statistical analysis (e.g. PCA).

Chapter IV contains preliminary results on perturbation methods for constructing a robust k nearest neighbor (NN) graph. The motivation comes from the Isoband methodology in Chapter III, which relies on NN graphs, but many statistical procedures in classification, clustering, semi-supervised learning, and dimension reduction rely on k-NN graphs as well. The stability and robustness to noise of these graphs has not been extensively studied. We examine ways of perturbing graphs with the goal of improving robustness, propose two new methods, and apply these ideas to

the Isoband methodology where constructing a more robust NN graph improves covariance estimation.

1.1 Chemistry background

In Raman spectroscopy, an object which is a mixture of chemical components is hit with a laser, molecular excitations or absorptions of photons occur, the laser photons may be altered as a result, and are then recollected. The photons are altered if they interact with molecular bonds that are vibrating, and since not all molecules vibrate at the same time, not all photons are altered (the fraction of altered photons is small). It should be noted that some molecules never vibrate. Information about all the returned photons is collected, and data from the photons that were not altered is discarded. The information collected about the altered photons can be used to determine the chemical components present in the scanned object. The object is scanned as an image, hence data is collected by pixel. At each pixel, changes in the laser photons are measured at specific wavelengths. Each wavelength corresponds to a wavenumber, which is simply the inverse of the wavelength and hence is a measure of the frequency of the vibration in the molecules and therefore also of the energy required to excite the molecules. The observation at each pixel is a vector of energy intensities by wavenumber, and is called a mixture spectrum. The spectra of the chemical components present in the mixture are referred to as pure component spectra. The mixture spectrum at each pixel is a function of the pure component spectra and their relative concentrations.

The chemistry problem is to extract pure component spectra from the observed mixture spectra, which allows identification of the chemical components in the mixture. However, the researcher may not have a priori knowledge about the number

of pure component spectra that are present in the mixture. Therefore, a number of components to extract must be determined and the extraction performed based on that estimated number of components. It is not guaranteed that the number of components present is going to be homogeneous throughout the image under analysis. As a general rule, regions of the image (i.e., parts of the object) with more components are more interesting, and may warrant further study. Hence, when considering how to determine the number of components to extract, local counting methods must be considered along with global counting methods.

We note that the setting is high-dimensional. Each image is at least 64 by 64 pixels, hence the number of observations n is at least 4096 (other example image sizes are 128×128 and 256×256). The number of observations is typically larger than 10,000, but this may be reduced by summing over one dimension. For example, a data set that is 100×280 may be reduced to 100×70 simply by summing up the spectral information for groups of 4 observations along the second spatial dimension. This amplifies the signal by enhancing the signal-to-noise ratio to help with further analysis. The number of wavenumbers p where intensities are recorded at each pixel ranges from 500 to 900 (depending on the laser used), but is consistent throughout a single image. Thus, the data matrix (observations by wavenumber) is larger than 4000×500, but the number of chemical components present in the mixture is usually under 20.

The extraction phase as currently practiced (further information in Section 1.2) involves user intervention and utilizes expert knowledge to obtain the pure component spectra. Chemists may suspect certain chemicals are present and can examine the extracted spectra to see if any chemicals are easily recognizable (based on peak location and other spectral features). This knowledge enables chemists to use/modify

the extraction procedure to obtain "meaningful" results, but can be time-consuming and subjective. Future applications will require faster analysis as the technology becomes adapted to scan human beings rather than laboratory slides.

To further discussion, we introduce some notation. For each Raman spectroscopy image, let n be the total number of pixels, and let p be the number of wavenumbers measured at each pixel. Each individual spectrum is a $1 \times p$ row vector denoted $\boldsymbol{X}_i$ where $i = 1, ..., n$. We call the collected observed mixture spectra X which is an $n \times p$ matrix. The pure component spectra are combined into a single (unknown) matrix, A, which is $s \times p$ where s is the number of component spectra. Finally, we have an unknown weight (concentration) $n \times s$ matrix W, containing weights with which the pure component spectra are combined to obtain the mixture spectra.

There are two main physical properties that arise from the underlying chemistry of the problem, including one that yields a model for the setting.

1. Linear additivity of spectra.

   The first physical property of the system is that the observed mixture spectra are simply a linear combination of the pure component spectra, with the concentration matrix providing the weights. For the Raman scattering process, under the assumptions that the chance of exciting a molecule is constant and that identical molecules behave independently, the model implies that the molecular changes in the altered photons that are caused by each pure component simply add up at each measured wavenumber [51]. Thus we have the following relationship:

   $$X = WA + \epsilon, \tag{1.1}$$

   where ϵ is noise (details below). This model is analogous to the Lambert-Beer-

Bougeur Law for absorption processes [59]. Note that X is observed, while W, A, ϵ, and s are not. Each entry in X, denoted X_{ij}, corresponds to the spectral intensity for pixel i at wavenumber j. Here, ϵ is an $n \times p$ matrix of error terms, which are assumed to be independent and identically distributed Gaussian random variables. This assumption is made throughout the chemistry literature [39] [58], though sometimes the errors are instead assumed to be uniform on (-1,1) [59]. Note that the errors may be spatially correlated, and hence the independence assumption for errors may not be valid in practice. The linearly additive property is the primary feature of the problem, and is the reason why principal components analysis (PCA) has been widely applied in this area.

2. Non-negativity of spectra and spectral weights.

   The spectra cannot have negative intensity values at any wavenumber because negative numbers of photons cannot be observed. In addition, the spectral weights (concentrations) must also be non-negative. Hence, each entry in W and in A must be non-negative [58]. That is, we need each $W_{ik} \geq 0$ and $A_{kj} \geq 0$, where k is the index for the pure component spectra, $k = 1, .., s$.

Apart from the constraints, the general setup of this problem implies that A and W can be recovered from X by either factor analysis or principal components analysis. Each $\boldsymbol{X}_i$ is a linear combination of the spectra $\boldsymbol{A}_k$ and corresponding weights $\boldsymbol{W}_k$, where each $\boldsymbol{A}_k$ is a 1 by p vector representing a single spectrum and each $\boldsymbol{W}_k$ is an n by 1 vector representing the weights given to spectrum k in each pixel. If the $\boldsymbol{A}_k$'s are viewed as unknown factors, with the $\boldsymbol{W}_k$'s as factor loadings, then this scenario is akin to factor analysis (which we will see is utilized by some of the current chemistry techniques).

1.2 Current methods in Raman spectroscopy

The process of identifying the pure components present in the mixture spectra currently used in the chemistry literature can be broken down into two main stages. Conceptually, the first stage is both a counting and extraction stage using PCA, where s and the matrix of principal components V are found. The second stage is rotation, where the proposed extracted spectra (the principal components in V) are rotated into spectrally "meaningful" components. That is, a rotation matrix T is applied to V such that $\hat{A} = TV$, and the resulting spectra found in $\hat{A}$ meet the necessary constraints. The first stage can be broken further down into a counting stage where the number of components to extract is determined and an extraction stage. The counting is usually done using a PCA-based technique, such as a scree plot, or Malinowski's F-test (to be discussed later), or by eye [58][39].

Different chemistry techniques utilize different manipulations during the rotation stage to satisfy the constraints, but do not make any adjustments to PCA itself in the extraction phase. Recall that principal components analysis is a dimension reduction technique which finds linear combinations of the original variables that best explain the variability in the data. Recall that $\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_n$ are the p-dimensional data vectors. PCA consists of computing the eigenvalues and eigenvectors of the data covariance matrix $\hat{\Sigma} = 1/n \sum_{i=1}^{n} (\boldsymbol{X}_i - \bar{\boldsymbol{X}})(\boldsymbol{X}_i - \bar{\boldsymbol{X}})'$, where A' denotes matrix transpose, and $\bar{\boldsymbol{X}} = 1/n \sum_{i=1}^{n} \boldsymbol{X}_i$. Alternatively, the principal components can be computed from the singular value decomposition of the data matrix X. The eigenvalues of $\hat{\Sigma}$, $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq ... \geq \hat{\lambda}_p \geq 0$, represent the amount of variation in the data explained by the corresponding principal component. Briefly, we examine some of the current chemistry algorithms for extracting and rotating spectra, which fall

into a class called self-modeling curve resolution algorithms (SMCR) [26, 25]. These algorithms have a variety of applications, though we limit our discussion to their use in Raman spectroscopy.

Factor analysis-based SMCR algorithms have been used on Raman images to determine the mineral and matrix components contained within bone tissue [56, 42, 54, 57]. In this context, factor analysis is used interactively to allow the user to visualize different linear combinations of potentially useful eigenvectors, which are in turn extracted by principal component analysis (PCA) [44]. This approach has been used to distinguish between healthy and diseased bone tissues [42, 54, 57], as well as to highlight chemical differences between trabecular and cortical bone structures at the micro-structural level [56]. Factor analysis is also useful for removal of non-informative eigenvectors associated with background tissue fluorescence and bone tissue embedding reagents, such as poly(methyl methacrylate) (PMMA) [56, 42].

While factor analysis-based starting techniques are useful and popular, they frequently run into difficulties with over- or under-determination because they assume that an appropriate number of eigenvectors (number of components) has been selected [20, 58]. Alternatives to factor analysis include Simplisma and band-target entropy minimization (BTEM) [58], which uses a larger number of eigenvectors than the number of components, s. BTEM has been shown to outperform Simplisma in terms of recovering minor components [58]. However, even in BTEM, the final decision on the number of extracted components has to be made by the user and is based on visual inspection of the eigenvectors. It also places considerable demands on the user in terms of computational time and human interaction (exhaustive band targeting) [58]. This is especially undesirable if the user needs to analyze a large collection of Raman images or spectra, or to perform the analysis quickly.

Finally, we note the data from Raman experiments are very complex, with background fluorescence and other illumination issues to be considered, that signal-to-noise (s/n) ratios in real-time *in situ* Raman experiments are often low, and the structure of the noise may, in general, be non-linear (beyond the spatial correlation already mentioned). This situation may be further compounded by local variations in s/n ratios as the Raman scattering properties of the irradiated specimen depend on its surface morphology and chemical composition; this may make the use of a local method more appropriate in some situations.

In summary, homogeneity, concentration, and the number of pure components within the chemical system are seldom known in advance [49, 8] and low signal-to-noise ratios must be taken into account. With this in mind, we focus on the counting stage of the analysis in Chapter 2. We employ a maximum likelihood based counting method, and compare it to existing methods. We show how our counting method can be used for identifying non-homogeneous samples and homogeneous regions within those samples. The new counting method is adapted to deal with the high levels of noise present in the data using smoothing methods.

1.3 Background on covariance estimation in high dimensions

Estimation of the covariance matrix has always been a fundamental problem in statistical inference, since the covariance matrix plays a key role in many data analysis techniques. Principal component analysis (PCA), classification by linear and quadratic discriminant analysis (LDA and QDA), inference about the means (e.g., setting confidence intervals on contrasts), and analysis of independence and conditional independence in graphical models all require an estimate of the covariance matrix or its inverse, also known as the precision or concentration matrix. Advances

in random matrix theory – from the classical results of [41] to the recent work of Johnstone and his students on the theory of the largest eigenvalues and eigenvectors [27, 28, 45, 30], and many others – allowed in-depth theoretical studies of the traditional estimator, the sample (empirical) covariance matrix, and showed that there are problems with the sample covariance in high dimensions. Specifically, under the normal assumption on the variables the sample covariance performs very poorly when the number of variables p is large relative to the sample size n. In particular, unless $p/n \to 0$, the sample covariance eigenvalues are over-dispersed and the eigenvectors are not consistent. It has also been shown that classification by LDA breaks down and reduces to random guessing when $p/n \to \infty$ [3]. These results have demonstrated that alternative ways of estimating the covariance matrix are needed in high dimensions. Next, we briefly review some known properties of the sample covariance eigenvalues and eigenvectors, and introduce alternative covariance estimators.

1.3.1 Known properties of covariance

Within our context of Raman spectroscopy (performing PCA on a spectral data set), we are most interested in the behavior of the sample eigenvectors and so we begin our discussion with sample eigenvectors in high dimensions. Johnstone and Lu showed that the sample eigenvector $\hat{e}_1$ corresponding to the largest sample eigenvalue is an inconsistent estimate of e_1, the population eigenvector corresponding to the largest population eigenvalue unless certain conditions are met [28]. This was shown for a factor model where the observations $\boldsymbol{X}_i$ are given by

$$\boldsymbol{X}_i = \mu + \sum_{j=1}^{m} (v_i^j)(e_j) + \sigma z_i, i = 1, ...n \tag{1.2}$$

where v_i^j is a $N(0,1)$ random effect and the z_i's are $N_p(0, I)$, i.i.d. noise vectors independent of the v's. It is assumed that $p/n \to \gamma$ as $n \to \infty$ and also that

$\|e_1(n)\| \to \varrho > 0$ as $n \to \infty$. Then Johnstone and Lu [28] showed that if $p/n \to \gamma > 0$, then $\liminf_{n\to\infty} E\angle(\hat{e}_1, e_1) > 0$ where $\angle$ represents the angle between the two vectors. When the number of dimensions $p = o(n)$, $\hat{e}_1$ is a consistent estimate of e_1.

There are some known results about the angles between sample and population eigenvectors for the leading eigenvalues in a "spiked" covariance model [45], [22], [23]. This model assumes that most of the population eigenvalues are 1, while a few are greater than 1 and well-separated from the rest. Paul [45] and others [22], [23] consider this "spiked" model where $p/n \to \gamma \in (0, \infty)$ as $n \to \infty$. The population model under consideration is that each $\boldsymbol{X}_i$, $i = 1, ..., n$ is $N_p(0, \Sigma)$, the $\boldsymbol{X}_i$'s are independent, and $\Sigma = \text{diag}(\lambda_1, \lambda_2, ..., \lambda_M, 1, ..., 1)$, where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M > 1$. For the following results, γ is also restricted to be in $(0, 1)$, but the results can be extended to $\gamma \in [1, \infty)$.

Hoyle and Rattray [22], [23] showed heuristically that if $1 < \lambda_j \leq 1 + \sqrt{\gamma}$, where λ_j is an eigenvalue of multiplicity one, then the cosine of the angle between e_j and $\hat{e}_j$ almost surely converges to 0. If instead, $\lambda_j > 1 + \sqrt{\gamma}$, then the limit is positive. Paul derives a similar result rigorously in [45]: Let d_j denote the j^{th} row of I_n (the n by n identity matrix). Then if $\lambda_j > 1 + \sqrt{\gamma}$ and is of multiplicity one, as $n \to \infty$, almost surely,

$$|\langle \hat{e}_j, d_j \rangle| \to \sqrt{\left(1 - \frac{\gamma}{(\lambda_j - 1)^2}\right) / \left(1 + \frac{\gamma}{\lambda_j - 1}\right)}, \tag{1.3}$$

where $\langle x, y \rangle$ is the inner product between x and y. If instead, $1 < \lambda_j \leq 1 + \sqrt{\gamma}$, then as $n \to \infty$,

$$\langle \hat{e}_j, d_j \rangle \to 0 \text{ a.s.} \tag{1.4}$$

These results indicate that the sample eigenvectors are not a good basis for PCA in high dimensions. There are known problems with the sample eigenvalues in high

dimensions as well. It is known that the sample eigenvalues, $\hat{\lambda}_j$'s, are more spread out than the population eigenvalues λ_j's, and the larger the ratio of p/n, the more spread out the sample eigenvalues are. For the largest sample eigenvalue $\hat{\lambda}_1$, if $p/n \to \gamma \leq 1$, then, [17]

$$\hat{\lambda}_1 \to (1 + \gamma^{1/2})^2 \text{ a.s.} \tag{1.5}$$

The distribution of $\hat{\lambda}_1$ has also been derived, for details, see [27].

In the "spiked" covariance model, it has been shown that a "threshold" value for the sample eigenvalues is $1 + \sqrt{\gamma}$ [45],[22],[23] . The sample eigenvalues behave differently depending on whether the corresponding population eigenvalues are larger or smaller than the threshold. Under the "spiked" covariance model, Paul showed that if $1 < \lambda_j \leq 1 + \sqrt{\gamma}$, then $\hat{\lambda}_j \to (1 + \sqrt{\gamma})^2$, almost surely as $n \to \infty$. If $\lambda_j > 1 + \sqrt{\gamma}$, then $\hat{\lambda}_j \to \lambda_j \left(1 + \frac{\gamma}{\lambda_j - 1}\right)$, almost surely as $n \to \infty$ [45]. Paul also derives an asymptotic distributional result for $\hat{\lambda}_j$ when $\lambda_j > 1 + \sqrt{\gamma}$ [45]. If $\lambda_j > 1 + \sqrt{\gamma}$ and λ_j is of multiplicity 1, then, as n and $p \to \infty$ and $p/n - \gamma = o(n^{-1/2})$,

$$\sqrt{n}(\hat{\lambda}_j - p_j) \Rightarrow N(0, \sigma^2(\lambda_j)), \tag{1.6}$$

where

$$p_j = \lambda_j \left(1 + \frac{\gamma}{\lambda_j - 1}\right) \; ; \; \sigma^2(\lambda_j) = 2\lambda_j^2 \left(1 - \frac{\gamma}{(\lambda_j - 1)^2}\right). \tag{1.7}$$

These results on sample covariance eigenvectors and eigenvalues demonstrate why alternative covariance estimators are needed in high dimensions. A discussion of alternative estimators follows.

1.3.2 Alternative estimators of covariance matrices

Regularized covariance estimators proposed as alternatives to the sample covariance in high dimensions can be loosely divided into two types. One large class of

methods covers the situation where variables have a natural ordering or there is a notion of distance between variables, as in longitudinal data, time series, spatial data, or spectroscopy. The implicit regularizing assumption underlying these methods is that variables far apart are only weakly correlated, and therefore one can improve on the sample covariance by taking advantage of the ordering [60, 24, 15, 5, 37]. Consistency and convergence rates have been established for some of these estimators in the high-dimensional setting; see more on this in Section 3.1. Apart from the normal assumption, these methods do not make any parametric assumptions on the covariance structure.

For regularizing the inverse covariance matrix in the case of ordered variables, a popular tool is the Cholesky decomposition. The modified Cholesky decomposition of the inverse covariance matrix Σ^{-1} is

$$\Sigma^{-1} = T'DT, \tag{1.8}$$

where D is a diagonal matrix and T is a lower-triangular matrix with ones along the diagonal. The elements of T and D have useful interpretations. If we let $\boldsymbol{X} = (X_1, ..., X_p)'$ be the time-ordered vector with mean 0 and covariance matrix Σ, and perform a linear regression of X_t on its predecessors $X_{t-1}, ..., X_1$, so that $X_t = \sum_{j=1}^{t-1} \phi_{tj} X_j + \epsilon_j$, then the elements of T are the $-\phi$'s and the diagonal elements of D are the variances of the ϵ's. Thus, the covariance estimation problem is reduced to a series of regressions [46].

Methods for regularization of the Cholesky factor include banding T [60] [5], adding an l_1 or l_2 penalty on the elements of T to the normal likelihood [24], and adding a nested Lasso penalty [37] which results in variable bandwidth banding of T.

For regularizing the covariance matrix itself rather than its inverse a simple and

popular approach is banding, or more generally, filtering [5] [15]. In filtering, the sample covariance matrix is replaced by an entry-wise (Schur) product of itself and a weight matrix, $\hat{\Sigma} * F_k$, where F_k is the weight matrix and k is a tuning parameter.

The simplest filtered estimator (referred to as simple banding) produces a k-diagonal estimate. The $(i,j)^{th}$ entry of this weight matrix is given by $F_k(i,j) = I(|i-j| \leq k)$, simply the indicator of whether the entry is on the first k subdiagonals. The covariance estimate resulting from simple banding may not be non-negative definite. Another choice of the weight matrix is

$$F_k(i,j) = \left(1 - \frac{|i-j|}{k+1}\right)_+ . \tag{1.9}$$

We call this weight matrix a triangular filter. Like banding, it results in zeros once $|i-j| > k$, but the resulting covariance estimate is non-negative definite, because the Schur product of two non-negative definite matrices is non-negative definite. Finally, we can consider a smooth weight matrix. One example where the resulting estimate is no longer banded, but is non-negative definite, is a Gaussian filter

$$F_k(i,j) = \exp(-\frac{|i-j|^2}{\tau(k)}). \tag{1.10}$$

To make Gaussian filtering comparable to banding, one could choose $\tau(k) = -k^2/\log\epsilon$, which ensures $F_k(i,j) \leq \epsilon$ once entries are more than k units away from the diagonal.

The tuning parameter k can be chosen by cross-validation, for example via the scheme proposed by [5], which involves computing all possible k banded estimates on a training set and comparing the estimates to the sample covariance on a test set. We give more details on banding in Chapter III.

Regularization methods we have discussed so far require an ordering of the variables. There are, however, many applications where such an ordering is not available: genetics, for example, or social, financial and economic data. Methods that are in-

variant to variable permutations, like the covariance matrix itself, are appropriate for such applications, resulting in a second class of methods of permutation-invariant estimators.

The class of permutation-invariant estimators includes estimators which shrink the sample eigenvalues. Regularizing large covariance matrices by Steinian shrinkage of eigenvalues has been proposed early on, originally by Stein in a Rietz lecture in 1975, and developed further by [19] and [12]. A more recent shrinkage estimator of Ledoit and Wolf replaces the sample covariance with a linear combination of the sample covariance and the identity matrix, with optimal (in a suitable sense) coefficients estimated from data [35]. Specifically, Ledoit and Wolf set out to find an estimator Σ^* such that $\Sigma^* = \rho_1 I + \rho_2 \hat{\Sigma}$ which minimizes expected quadratic loss $E\left[\|\Sigma^* - \Sigma\|^2\right]$, where $\|A\| = \sqrt{\mathrm{tr}(AA')/p}$, a modified Frobenius norm. The sample estimate $\hat{\Sigma}^*$ derived by Ledoit and Wolf is a consistent estimate for Σ^*, and it also has the same asymptotic risk [35]. Other shrinkage estimators include the empirical Bayesian estimator [19] which is also a linear combination of I and $\hat{\Sigma}$, but the weights are determined by n and p only, as well as the Stein-Haff [52] and minimax estimators [12] which replace the sample eigenvalues with shrunken versions. While shrinkage estimators are invariant to variable permutations, they do not affect the eigenvectors of the covariance, only the eigenvalues, and hence cannot be relied on to improve PCA. Shrinking eigenvalues also does not create sparsity in any sense, so these estimators cannot be used to analyze independence and conditional independence relations.

A number of non-shrinkage estimators have been developed. A penalized likelihood approach has been used to derive a sparse permutation-invariant estimator of the inverse covariance matrix based on adding an l_1 penalty to the normal likelihood

[10], [61], and [47]. An estimator for a factor analysis model with known factors has been proposed by [14]. Thresholding the sample covariance (i.e., setting small entries to zero), which is obviously permutation-invariant, has been analyzed in the high-dimensional setting by [4] and [29]. The permutation invariant thresholded covariance estimator is defined by

$$T_t(\hat{\Sigma}) = [\hat{\sigma}_{ij}\ I(|\hat{\sigma}_{ij}| \geq t)]\ , \tag{1.11}$$

where $I(\cdot)$ is the indicator function. We will include more detailed results on thresholding in Chapter III.

In Chapter III, we propose an approach that complements these covariance estimation methods by taking a different view on covariance structure. Rather than focus purely on permutation-invariant estimators when the data is unordered, we seek to find an ordering in the data that allows for application of covariance estimators which are not permutation-invariant. We focus on finding an ordering by using a manifold projection method, the Isomap, which makes simple banding appropriate and finds block-diagonal covariance structures.

CHAPTER II

Estimating Intrinsic Dimension for Chemical Components

To reliably extract pure component spectra in Raman spectroscopy, chemists need a counting procedure to determine the number of components to extract. In this Chapter, we adapt a non-linear dimension estimator, the maximum likelihood estimator (MLE) of intrinsic dimension, to the problem of determining the number of pure components in a mixture from Raman spectroscopy data, though the method can be applied to any spectral data and even more generally. We show how the intrinsic dimension corresponds to the number of pure components and introduce the MLE, as well as review existing counting methods in Section 2.1. In Section 2.2, we discuss the selection of a tuning parameter, and show on simulated mixtures that the MLE produces superior results compared to other methods, and is accurate even when minor components are present. In Section 2.3, we show how to handle low s/n ratios in the data to obtain accurate estimates, and illustrate by applying the MLE to two real datasets with high noise levels in Section 2.4. In Section 2.5, we show how computing local estimates at every image pixel can be used to automatically divide the image into homogeneous regions. Section 2.6 concludes with discussion.

2.1 Methods of estimating the number of pure components

Many extraction (SMCR) methods require an estimate $\hat{s}$ of the number of the components to be extracted, but few in fact make use of formal estimates. Examining a scree plot or plots of extracted eigenvectors by eye remains the prevalent method of analysis; and while the human eye can often be more accurate than an automated method, visual procedures are subjective, inconsistent across users, and time-consuming. In this section, we first review the methods that are currently available (even if not necessarily used) for estimating the number of components, and then present the new maximum likelihood method.

2.1.1 Current methods for estimating the number of pure components

Principal components analysis may be used for both the counting and extraction of the pure components. Recall that the eigenvalues of $\hat{\Sigma}$, $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq ... \geq \hat{\lambda}_p \geq 0$, represent the amount of variation in the data explained by the corresponding principal component. A scree plot of these eigenvalues can be used to estimate the true dimension by eye using the traditional "elbow" method. A somewhat more principled approach is to estimate the dimension of the data by the number of principal components that explain a pre-specified (large) fraction of the variance in the data:

$$\hat{s} = \arg\min_{s}\{s : \sum_{j=1}^{s} \hat{\lambda}_j \geq (1-\varepsilon)\sum_{j=1}^{p} \hat{\lambda}_j\}. \tag{2.1}$$

Choosing an appropriate fraction $1-\varepsilon$ generally depends on the amount of noise in the data, which is not known in advance. This is one difficulty with using PCA for estimating dimension. Typically the fraction chosen is at least 90%; we use $\varepsilon = 0.01$ throughout this Chapter.

The Malinowski's F-test [39] was introduced in the chemometrics literature to differentiate between significant and noise eigenvectors in PCA. The sum of the

eigenvalues $\sum_{j=1}^{p} \lambda_j$ can be decomposed into pieces representing significant and noise eigenvalues, with the number of significant eigenvalues providing an estimate of the number of pure components. The test starts from the smallest eigenvalue λ_p and goes through the eigenvalues in increasing order until it finds the first significant one. Once one eigenvalue has been determined to be significant, all larger eigenvalues are also considered significant. The Malinowski's F-statistic for testing the significance of the s-th eigenvalue is given by

$$F_s = \frac{\lambda_s}{\sum_{j=s+1}^{p} \lambda_j/(p-s)}. \tag{2.2}$$

Under the null hypothesis that the s-th eigenvector is noise, Malinowski argued that F_s has an F distribution with 1 and $p-s$ degrees of freedom. The estimated number of components based on the F-test can be computed as

$$\hat{s} = \min_s \{F_s > f_{1,p-s}(1-\alpha)\}, \tag{2.3}$$

where $f_{1,p-s}(1-\alpha)$ is the $(1-\alpha)$ critical value for the $F(1, p-s)$ distribution. Again, the choice of α, like the choice of ε in (2.1), is at the user's discretion; we will use $\alpha = 0.01$ throughout. In any case, since the test is repeated until the first significant eigenvalue is found, this creates a multiple testing problem (see, e.g., [53]), and the actual overall significance level will be higher than α. A comparison of similar techniques and a modified F-test can be found in [40].

2.1.2 Maximum likelihood estimation of dimension

The maximum likelihood estimator (MLE) of intrinsic dimension [36] was originally proposed for estimating the intrinsic dimension s of data $\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_n$ which are measured as p-dimensional vectors, but in fact lie on a manifold, that is, in an s-dimensional subspace of the p-dimensional space, with s typically much smaller

than p. Here we show how the MLE of intrinsic dimension can be applied to spectroscopy data and resolve practical issues that arise in the process, such as choosing the tuning parameter and dealing with high levels of noise in the data.

Let the Euclidean distance from a fixed observation x to its j^{th} nearest neighbor,x_j in the sample be denoted $T_j(x)$.

The MLE is derived by fixing an arbitrary point x and assuming the density $f(x)$ of the observations is constant in a small sphere of radius R around x, denoted $S_x(R)$. Then, define $N(t,x)$, a process which counts the number of observations within distance t of x, as

$$N(t,x) = \sum_{r=1}^{n} I(X_r \in S_x(t)), 0 \leq t \leq R. \tag{2.4}$$

With the assumption of constant density around x, the process N is a Poisson process whose rate can be calculated explicitly and used to write the log-likelihood of the process. Then, the MLE can be found by solving the likelihood equations. The final estimate $\hat{s}_R(x)$ is computed as

$$\hat{s}_R(x) = \left[\frac{1}{N(R,x)} \sum_{j=1}^{N(R,x)} \log \frac{R}{T_j(x)} \right]^{-1} \tag{2.5}$$

Notice that for a given radius R, there may be different numbers of nearest neighbors within radius R for each x. It is possible to rewrite the MLE based on considering a fixed number k of nearest neighbors at each point, rather than a fixed neighborhood radius R, which is often more intuitive and hence easier to pick. The estimate is then rewritten as

$$\hat{s}_k(x) = \left[\frac{1}{k-1} \sum_{j=1}^{k-1} \log \frac{T_k(x)}{T_j(x)} \right]^{-1} \tag{2.6}$$

This estimate can be made unbiased by replacing $k-1$ with $k-2$. We will use the fixed k version with $k-2$ (2.6).

Equation (2.6) allows us to obtain an estimate of the intrinsic dimension at every data point $\boldsymbol{X}_i$. Then the global estimate (or an estimate over a particular region) can be computed by averaging the local estimates over the entire data set (or the region in question). The global MLE for the whole dataset is given by

$$\hat{s}_k = \frac{1}{n}\sum_{i=1}^{n} \hat{s}_k(\boldsymbol{X}_i) \tag{2.7}$$

We discuss the choice of k and the sensitivity of the estimator to k in Section 2.2; usually k is chosen to be a relatively small number, and the estimator is robust to the choice of k. Note also that in general neither (2.6) nor (2.7) give an integer estimate of dimension; in practice (2.7) is rounded to the nearest integer.

For the specific context of Raman spectroscopy, the intrinsic dimension of the space generated by the spectra is not in itself an estimate of the number of pure components present in the mixture. Consider the following: a "mixture" of two points generates a line, which has intrinsic dimension one, and three points generate a plane, which has intrinsic dimension two. Since the MLE estimates the dimension of the manifold generated by the mixture (the line or the plane), we add one to the MLE of intrinsic dimension in order to obtain the MLE for the number of pure components present in the mixture.

An advantage of the MLE is that it automatically generates an estimate of the number of components at every data point (pixel). While some variability in local estimates is expected even in homogeneous mixtures due to noise, a lot of variability indicates that the mixture under examination is likely not homogeneous. Another potential application is using the local MLEs for testing for mixture homogeneity. An application of local MLEs to segmenting the Raman image into homogeneous regions is presented in Section 2.5.

2.2 Simulation results

In this section we investigate the performance of all three estimators (PCA, Malinowski F-test, and the MLE) on simulated mixtures obtained from real spectra. Each pure material was scanned separately, and their individual spectra are combined into a $s \times p$ pure component spectra matrix A, where s is the number of pure components and p the number of different wavenumber values at which the spectra were measured. To generate n mixture spectra $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$, which we combine into a single $n \times p$ matrix X, we first generate a random $n \times s$ matrix of component weights W where the distribution of the weights in W is described in detail below. The mixture spectra are then generated according to the linear additivity of the system with i.i.d. Gaussian noise, ϵ, as discussed in the Introduction:

$$X = WA + \epsilon. \tag{2.8}$$

2.2.1 Data description

For generating simulated mixtures, we used two separate test sets of pure component spectra generously provided by the Morris lab. The first set consists of five plastics and one bovine bone spectra collected on the visible Raman system, which are quite dissimilar from each other and should be easy to discriminate. The pure components in this set are polyethylene (PE), Delrin (Del), polystyrene (PS), poly(methyl methacrylate) (PMMA), bovine bone (Bone), and Teflon (Tefl), measured at $p = 512$ wavenumber values in the range $700 - 1600$ cm^{-1}. For details on the visible Raman system and experimental conditions, see the Appendix. The pure spectra were rescaled to have maximum intensity 1 (to compensate for different amount of material present in each scan) and are shown in Figure 2.1, with distinct spectral features of each component clearly visible by eye, although there is some

overlap in peak locations (wavenumber location, also called Raman shift), as shown in Table 2.1. Table 2.1 shows, however, that each spectra has at least one peak that allows it to be easily discriminated from the others. Note that in each spectrum, there is a fluorescence background present which varies across the spectrum. This background is usually removed before pure component spectra are extracted from the mixture, but the procedure to remove the background is subjective, time consuming, and not always fully successful. For testing the counting methods, we leave the background present as it does not affect the number of pure components present.

Table 2.1: Spectral Peak Locations for Dissimilar Spectra.

Pure Component	Approximate Main Peak Locations
Del	925, 1100, 1350, 1400, 1500
PE	825, 875, 990, 1000, 1050, 1175, 1340, 1450
PS	1000, 1030, 1200, 1600
Tefl	730, 1220, 1300, 1400
Bone	960, 1450
PMMA	800, 990, 1000, 1140, 1180, 1200, 1250, 1450

The second set of spectra contains five spectra of a fractured mouse tibia bone and one plastic (PMMA) collected on the NIR system (see the Appendix for details) and measured at 815 different spectral values. PMMA is used to embed the fractured bone, and the five bone spectra are measured at different distances along the mouse bone, gradually moving away from the fracture. The five bone spectra vary with the distance away from the fracture but these differences in the spectra are minute (see Figure 2.2). The further away from the fracture, the less the spectra differ; in fact, the last two bone spectra, measured at 900 μm and 1100 μm away from the fracture, are identical. Hence, this set of six spectra contains only five distinct pure components. It is nearly impossible to see any differences between the spectra in Figure 2.2, but there are changes to the amide and phosphate components in the bone, which occur around wavenumbers 1240 to 1270 and also from 1600 to 1700.

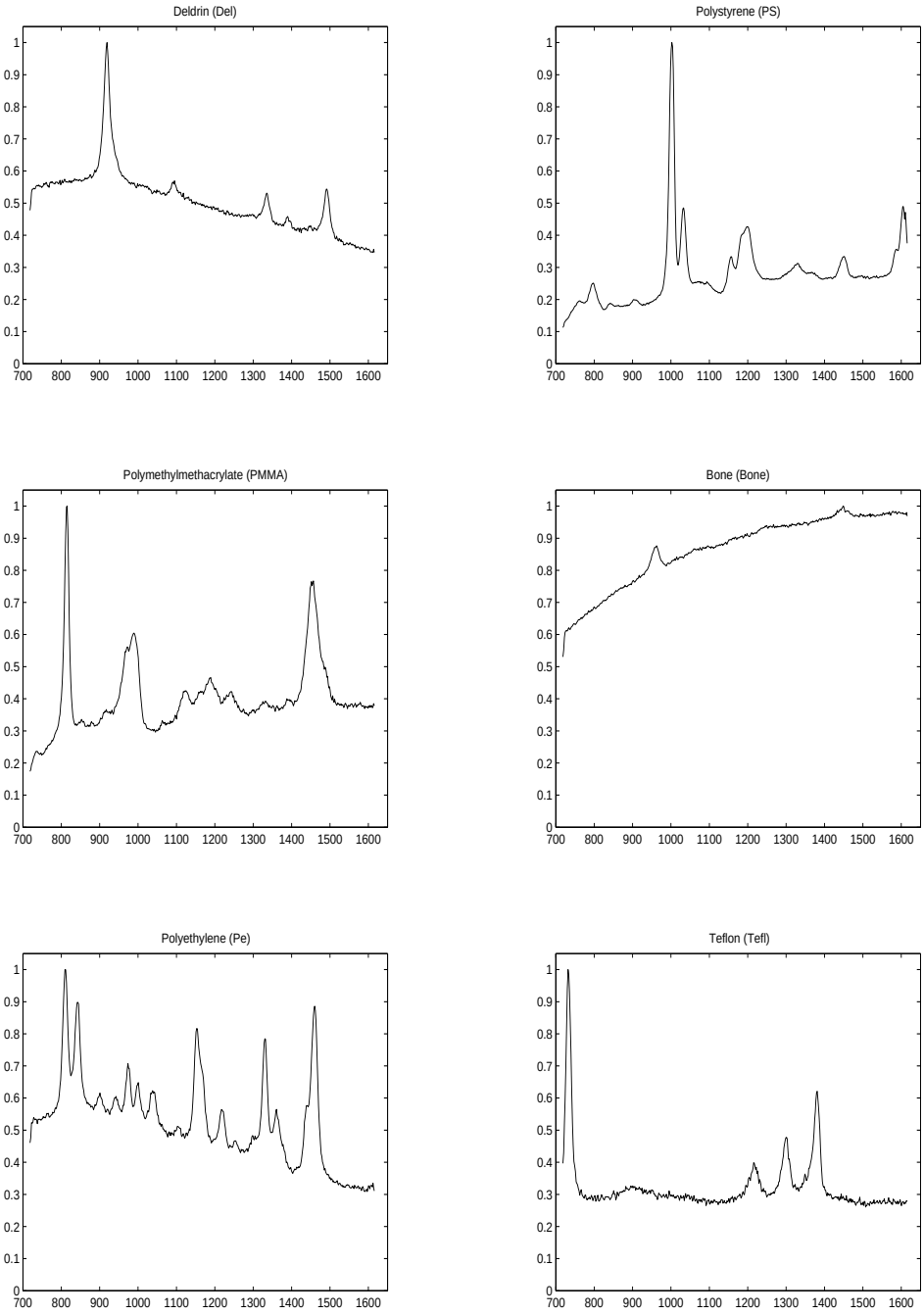


Figure 2.1: Test set 1 (dissimilar spectra). The pure component spectra of plastics and bovine bone are rescaled to maximum intensity 1; horizontal axis shows Raman shift (cm^{-1}).

We examined many combinations of weight matrices and noise levels in our simulations. For the results presented here, for each set of six spectra we always select four

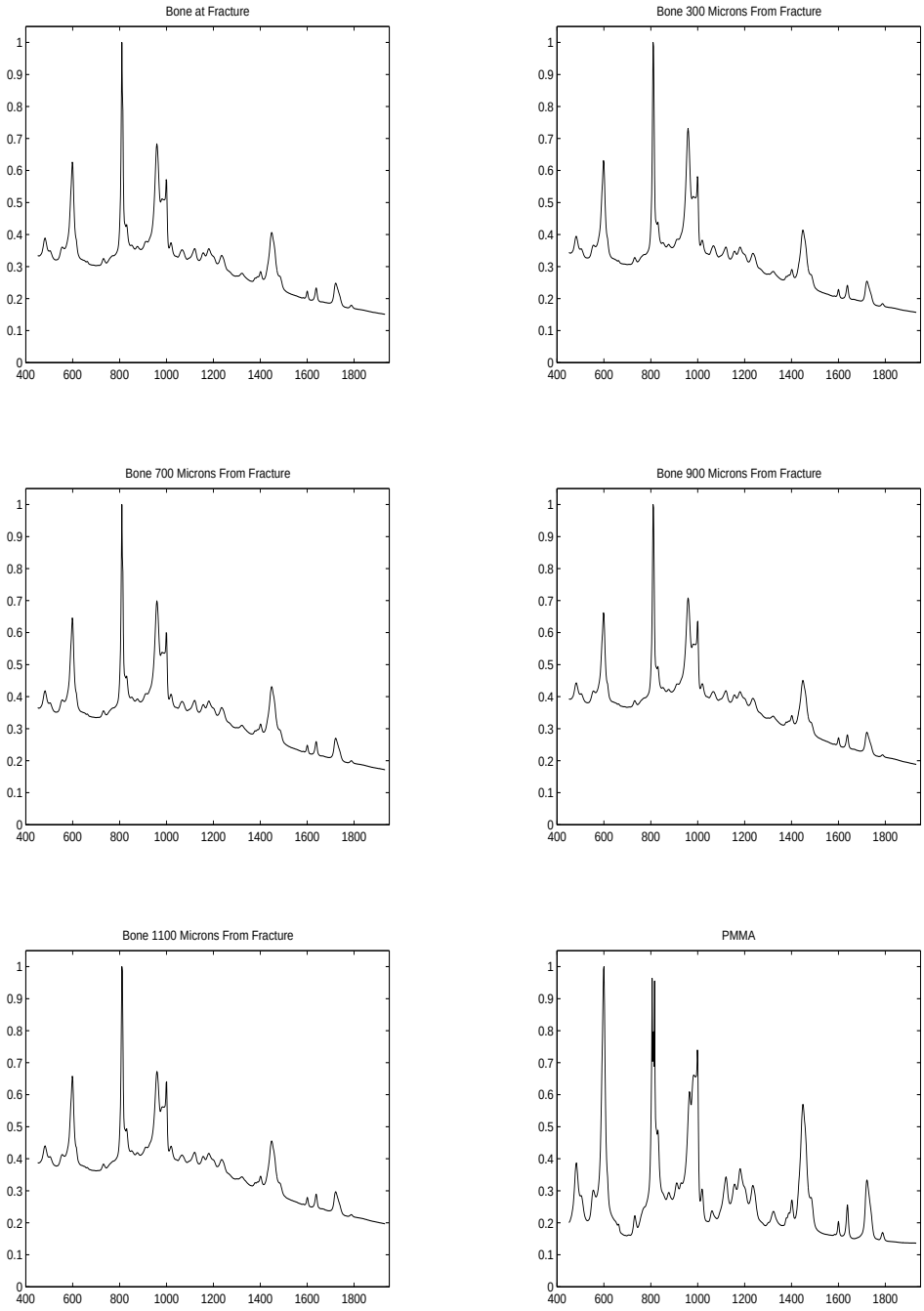


Figure 2.2: Test set 2 (similar spectra). The pure component spectra of mouse bone and PMMA are rescaled to maximum intensity 1; horizontal axis shows Raman shift (cm^{-1}).

major components (Delrin, polystyrene, PMMA, and bone for set 1, and PMMA and the three bone spectra closest to the fracture for set 2). The remaining two spec-

tra in each set were used as minor components, to test whether the methods are able to pick components present in small amounts. For the setting with just four major components, each component's weight was drawn uniformly from the interval (.15,.30). When minor components were added at 10% and 5% levels, major component weights were drawn uniformly from (.15,.25), and minor weights from (.05,.15) and (.03,.07), respectively. In Section 2.3, we push the minor components level down to 1%, in which case we draw the major weights from (.20,.30), and the minor weights from (.00,.02). In each case, weights are randomly drawn from a uniform distribution on each interval were rescaled to sum to 1. Gaussian noise was added at a q% level, which means that the noise has mean 0 and variance $(0.01q)^2$. We also studied settings with only three major components and obtained similar results.

2.2.2 Results

The results presented for each spectra set (Tables 2.2 and 2.3) are representative of all simulations we performed. In each case, the number of pixels is $n = 3600$ (60 by 60 image), and the estimated numbers of components are averaged over 100 replications. We compare the MLE at $k = 20$ (the choice of tuning parameter is discussed below), PCA at 99% variance explained, and Malinowski's F-test at 1% significance level. If the estimate is given as an integer (e.g., 4), it means there was no variation in the estimate across the 100 replications. If there was variation but the average came out to be 4, it is given as 4.0. Although global MLEs are usually rounded to integer values in practice, for these results, the global MLEs are not rounded to integer values before averaging in order to produce more accurate average results.

The levels of noise are chosen to show where the MLE starts picking up noise components and overestimates the number of pure components. For both test sets,

Table 2.2: Test set 1 (dissimilar spectra): estimated number of pure components.

No. comp.	4				6				6			
Minor level (%)	0				10				5			
Noise level (%)	0	.05	.1	.3	0	.05	.1	.3	0	.05	.1	.3
MLE	4.0	4.2	4.9	10.9	5.6	5.8	6.4	12.2	5.5	5.8	6.6	15
PCA (99%)	4	4.0	5	5	5	5	5.2	6	5	5	5.1	5.1
F-test (1%)	4	4	4	1	6	6	1	1	6	6	1	1

Table 2.3: Test set 2 (similar spectra): estimated number of pure components.

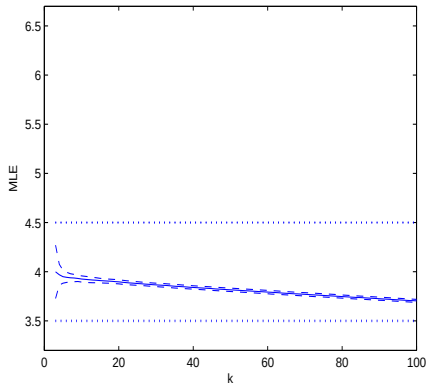
No. comp.	4				5				5			
Minor level (%)	0				10				5			
Noise level (%)	0	.005	.01	.03	0	.005	.01	.03	0	.005	.01	.03
MLE	3.9	4.1	4.7	10.2	4.7	4.9	5.5	11.7	4.5	4.5	5.7	14
PCA (99%)	3	4	4.0	5	4	5	5	6	4	5	5.1	6
F-test (1%)	4	4	3	2	6	3	3	2	6	3	2	2

it is clear that at low noise levels the MLE estimate performs as well as or better than the other two methods. Note that PCA fails to obtain the correct number of components even with no noise if minor components are present. The F-test can obtain the correct estimates with no noise, but fails once noise is added. In fact, as noise levels increase, all methods begin to suffer, but the MLE is more sensitive to noise than PCA. As expected, it takes less noise for the estimates to break down when the spectra are very similar than when they are different. Otherwise, the same pattern holds for both test sets. The issue of dealing with high levels of noise is addressed in Section 2.3.

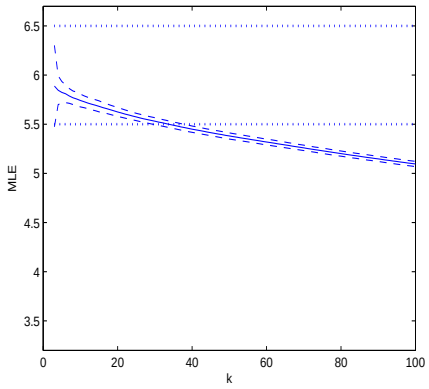
2.2.3 Choice of the tuning parameter k for the MLE

The MLE estimate requires choosing a value of k, the number of nearest neighbors around each point on which the local estimator is based. The impact of k on the estimate was examined via simulation. Figure 2.3 shows the global MLE estimate plus and minus one standard deviation versus k over 100 replications for three different settings. Keeping in mind that in practice the estimate is rounded to the

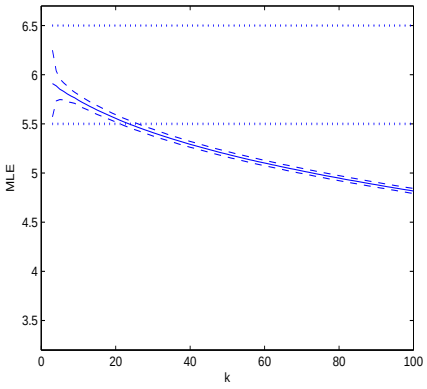
nearest integer, we see that the MLE estimates vary very little across replications (small standard deviation), and are fairly robust to the choice of k. The derivation of the MLE involves an approximation that requires k to be small relative to n, and smaller values of k reduce the amount of computation; on the other hand, very small values of k may lead to too much variability in local estimates. On the balance, we chose $k = 20$ and kept it constant for all simulations and real data applications. We note that for other applications the user may choose to average MLEs derived for a range of k values.



(a) 4 major components



(b) 4 major and 2 minor components at 10%



(c) 4 major and 2 minor components at 5%

Figure 2.3: Sensitivity to k for sample size, $n = 1000$, and results averaged over 100 replications. Dashed lines show the range where the estimate is rounded to the correct value.

2.2.4 Impact of image size

The behavior of all estimators can in general be affected by the amount of data available. In general, larger images are better since they contain more information about the mixture. To study this effect, we performed simulations with smaller image sizes of $n = 400$ and 1000 ($n = 3600$ in Tables 2.2 and 2.3). Table 2.4 gives results for $n = 400$. For quick reference, Table 2.5 compares $n = 400$ with $n = 3600$ directly.

Table 2.4: Test set 1 (dissimilar spectra): estimated number of pure components for $n = 400$.

No. comp.	6								6							
Minor level (%)	10								5							
Noise level (%)	0	.01	.03	.05	.1	.15	.3	.5	0	.01	.03	.05	.1	.15	.3	.5
MLE (k=5)	5.4	5.5	5.5	5.6	5.9	6.6	9.7	16.7	5.7	5.7	5.9	6.1	7.1	8.7	17.1	33.1
MLE (k=10)	5.3	5.3	5.3	5.4	5.7	6.1	8.5	13.7	5.7	5.7	5.8	6.0	6.7	8.0	14.4	27.7
PCA (99 %)	5	5	5	5	6	6	7	10	5	5	5	5	5	6	6	7

Table 2.5: Test set 1 (dissimilar spectra): sample size comparison (rounded to nearest integer).

No. comp.	n	6								6							
Minor level (%)		10								5							
Noise level (%)		0	.01	.03	.05	.1	.15	.3	.5	0	.01	.03	.05	.1	.15	.3	.5
MLE (k=10)	400	5	5	5	5	6	6	8	14	6	6	6	6	7	8	14	28
	3600	6	6	6	6	7	8	14	28	6	6	6	6	7	9	18	36
PCA (99 %)	400	5	5	5	5	6	6	7	10	5	5	5	5	5	6	6	7
	3600	5	5	5	5	5	6	6	7	5	5	5	5	5	5	5	7

The standard errors of the MLE estimate decrease as the image size increases, as expected. When the noise level is high, the MLE estimates for $n = 3600$ are much higher than the $n = 400$ estimates. This is expected: the noise is overwhelming the signal, and as the estimate becomes more accurate for larger n, it picks up more noise components. This issue is addressed in detail in Section 2.3. Finally, we note that for large image sizes, the computational complexity associated with the singular value decomposition makes the MLE, which only requires finding the nearest neighbors, a more attractive choice than both the PCA and the F-test.

2.3 Dealing with high levels of noise

Simulations showed the MLE method is sensitive to noise, which is common in real data. When high levels of noise are present, smoothing the data before applying the procedure can enhance the performance of the estimator. There are two types of smoothing one can consider: smoothing along each spectrum, and smoothing spatially across the image. Individual spectra can be smoothed, for example, with a Blackman-Harris filter, a signal processing tool available in many software libraries [21]. We found that smoothing the spectra helps somewhat, but is less efficient than spatial smoothing. When both methods of smoothing are combined, the effect is the same as that of spatial smoothing alone. Therefore we choose not to smooth the individual spectra at all, which allows us to better preserve the peaks and other spectral features.

Spatial smoothing can be achieved via a convolution of the image X with a filter matrix Q. At each spectral wavenumber l and pixel location (x, y), we compute

$$X_Q(l, x, y) = \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} X(l, u, v) Q(x - u, y - v), \tag{2.9}$$

where values of matrices outside of the valid index range are defined to be zero.

The filter matrix, generally speaking, averages the values around (x, y), and many choices are possible (simple averaging, weighted averaging over a fixed window, exponentially decaying weights over the whole image, etc). We found a simple spatial moving average (MA) filter to perform very well in this context. The MA filter replaces the value at each pixel with the average of pixel values in a $w \times w$ window around it. The window size is taken to be odd for convenience, $w = 2m + 1$, and the

convolution formula reduces to

$$X_{MA}(l,x,y) = \frac{1}{w^2} \sum_{u=x-m}^{x+m} \sum_{v=y-m}^{y+m} X(l,u,v). \tag{2.10}$$

We only compute this for x and y that are at least m pixels away from the edges of the image and discard the rest. Alternatively, the convolution could be modified to "pad" the edges so that no values are discarded.

Finally, we investigated smoothing across neighbors in terms of spectral similarity rather than spatial location. This technique is often used for data on a manifold, for smoothing over manifold neighbors. We have investigated a moving average smoother over "spectral" neighbors and the iterative locally linear smoothing technique [62]. The results were found to be inferior to spatial smoothing. The spatial moving average is therefore our final choice and the only technique we present formal results for.

2.3.1 Choice of window size for smoothing

The window size w for the MA smoother is another tuning parameter to be selected. We investigated many window sizes in extensive simulations; Figure 2.4 shows a representative plot of MLE estimates applied to smoothed data for several noise levels as a function of window size. The setting is dissimilar spectra with four major components and two minor components at 10%.

An important conclusion from Figure 2.4 is that it is better to over-smooth than to under-smooth. For consistency, we selected the first window size for which the MLE estimates at all noise levels obtained a correct estimate, which is $w = 9$ (a 9×9 window), which we will use in all results below. In general, we recommend this as a rule-of-thumb starting value, since over-smoothing does not appear to present a problem. Plots of the MLE as a function of window size like the ones in Figure 2.4

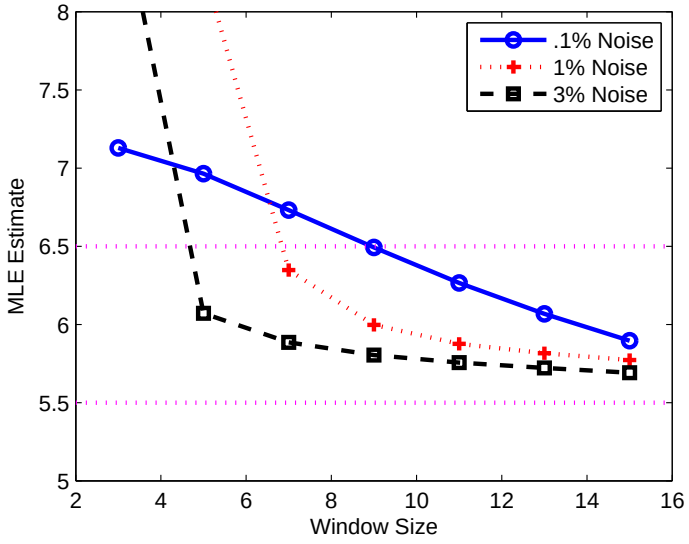


Figure 2.4: MLE estimates on smoothed data as a function of MA window size at various noise levels for dissimilar spectra with 4 major + 2 minor components at 10%, $n = 3600$, $k = 20$. The horizontal lines show the range where the estimator is rounded to the correct value.

can also be investigated for different applications and w selected as the point where the plot "levels off".

2.3.2 Simulation results with high levels of noise

To examine the impact of smoothing on all three estimators, we performed simulations with increased noise levels. All settings were the same as before, except Gaussian noise was added at higher levels (.1%, 1%, or 3%). The results are shown in Table 2.6 (dissimilar spectra, all six are distinct) and Table 2.7 (similar spectra, 5 out of 6 are distinct). A 9×9 window was used for spatial smoothing of the 60×60 image.

The results show that, while the MLE estimate is the most sensitive to high levels of noise, it is also the only one that is able to obtain correct (on average) estimates on smoothed data. Both PCA and the F-test do not come close to the true number of components whether the data are smoothed or not, which suggests that generally

Table 2.6: Dissimilar Spectra smoothing results: $n = 3600$, $k = 20$, $w = 9$, 4 major + 2 minor components, results averaged over 100 replications

Minor level (%)	10			5			1		
Noise level (%)	.1	1	3	.1	1	3	.1	1	3
MLE	6.7	58.7	101.2	7.1	65.7	104.6	8.2	74.5	110.7
Smoothed MLE	6.0	5.4	5.3	6.1	5.5	5.2	6.4	5.5	5.2
PCA	5	31	38	4	31	37	4	31	36.1
Smoothed PCA	1	5	20	1	4.1	19	1	4.6	17.9
F-test	1	0	0	1	0	0	1	0	0
Smoothed F-test	1	1	1	1	1	1	1	1	1

Table 2.7: Similar Spectra Smoothing Results: $n = 3600$, $k = 20$, $w = 9$, 4 major + 2 minor components (5 distinct), results averaged over 100 replications.

Minor level (%)	10			5			1		
Noise level (%)	.1	1	3	.1	1	3	.1	1	3
MLE	64.7	132.9	153.1	59.9	128.5	155.6	82.1	137.6	158.4
Smoothed MLE	6.5	5.4	5.4	6.0	5.4	5.6	5.7	5.5	5.6
PCA	2	33	37	2	34	38	2	34.1	40
Smoothed PCA	1	14	24.9	1	14	26	1	14.1	26.9
F-test	1	0	0	1	0	0	1	0	0
Smoothed F-test	1	1	1	1	1	1	1	1	1

their results cannot be trusted for data with high noise levels. The MLE, on the other hand, performs well on smoothed data even when the noise level (3%) is higher than the amount of minor components present (1%).

2.4 Applications to real data

The simulation results in the previous section were based on using real spectra which were artificially combined into a simulated mixture. In contrast, here we apply the proposed methodology to Raman images of real specimen (see Appendix for experimental details). Based on results in Sections 2.2 and 2.3, we set the MLE tuning parameter to $k = 20$ and apply a spatial moving average smoother with a window size of 9 as a preprocessing step. For a fair comparison, we report the results for the other two estimators for both raw and smoothed data.

2.4.1 Dataset 1: PMMA with two different curing times

Our first real dataset is a 130 by 30 image of a polymer (PMMA), with Raman spectra measured at 512 spectral values. The specimen was obtained by combining two Koldmount mixtures at different stages of polymerization. Koldmount is commonly used to embed biological specimens. The solid component and the liquid are mixed; the reaction proceeds quickly to produce a translucent material. In this dataset, "fresh" PMMA (three minutes after mixing) was layered onto partially cured PMMA (eight minutes after mixing). The image was taken at the interface. The details of the experiment are given in the experimental section.

The initial mixture contains four chemical components – PMMA particles, unreacted monomer, and two initiators (trace amounts). The mixture is not necessarily homogeneous, especially because the reactions continued as Raman measurements were taken. The volume fraction of unreacted monomer depends on the reaction rate and the time (post-mixing) at which any particular pixel was imaged. As a result, substantial variation in the proportions of the two major components is expected. This relatively simple system serves as an excellent test case for estimating the number of pure components.

Table 2.8: Real data results for the three estimators applied to raw and smoothed (denoted by Sm) images.

Data	MLE	MLE(Sm)	PCA	PCA(Sm)	F-test	F-test(Sm)
PMMA with 2 curing times	50	4	6	4	1	1
Bone embedded in PMMA	47	5	5	4	9	9

The results are shown in Table 2.8. If no smoothing is applied as a pre-processing step, all estimators give incorrect results. With smoothing, the MLE and PCA both pick up 4 components. The F-test only detects 1 component, with and without smoothing. The MLE on smoothed data was the same for a range of values of k and

the moving average window size w.

2.4.2 Dataset 2: Bone

We also examined a 300 by 50 bone image consisting of Raman spectra measured at 512 spectral values. The bone was a murine femur, embedded in PMMA resin. A transverse section was chosen at the edge of the bone to include both bone and resin in the field of view. However, no significant concentration of resin was seen in the data; the reduced collection efficiency at the edges of the CCD camera left the section known to contain PMMA relatively dark. Based on previous experiments on similar specimens, the presence of PMMA distributed within the bone tissue is still expected. Thus, there are at least three major components expected in the data – PMMA, bone mineral, and bone matrix. There may also be additional bone components, depending on age and damage [58]. Here, MLE and PCA obtain 5 and 4 components respectively on the smoothed data, with the F-test obtaining 9.

Even though in real data, unlike in simulations, we do not know the correct answer exactly, the MLE appears to perform well on smoothed data. For these datasets, PCA and the MLE give comparable results; however, results in Tables 2.6 and 2.7 suggest that in general the MLE of smoothed data is likely to be more reliable than PCA when high levels of noise are present.

2.5 Using local dimension estimates for image segmentation

The MLE in (2.6) is computed at every pixel, but so far we have been using the global average (2.7) as the estimate for the number of components. We can also use the pointwise estimates for other tasks, such as finding regions with different numbers of components (areas with more components may be more chemically interesting), or evaluating homogeneity of the mixture. To illustrate the potential of

local estimators, we demonstrate how they can be used to segment an image into regions with homogeneous numbers of components.

2.5.1 Image segmentation technique: normalized cuts

Normalized cuts, or Ncuts [50] is an image segmentation procedure that divides an image into regions by both maximizing similarity of points within each region and maximizing dissimilarity between regions. The procedure treats segmenting the data into regions as a graph partitioning problem. Pixels form the set of vertices V, and weights $w(x, y)$ on the edges between points $x \in V$ and $y \in V$ represent a measure of similarity between x and y. The partition of V into two non-overlapping sets A and B is then found by minimizing a function of the data called the normalized cut. The normalized cut between two regions A and B is defined to be

$$\text{Ncut}(A, B) = \frac{\text{cut}(A, B)}{\text{assoc}(A, V)} + \frac{\text{cut}(A, B)}{\text{assoc}(B, V)}, \tag{2.11}$$

where association and cut are defined as

$$\text{assoc}(A, V) = \sum_{x \in A, y \in V} w(x, y); \qquad \text{cut}(A, B) = \sum_{x \in A, y \in B} w(x, y). \tag{2.12}$$

The idea is to find A and B that have the least similarities between them (minimize the cut) but penalize for segmenting out the regions that are not well connected within themselves – that is the purpose of normalizing by the association. If the normalization is omitted, the segmentation will tend to cut off single points. Note that this problem can be restated in terms of normalized association. When solving for the partition, minimizing the normalized cut means minimizing the association between the split groups while maximizing the normalized association means searching for a segmentation that leaves the split groups as similar as possible internally. It turns out that these two formulations are equivalent [50].

In order to implement the normalized cuts, we need an appropriate measure of similarity between pixels. For regular image segmentation (original context of the application), Shi and Malik [50] used a similarity measure based on spatial distance between pixels and differences in their brightness values. For our application, we propose a similarity measure which reflects both the spatial distance between pixels and the differences in the estimated number of components at each pixel. For a pair of data points, where x is located at (i_x, j_x) and y is located at (i_y, j_y), with $\hat{s}_x$ and $\hat{s}_y$ the number of components estimated at each point, we define the weight on the edge between x and y to be

$$w(x,y) = \exp\left(-\frac{(\hat{s}_x - \hat{s}_y)^2}{\sigma_1} - \frac{(i_x - i_y)^2 + (j_x - j_y)^2}{\sigma_2}\right). \tag{2.13}$$

The scaling factors σ_1 and σ_2 can be used to vary how much importance is given to spatial proximity versus the similarity in the number of components. They can also be used to resolve scaling issues if the two measures combined are not on the same scale. In this case, the two components in (2.13) are on the same scale, and we set $\sigma_1 = \sigma_2 = 1$ in the results shown below.

The normalized cut problem itself is NP-hard, but a relaxation can be solved efficiently through a generalized eigenvalue problem. Here we briefly summarize the algorithm (see [50] for further details). Let $d(i) = \sum_j w(i,j)$ and let D be a diagonal matrix with diagonal d. Let $W = [w(i,j)]_{1 \le i,j \le n}$ be the symmetric matrix of edge weights. Finally, let v be an $n \times 1$ indicator vector with $v(i) = 1$ if the i-th data point is in A and -1 if it is in B. Now let

$$b = \sum_{x \in A} d(x) / \sum_{x \in B} d(x), \tag{2.14}$$

$$c = (1+v) - b(1-v). \tag{2.15}$$

In effect, c is a continuous approximation to v. Shi and Malik show that the solution

to the normalized cuts problem can be found by solving the eigenvalue system

$$(D - W)c = \lambda Dc. \tag{2.16}$$

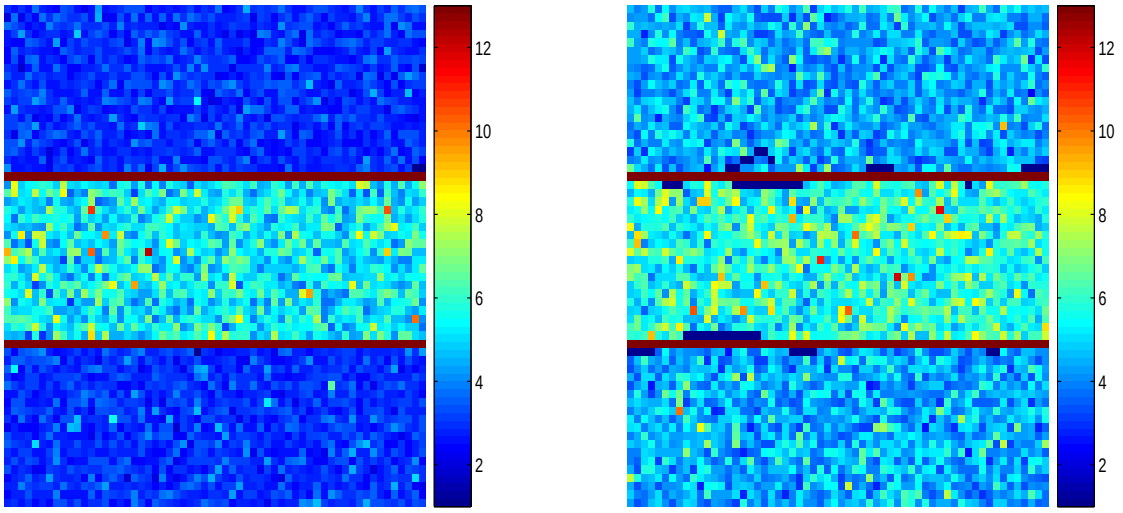
The second smallest eigenvector of the system gives the first split, with the partition based on the signs of the entries in the eigenvector. The procedure can then be repeated to split the two regions A and B further.

2.5.2 Segmentation results

For simulations, we divided a 60×60 image into three 20×60 horizontal strips with 3, 6, and 3 components, respectively. The three major components were bone, PMMA, and Delrin, with polystyrene added as a major component and Teflon and polyethylene as minor components at 10% in the middle region. Then we generated two images, with two levels of noise (.1% and .3%). The local MLEs were computed at each pixel with $k = 20$. To keep the example straightforward, we set the levels of noise low enough so that smoothing was unnecessary. Similar results were obtained with MA smoothing as a preprocessing step. The local estimates were used in (2.13) to create the weights and normalized cuts were applied to segment out three regions. The resulting segmentations are shown in Figure 2.5. The procedure correctly finds the three regions we built into the data generation. For low noise, the average MLE in each region was 3.0, 5.4, and 3.0, respectively. For higher noise, the region averages were 4.4, 5.9, and 4.5. We can see that the MLE performs better at low noise levels in terms of obtaining the correct number of components (recall that no smoothing was applied to the data), but the segmentation is still correct at the higher noise level.

Since normalized cuts require the user to specify the number of regions to be segmented, we experimented with asking for more than three regions. In this case

the procedure segments out additional very small areas, but the main three regions are still clearly visible. Hence this segmentation procedure can be used even if little is known a priori about the number of different regions in the image. Also note that we did not incorporate spectral similarity into the measure (2.13), and it is of course possible to have spectroscopically different regions with the same total number of pure components. This example was intended to illustrate the potential of local estimates; an in-depth investigation of their applications to segmentation and homogeneity testing is a subject for future research.



.1% Gaussian noise .3% Gaussian noise

Figure 2.5: Segmentation results with .1% and .3% Gaussian noise; the pixel color values show local dimension estimates. Estimated boundaries are shown in dark blue where they do not match the true boundaries shown in red.

2.6 Conclusions

Determining the number of pure components present in a mixture is an important step before extracting and identifying the chemical components, and having a reliable estimate of how many components to extract leads to more objective and

accurate data analysis, as well as reduces the amount of human visual inspection and other manipulations. We have shown the maximum likelihood estimator of intrinsic dimension, designed for general data on non-linear manifolds, can be successfully applied to this problem, and tested its performance on both real and simulated mixtures of Raman spectra. The method is robust to the choice of the tuning parameter k and outperforms PCA and the Malinowski's F-test, particularly when minor components are present and/or the signal-to-noise ratios are low. When the noise level is very high, additional preprocessing via spatial smoothing has been shown to produce good results.

The MLE of intrinsic dimension is a general method and is likely to find applications in other areas of chemometrics and in other fields. One advantage of the MLE is that it automatically generates an estimate of the number of components at every data point (in case of images, at every pixel). Here we illustrated the potential of these local estimates by using them to segment the specimen into homogeneous regions in terms of the number of components present. Local estimates can also be used to test for mixture homogeneity, which is an important pharmaceutical application; this application is a subject of future work.

CHAPTER III

Isoband - Reordering Variables for Banded Covariance Estimation

In this Chapter, we propose an approach that complements methods for permutation-invariant covariance estimation by taking a different view on covariance structure. Rather than restricting ourselves to methods that completely ignore potential structure in the order of the variables, we try to *discover* a structured ordering in the data and then use it to our advantage. The two types of structure we focus on are "approximately bandable" matrices, where you expect variables far apart in the ordering to be weakly correlated, and block-diagonal structures. The block-diagonal structure is a reasonable assumption for many types of data, including gene networks, where genes can often be clustered into strongly connected groups. Our main idea is to use the correlations between variables as a measure of similarity, and embed the variables in one dimension preserving the similarities as closely as possible. The coordinates of the variables in one dimension then provide an ordering.

In general, our method is most appropriate when there is structure, but it is non-trivial to describe the correct ordering. For instance, the example we consider in Section 3.5 has data on consumption of protein from various food sources in 25 European countries. It is clear that some foods are "closer" than others, e.g., one might think that beef and pork are more similar to each other than fish and fruit

in terms of protein consumption; but it is not obvious how such variables can be ordered. We show that in these situations, one will generally do better by discovering an ordered structure in the data than by ignoring it. However, note that our method is also invariant to permutations, in the sense that the order in which the variables are provided plays no role.

The rest of the chapter is organized as follows. In Section 3.1, we give some additional background on banding and thresholding a covariance matrix in high dimensions. In Section 3.2 we present the proposed methodology for discovering the ordering, which is based on the Isomap manifold projection method. Section 3.3 addresses selection of tuning parameters and gives extensive simulation results on discovering bandable and block-diagonal structures in the data. A discussion of comparison measures for comparing eigenvectors and using those measures to choose k for banding is given in Section 3.4. Section 3.5 presents an application to data on protein consumption from various sources in 25 European countries, and Section 3.6 gives some concluding remarks.

3.1 Background on banded and thresholded estimators

We focus on finding an ordering of the variables that will make the matrix as close to "bandable" as possible, and/or block-diagonal. To formalize what we mean by bandable, we provide a more in-depth review of the results on banded covariance estimators obtained by [5]. For this discussion, the observed p-dimensional random vectors $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ are distributed according to a distribution F, with $E\boldsymbol{X} = 0$ (without loss of generality), and $E(\boldsymbol{X}\boldsymbol{X}^T) = \Sigma$ which is estimated by the sample covariance matrix, $\hat{\Sigma}$. For simplicity, we assume that F is Gaussian, although [5] showed that Gaussianity can often be replaced by a tail condition on F.

Define the class of approximately bandable covariance matrices $\Sigma = [\sigma_{ij}]$ by

$$\mathcal{U}(\varepsilon, \alpha, C) = \big\{\Sigma : \max_j \sum_i \{|\sigma_{ij}| : |i-j| > k\} \le Ck^{-\alpha} \text{ for all } k > 0,$$

$$\text{and } 0 < \varepsilon \le \lambda_{\min}(\Sigma) \le \lambda_{\max}(\Sigma) \le 1/\varepsilon\big\}, \tag{3.1}$$

where $\lambda_{\min}(\Sigma)$ and $\lambda_{\max}(\Sigma)$ are the smallest and the largest eigenvalues of Σ, respectively. The first condition makes the matrix approximately bandable, and the second condition ensures it is well conditioned. This condition holds for all p, and Σ can be thought of as an infinite-dimensional matrix (or operator), with Σ_p given by the upper $p \times p$ submatrix of Σ. Define the (simple) banded estimator

$$B_k(\hat{\Sigma}) = [\hat{\sigma}_{ij}\, I(|i-j| \le k)],$$

where $I(\cdot)$ is the indicator function. Then it was shown that [5] uniformly on $\mathcal{U}(\varepsilon, \alpha, C)$, if $k_n \asymp (n^{-1} \log p)^{-\frac{1}{2(\alpha+1)}}$, then

$$\|B_k(\hat{\Sigma}_{p,n}) - \Sigma_p\| = O_P\left(\left(\frac{\log p}{n}\right)^{\frac{\alpha}{2(\alpha+1)}}\right) = \|B_k(\hat{\Sigma}_{p,n})^{-1} - \Sigma_p^{-1}\|\,. \tag{3.2}$$

Thus, the banded estimator and its inverse are consistent as long as $(\log p)/n \to 0$, as opposed to the sample covariance matrix which requires $p/n \to 0$ for consistency. Note the condition on the eigenvalues of Σ is a necessary condition for convergence of the banded estimator's inverse only, not the banded covariance estimator itself. Here $\|M\|^2 = \max_i |\lambda_i(M'M)|$ is the operator norm, a.k.a. the matrix l_2 norm or spectral norm. Convergence in operator norm guarantees convergence of all eigenvalues and eigenvectors [5, 29], and thus suggests that the estimator would be useful for improving PCA.

The result (3.2) makes it clear that if an ordering of the variables that has such a banded structure (or gets as close to it as possible) can be recovered, then covariance estimation can be greatly improved upon as compared to the sample covariance.

However, a comparison to a permutation invariant method of regularizing the sample covariance is also of interest – for example, does the structure matter if one is thresholding small entries to zero anyway? A partial answer is provided by the result of [4] on thresholding estimators of covariance. Recall the thresholded covariance estimate is given by

$$T_t(\hat{\Sigma}) = [\hat{\sigma}_{ij}\ I(|\hat{\sigma}_{ij}| \geq t)]\ . \tag{3.3}$$

By analogy to banding, a permutation-invariant analogue of the class of approximately bandable matrices can be defined as

$$\mathcal{U}_\tau\big(q, M, C(p)\big) = \{\Sigma : \sigma_{ii} \leq M,\ \sum_{j=1}^{p} |\sigma_{ij}|^q \leq C(p),\ \text{for all } i\}\ , \tag{3.4}$$

where $M > 0$ and $0 \leq q < 1$. On this class, it is known [4] that if the threshold $t_n \asymp (n^{-1} \log p)^{1/2}$, then

$$\|T_{t_n}(\hat{\Sigma}_{p,n}) - \Sigma_p\| = O_P\left(C(p)\left(\frac{\log p}{n}\right)^{\frac{1-q}{2}}\right)\ . \tag{3.5}$$

It is easy to check that $\mathcal{U} \subset \mathcal{U}_\tau$ with appropriately chosen constants. It is also easy to check that on the subclass $\mathcal{U}$, the rate of banding is better than the rate of thresholding although the difference is not sharp [4]. Thus, if there is an ordering of the variables that can make the matrix approximately bandable, the theory indicates we can expect to do better than thresholding if we discover that ordering.

3.2 Reordering variables with the Isomap

To focus ideas, we start with looking for an ordering of the variables that makes the matrix as close to bandable as possible within a single block of variables, and postpone the discussion of block-diagonal structures for later. The main idea is to treat discovering an ordering as a dimension reduction problem. We have p points

(variables) whose pairwise similarities are given by their covariances, or correlations. Using the correlations eliminates scaling issues between the variables. If we can embed these points in $\mathbb{R}^1$ in a way that maps their *dis*similarities into distances in $\mathbb{R}^1$, the coordinates of the variables in $\mathbb{R}^1$ will give us an ordering where closely correlated variables are placed near each other, and variables with weak correlations are placed far apart.

This embedding problem may be solved by multi-dimensional scaling (MDS), see, e.g., [7]. MDS starts with pairwise dissimilarities for a set of objects and constructs their embedding in $\mathbb{R}^d$ for a given d so that the dissimilarities match the Euclidean distances in $\mathbb{R}^d$ between the embedded points as closely as possible. Dimensions $d = 2$ or $d = 3$ are often used for visualization, and the dimension of the input data is typically much higher.

For our purposes, however, MDS turns out to be a poor choice. The metric MDS works best when applied to distances in Euclidean space; dissimilarities based on correlations are not necessarily Euclidean (a Euclidean distance may be formed based on correlations but the measure we use is not Euclidean). Apart from that, in a sparse matrix many empirical correlations will be close to zero, and metric MDS turns out to be unable to order those correctly (see more on this in Section 3.3). This problem does not occur when the true zero correlations are used as dissimilarities, and thus is caused by the noise in estimating covariance rather than by the covariance structure itself. Non-metric MDS, which only preserves the ranking of similarities rather than their values, has the same problem. Instead, we use a non-linear dimension reduction method designed for data on a manifold, the Isomap [55]. The Isomap is one of many manifold projection methods that became popular in machine learning several years ago (see also [48], [2], [13], [36] and many others), although perhaps found fewer

applications than was initially hoped for. The Isomap algorithm seems particularly suited for our problem because it explicitly aims to preserve distances between variables, but does not assume they are Euclidean. However, it is possible that similar results could have been obtained with another manifold projection algorithm.

Next, we describe the Isomap algorithm. It takes a pairwise dissimilarity measure $d(i, j)$ as input, and requires setting an integer tuning parameter r.

The Isomap algorithm

1. For each point, find its r nearest neighbors using the dissimilarities $d(i, j)$ (the dissimilarity between point i and point j). Construct a neighborhood graph by connecting each point to its r nearest neighbors (NN), with dissimilarities as the edge weights.
2. Estimate the geodesic distance $\tilde{d}_r(i, j)$ between each pair of points i, j by computing the shortest-path distance from i to j through the neighborhood graph.
3. Apply metric MDS to the matrix of pairwise shortest-path distances to obtain an embedding in $\mathbb{R}^d$. In our case, since we need a one-dimensional solution to get an ordering, this means find $z_1, \ldots, z_p \in \mathbb{R}^1$ that minimize the stress function (known as stress 1 in the literature)
$$S(z_1, \ldots, z_p) = \frac{\sum_i \sum_j (|z_i - z_j| - \tilde{d}_r(i, j))^2}{\sum_i \sum_j |z_i - z_j|^2} \ .$$
This minimization can be reduced to an eigenvalue problem.

Then we simply read off the ordering of the variables by ordering their projections $z_1, \ldots, z_p$ on the line. Ordering by descending or ascending order of the coordinates makes no difference. From this ordering, we construct a $p \times p$ variable permutation matrix $\hat{P}$. The covariance matrix is then reordered by

$$\hat{\Sigma}_o = \hat{P}\hat{\Sigma}\hat{P}^T, \tag{3.6}$$

the banding operator B_k is applied to the new matrix $\hat{\Sigma}_o$, and the variables are then reordered back to obtain the final estimator

$$\hat{\Sigma}_I = \hat{P}^T B_k(\hat{\Sigma}_o)\hat{P} \ . \tag{3.7}$$

We will refer to this estimator as Isoband, for Isomap+banding.

There are many ways to define a dissimilarity measure based on the covariance matrix. We use

$$d(i,j) = 1 - |\hat{\rho}_{ij}|$$

as a measure of dissimilarity, where $\hat{\rho}_{ij}$ is the sample correlation coefficient between variables i and j. Other monotone decreasing functions of $|\hat{\rho}_{ij}|$ have been tested and shown to behave very similarly. Alternatively, one could use $d(i,j) = C - |\hat{\sigma}_{ij}|$ (dissimilarities need to be non-negative). In either case, we do not distinguish between positive and negative correlations. However, the measure can easily be adjusted to accommodate other desired features of the ordering: for example, using $d(i,j) = 1 - \hat{\rho}_{ij}$ would result in strongly negatively correlated variables placed as far apart as possible, and positively correlated variables closer together. This case is related to the correlation clustering problem in computer science [1, 11], which aims to partition a weighted graph with positive and negative edge weights so that negative edges are broken up and positive edges are kept together. However, the correlation clustering algorithm does not look for the ordering that we need for banding, as we do not wish to remove strong negative correlations, and has also been shown to be NP-hard.

3.2.1 The case of disconnected neighborhood graphs

So far, we have assumed that the neighborhood graph constructed by the Isomap is connected, and thus shortest-path distances can be computed between all pairs of

variables. This is not guaranteed to be the case. If the graph consists of two or more connected components, then it seems reasonable to infer that the variables can be separated into independent blocks (this amounts to connected component clustering on the correlations), and set all between-block correlations to zero. Then Isomap can be applied to each component separately to make each block as bandable as possible, followed by banding each reordered block. The resulting estimator is both block-diagonal and banded, but we select the bandwidth separately for each block; we will still denote it $\hat{\Sigma}_I$ and refer to it as Isoband.

Note that we do not explicitly seek to construct a block-diagonal estimator, and only impose a block-diagonal structure if more than one connected component is found. An alternative would be to select a fixed number of blocks B, apply a clustering method to correlations to obtain B clusters, and construct a block-diagonal estimator with a block per cluster. We do not pursue this approach here, and note that in our case, the number of blocks B is determined from the data rather than supplied by the user.

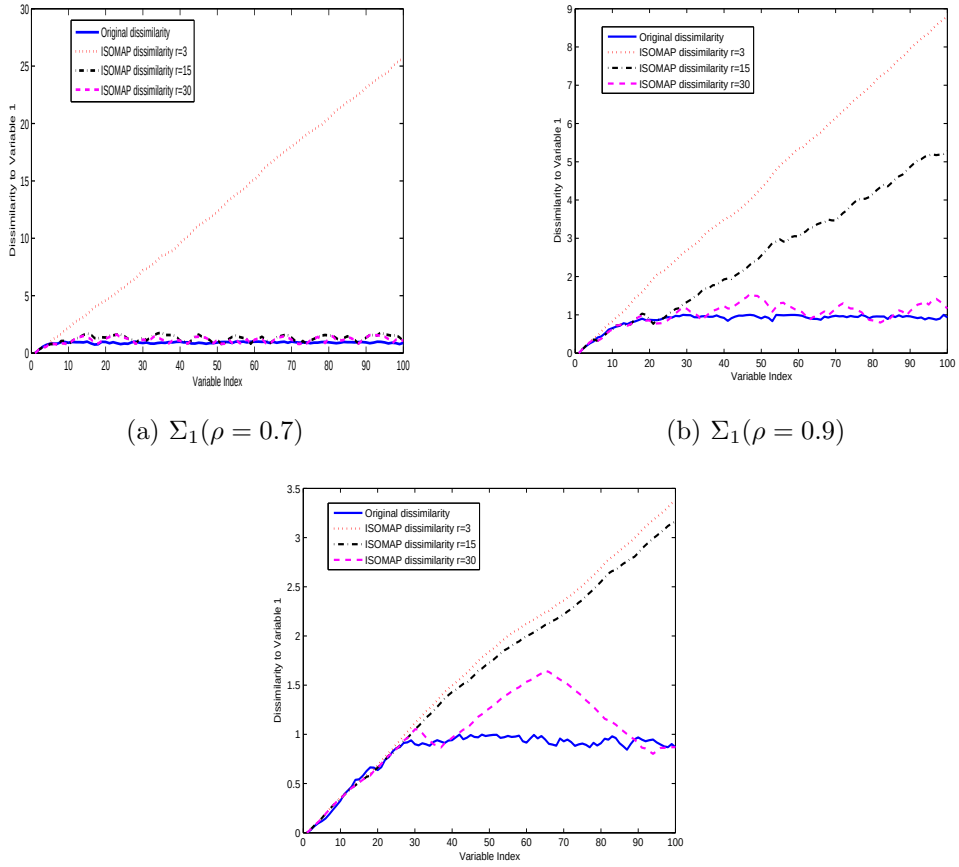
3.3 Simulation results

In this section, we investigate Isoband's performance by simulations, and address algorithmic issues such as tuning parameter selection. For simulations, the natural test is to take a covariance matrix that is approximately bandable and see whether (a) the Isomap can recover the correct ordering of the variables and (b) the Isoband estimator is closer to the true covariance matrix than its competitors. Throughout, we consider two types of bandable covariance structures:

$$\Sigma_1(\rho) \,:\, \sigma_{ij} = \rho^{|i-j|}\,, \qquad \Sigma_2(m) \,:\, \sigma_{ij} = \left(1 - \frac{|i-j|}{m+1}\right)_+ . \tag{3.8}$$

Σ_1 corresponds to an AR(1) process and its entries decay exponentially as one moves away from the diagonal; the entries of Σ_2 decay linearly and are set to zero outside the first m sub-diagonals. Both Σ_1 and Σ_2 are in the class $\mathcal{U}$, but Σ_1 is only approximately banded, whereas Σ_2 is m-diagonal (banded). Later, we will also concatenate these structures together in independent blocks to test Isomap's ability to discover independent blocks of variables.

3.3.1 Selecting the number of nearest neighbors for the Isomap



(a) $\Sigma_1(\rho = 0.7)$

(b) $\Sigma_1(\rho = 0.9)$

(c) $\Sigma_2(m = 25)$

Figure 3.1: Raw dissimilarities $d(1, j)$ plotted against variable index j, and Isomap shortest-path distances for several values of r (the number of nearest neighbors); $p = n = 100$.

Figure 3.1 shows the original dissimilarities for the first variable $d(1, j) = 1 - |\hat{\rho}_{1j}|$ plotted against variable index j, along with shortest-path distances $\tilde{d}_r(1, j)$ for the

number of nearest neighbors $r = 3$, 15, and 30. For all three cases, $p = n = 100$. It is clear that for MDS to recover the correct ordering the distances need to be increasing, and the more separated they are, the more likely the embedding is to be correct. Figure 3.1 explains why MDS fails to recover the ordering – all the distant variables are interchangeable; the Isomap, on the other hand, successfully builds up the distances through the neighborhood graph. The "humps" in the Isomap distances for larger r are due to "short-circuit" edges between points that are not true neighbors. At what value of r these short circuits begin to occur depends on how sparse the true matrix is: the sparser the matrix, the smaller r needs to be to avoid them. On the other hand, there does not appear to be a disadvantage in using smaller r for less sparse matrices; we fix $r = 3$ from this point on.

An alternative is to cross-validate for r following, for example, the cross-validation scheme proposed for selecting the bandwidth by [5]. A general result on the validity of this method was obtained in [4]. However, the embedding does not appear sensitive to the value of r as long as it is not too large, and we already have to apply cross-validation to select the bandwidth, so cross-validation for r would result in a grid search over k and r. This is in principle feasible, but to keep computational costs low, we fix the value of r instead.

3.3.2 Recovered orderings

Before comparing covariance estimators resulting from reordering the variables, we examine the orderings themselves. For model Σ_2, where the truth is banded, the order is recovered perfectly every time. For model Σ_1, which is only approximately banded, solutions are slightly more variable. To assess an ordering, we can plot the coordinates from the embedding against the variable index. The order will be recovered perfectly if the curve is monotone. Figure 3.2 shows the embedding coordinates

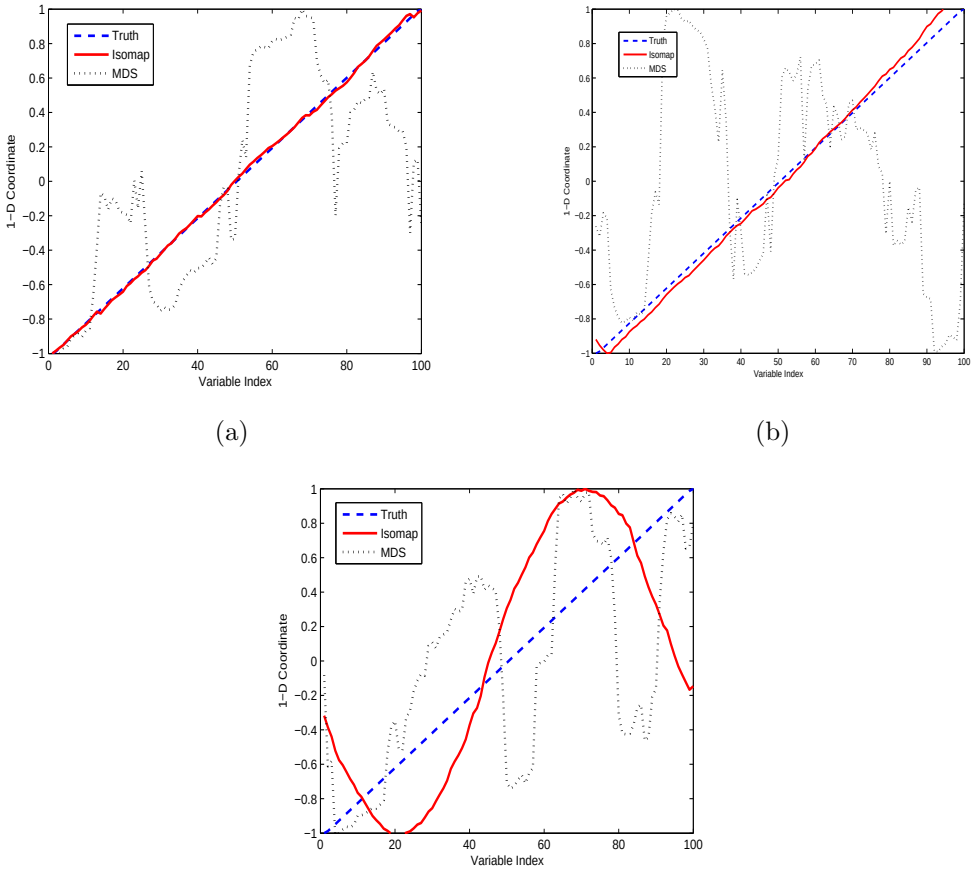


Figure 3.2: Coordinates in $\mathbb{R}^1$ plotted against variable index for the MDS and Isomap applied to the true $\Sigma_1(0.7)$ covariance and three realizations of the sample. Realization (a) is typical, (b) is less common, and (c) is rare.

from the true Σ_1 (which always results in a correct ordering), and coordinates from three different realizations of the sample covariance. The MDS orderings in all three realizations are completely wrong. For the Isomap, the correct ordering (a) is by far the most likely; the ordering shown in (b) is unusual, but it still does not significantly affect the performance of the estimator; the ordering in (c) does affect performance, but is rare.

3.3.3 Estimation results for approximately bandable matrices

Ultimately, the relevant measure of performance for our method is improved covariance estimation (rather than any measure related to the ordering itself). Here we compare the performance of the Isoband estimator $\hat{\Sigma}_I$, which is invariant to variable permutations, to three other permutation-invariant estimators: the sample covariance $\hat{\Sigma}$, the thresholded sample covariance $T_t(\hat{\Sigma})$, and the shrinkage estimator of Ledoit-Wolf [35], $\hat{\Sigma}_{LW}$. To demonstrate the advantage of the Isomap over MDS in recovering the ordering, we also include $\hat{\Sigma}_M$, a banded estimator obtained after reordering the variables according to their MDS projection onto 1-d. In addition, we include the banded estimator, which is not permutation invariant, applied to two different matrices: $B_k(\hat{\Sigma})$, where the variables are in their "correct" order as specified by the simulation models, and $B_k(P\hat{\Sigma}P^T)$, where P is a random permutation matrix. Of the banded estimators, we expect the best performance from $B_k(\hat{\Sigma})$ (since the correct order is given and the matrix is approximately bandable), and the worst from $B_k(P\hat{\Sigma}P^T)$. Note, however, that for a very sparse matrix, banding even in the wrong order may introduce enough zeros to improve on the sample covariance. The hope is that the performance of $\hat{\Sigma}_I$ will be close to $B_k(\hat{\Sigma})$.

For any estimator $\tilde{\Sigma}$, we measure the estimation performance by the operator (matrix l_2) norm of the difference between the estimator and the truth,

$$L(\tilde{\Sigma}, \Sigma) = \|\tilde{\Sigma} - \Sigma\|_2 = \max_i |\lambda_i(\tilde{\Sigma} - \Sigma)| \ ,$$

where $\lambda_i(A)$ denotes the i-th eigenvalue of A. Note that for the permuted estimator $B_k(P\hat{\Sigma}P^T)$, the correct loss is $L(B_k(P\hat{\Sigma}P^T), P\Sigma P^T)$, rather than $L(B_k(P\hat{\Sigma}P^T), \Sigma)$. We chose the operator norm because, as discussed in Section 3.1, convergence in operator norm implies good performance in PCA. However, in all simulations the matrix

l_1 norm and the Frobenius norm were also computed as alternative loss functions; all results are consistent across these three norms.

For each simulation, $n = 100$ observations were drawn from a multivariate normal distribution with mean zero and covariance matrix Σ, where Σ was from the class Σ_1 or Σ_2. For model Σ_1, we selected $\rho = 0.5$, 0.7, and 0.9; for Σ_2, $m = 0.1p$, $0.25p$, and $0.5p$. For all of these, simulations were conducted for $p = 10, 100$, and 200, with 50 replications for each setting. The random permutation P was fixed throughout the replications; the results are very similar if we average over many random permutations instead. For all banding estimators and for thresholding, the tuning parameter was chosen following the cross-validation scheme of [5] with 10 random splits of the data. In this set of simulations, the neighborhood graph with $r = 3$ was always connected.

Table 3.1: Average (SE) operator norm loss over 50 replications, for covariance models Σ_1 and Σ_2.

	p	Sample	Banding	Band.Perm.	MDS	Isoband	Thresh.	Ledoit-W.
ρ					Model Σ_1			
	10	0.75(.02)	0.68(.02)	1.05(.06)	0.78(.03)	0.74(.02)	0.87(.04)	0.69(.02)
.5	100	3.49(.04)	0.97(.02)	2.03(.00)	1.95 (.01)	1.96(.01)	1.74(.01)	1.80(.01)
	200	5.61(.04)	1.01(.02)	2.05(.00)	2.05(.00)	2.04(.00)	1.77(.01)	1.98(.01)
	10	0.80(.03)	0.86(.03)	0.80(.03)	0.90(.03)	0.86(.03)	0.87(.04)	0.82(.03)
.7	100	4.07(.08)	1.73(.03)	4.59(.02)	3.38(.05)	1.89(.04)	2.89(.03)	3.12(.03)
	200	6.68(.08)	1.77(.04)	4.67(.00)	3.93(.03)	2.04(.06)	3.00(.02)	3.81(.02)
	10	1.11(.08)	1.12(.08)	1.12(.08)	1.12(.08)	1.12(.08)	1.12(.08)	1.10(.08)
.9	100	6.10(.18)	4.81(.11)	9.54(.31)	6.45(.20)	4.99(.14)	7.02(.16)	5.84(.11)
	200	9.17(.16)	5.38(.11)	16.4(.23)	9.02(.17)	5.30(.11)	8.90(.11)	8.42(.12)
m					Model Σ_2			
	10	0.69(.02)	0.41(.02)	0.85(.01)	0.70(.02)	0.69(.03)	0.54(.03)	0.63(.02)
$.1p$	100	5.53(.13)	2.58(.07)	8.87(.14)	4.93(.10)	2.55(.07)	3.35(.08)	4.78(.07)
	200	10.5(.20)	5.06(.12)	17.7(.27)	10.3(.23)	4.99(.11)	6.55(.13)	9.26(.13)
	10	0.83(.03)	0.68(.03)	0.83(.03)	0.75(.04)	0.70(.03)	0.79(.03)	0.82(.03)
$.25p$	100	7.81(.32)	5.60(.27)	8.14(.37)	7.09(.32)	5.56(.25)	6.79(.30)	7.51(.29)
	200	14.8(.47)	10.1(.36)	16.1(.59)	13.8(.55)	10.1(.36)	12.6(.46)	14.4(.44)
	10	1.08(.06)	1.01(.06)	1.10(.06)	1.01(.06)	1.01(.06)	1.08(.06)	1.07(.05)
$.5p$	100	8.64(.45)	7.42(.42)	8.59(.44)	7.66(.41)	7.48(.39)	8.44(.43)	8.38(.42)
	200	17.8(1.0)	15.8(.96)	17.8(1.0)	15.8(.89)	15.6(.96)	17.3(1.0)	17.2(.94)

Table 3.1 shows the average and standard error (SE) of operator norm loss over

50 replications for the seven estimators. For Σ_1, for $\rho = 0.7$ and 0.9 and $p = 100$ and 200, Isoband comes close to banding in the correct order and performs better than any other estimator, including the estimator generated by the MDS reordering, thresholding and Ledoit-Wolf. For $p = 10$, the large p regime does not apply, so we see that banding even in the correct order does not necessarily improve on the sample covariance (and neither do thresholding or Ledoit-Wolf). For $\rho = 0.5$, which has very few entries that are substantially different from zero (and thus order is not as important), all the estimators show similar improvement relative to the sample covariance.

For Σ_2, the pattern is similar. For $p = 100$ and 200 and for all values of m considered, Isoband comes very close to the benchmark of banding in the correct order, and outperforms everything else. This is also true even for $p = 10$ for the less sparse structures with 3 and 5 sub-diagonals. For $p = 10$ and $m = 1$, thresholding comes closer to the benchmark banding result than Isoband, but that is the only exception.

The tuning parameters chosen by the various banding estimators (k) and thresholding (t) are shown in Table 3.2. Consistently in all situations, the bandwidth picked for banding in the correct order is very close to the bandwidth picked for banding in the Isomap order, another indication that the Isomap is recovering the ordering well. For banding after the variables have been permuted, the results are variable: it either tends to discard almost everything and keep the matrix near-diagonal if the underlying model is very sparse, or it tries to keep many more diagonals than needed in the right ordering. Finally, the threshold selections, which cannot be directly compared to bandwidths, are included for completeness. They do confirm that, in general, the larger the p, the more entries one needs to discard for best estimation

Table 3.2: Tuning parameter selection: averages (SE) over 50 replications (bandwidth for banding and threshold for thresholding).

	p	Banding	Band.Perm.	MDS	Isoband	Thresh.
ρ				Model Σ_1		
.5	10	2.9(.09)	5.64(.46)	3.96(.16)	4.08(.18)	0.27(.02)
	100	3.36(.07)	1.28(.09)	3.12(.20)	2.86(.26)	0.48(.001)
	200	3.44(.08)	1.06(.03)	2.08(.20)	1.42(.11)	0.49(.001)
.7	10	5.68(.23)	10.0(.00)	5.40(.19)	5.68(.21)	0.12(.01)
	100	5.68(.13)	1.78(.25)	6.72(.36)	6.22(.19)	0.47(.003)
	200	6.24(.11)	1.16(.07)	6.00(.37)	7.40(.23)	0.48(.002)
.9	10	9.84(.08)	9.98(.02)	9.72(.08)	9.82(.07)	0.01(.003)
	100	16.0(.39)	36.8(2.5)	21.3(.89)	15.2(.37)	0.38(.01)
	200	16.3(.44)	9.82(1.6)	21.3(1.0)	16.6(.38)	0.45(.004)
m				Model Σ_2		
$.1p$	10	2.00(.00)	2.48(.18)	3.32(.15)	3.24(.15)	0.37(.01)
	100	9.20(.11)	8.64(1.2)	15.8(.83)	9.04(.11)	0.44(.01)
	200	16.7(.19)	16.1(2.3)	27.7(1.7)	16.8(.21)	0.44(.01)
$.25p$	10	3.94(.07)	8.62(.09)	4.44(.11)	4.00(.08)	0.26(.01)
	100	20.5(.31)	87.1(2.1)	29.0(1.3)	20.9(.29)	0.35(.01)
	200	39.7(.53)	173(4.9)	53.4(2.4)	40.4(.81)	0.36(.01)
$.5p$	10	5.70(.15)	8.92(.10)	5.62(.10)	5.70(.18)	0.14(.01)
	100	41.3(.89)	95.8(.36)	44.2(2.1)	42.1(.89)	0.18(.01)
	200	88.2(3.7)	193 (.65)	81.6(2.7)	85.2(2.8)	0.16(.01)

performance.

3.3.4 Estimation results for block-diagonal covariances

Here we test the ability of the Isomap to discover independent variable blocks corresponding to the connected components of the neighborhood graph. Three types of within-block structure were considered: Σ_1 (AR(1)), Σ_2 (triangular), and, additionally, Σ_3, a constant correlation structure with $\sigma_{ij} = \rho$ for all $i \neq j$, and $\sigma_{ii} = 1$. Here we only consider dimensions $p = 100$ and 200, and $n = 100$. The number of blocks was fixed at 3, with sizes 50, 30, and 20 for $p = 100$ and 100, 60, and 40 for $p = 200$. We only show results for concatenating blocks of the same type, although many other settings with different numbers of blocks, block sizes, and block structures were examined. We use notation $\Sigma_1(0.7, 0.8, 0.9)$ to refer to a model with three AR(1) blocks with values of ρ of 0.7, 0.8, and 0.9, and size of the blocks as described

above. Similarly, $\Sigma_2(1/2, 1/2, 1/2)$ has three triangular blocks with m determined as a fraction of the corresponding block size; e.g., for $p = 100$ the m values for this model would be 25, 15, and 10. Again, r was fixed at 3, number of replications at 50, and a single random permutation P was fixed; there was no significant difference in performance for different permutations. The performance is again measured with the operator norm loss.

We also add another estimator, $\hat{\Sigma}_{BD}$, which is block-diagonal and assumes known blocks, but does not apply banding. The benchmark banding estimator, $B_k(\hat{\Sigma}_{BD})$, in a slight abuse of notation, represents banding each of the known blocks with variables given in their correct order, but the bandwidth k is selected separately for each block. Recall that for Isoband, the bandwidth for each found block is also selected separately. The estimator $B_k(\hat{\Sigma}_{BD})$ is expected to be the best, and others can be compared to it to see relative improvement. The MDS estimator is omitted from comparisons in this section, because it was shown to be inferior to the Isomap and it does not have the ability to generate a block-diagonal estimator.

Table 3.3: Average (SE) operator norm loss over 50 replications for block-diagonal covariance models. Block sizes are 50, 30, 20 for $p = 100$ and 100, 60, 40 for $p = 200$.

Setting	p	Sample	Banding	Band.Perm.	$\hat{\Sigma}_{BD}$	Isoband	Thresh.	Ledoit-W.
Σ_1	100	4.04(.07)	1.65(.04)	4.53(.01)	3.19(.09)	1.88(.05)	2.74(.03)	3.01(.03)
(.7,.7,.7)	200	6.51(.06)	1.80(.03)	4.64(.00)	5.22(.14)	2.21(.05)	2.98(.02)	3.79(.02)
Σ_1	100	4.59(.12)	2.41(.08)	9.41(.16)	2.96(.08)	2.59(.10)	3.61(.10)	4.42(.10)
(.7,.8,.9)	200	7.96(.17)	3.53(.14)	13.64(.08)	5.10(.15)	3.61(.12)	5.44(.13)	6.85(.13)
Σ_2	100	6.47(.19)	4.29(.20)	7.70(.26)	4.65(.21)	4.23(.21)	5.26(.20)	6.27(.16)
$(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$	200	12.50(.39)	7.75(.40)	16.65(.84)	8.73(.38)	7.75(.38)	9.91(.42)	12.32(.38)
Σ_2	100	4.99(.10)	2.47(.10)	9.55(.19)	3.16(.08)	2.44(.10)	3.13(.08)	4.54(.09)
$(\frac{1}{10}, \frac{1}{3}, \frac{3}{4})$	200	10.34(.28)	5.06(.28)	17.54(.44)	6.42(.23)	5.09(.29)	6.42(.25)	9.19(.16)
Σ_3	100	7.74(.35)	5.22(.36)	8.87(.44)	5.60(.39)	5.48(.33)	5.78(.33)	7.30(.27)
(.7,.5,.3)	200	14.59(.58)	10.52(.62)	19.58(.79)	10.59(.60)	10.89(.61)	11.72(.50)	14.67(.62)

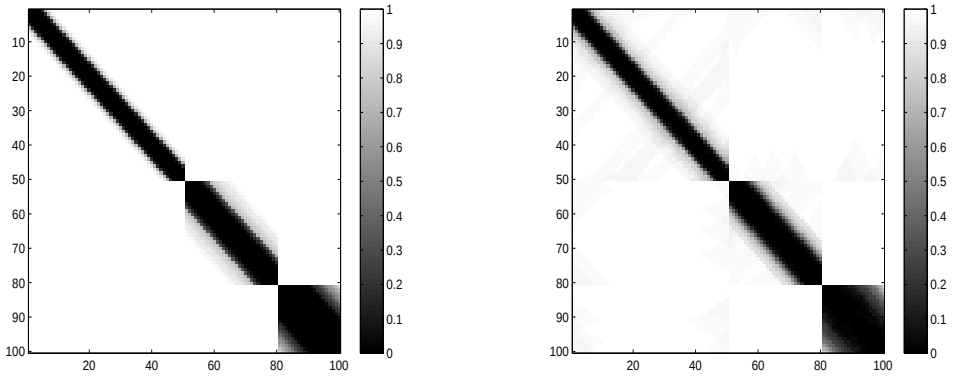
Block diagonal covariance results are shown in Table 3.3. For the first two models, Σ_1 and Σ_2, Isoband comes significantly closer to the benchmark $B_k(\hat{\Sigma}_{BD})$ than

anything else, being just slightly worse for Σ_1 and essentially the same for Σ_2. Thresholding and the unbanded block-diagonal estimator $\hat{\Sigma}_{BD}$ perform noticeably worse, and Ledoit-Wolf performs poorly, only slightly better than the sample covariance. Banding in the wrong order does very poorly, as it should (though still better than the sample in one case).

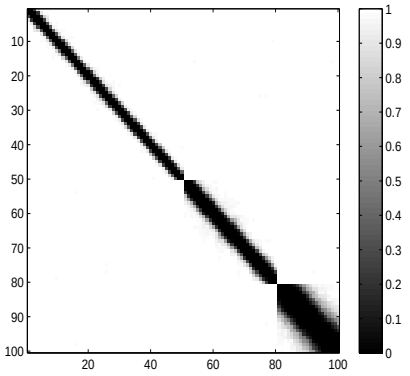
For the last model, Σ_3, the blocks are not banded, and there is little difference between the block-diagonal estimator, the block-diagonal estimator banded, and Isoband; thresholding is slightly worse than these three. The Ledoit-Wolf estimator is very similar to the sample, and banding in the wrong order is worse than the sample. This example serves to reassure us that when there is no structure, banding after reordering by the Isomap will do no harm, as the banding procedure can choose a large enough k to retain most of the diagonals: in this case, we discovered the blocks, but did not do any worse by trying to reorder the variables within the blocks.

To assess the sparse structures recovered by each estimator, we plot heatmaps of percentage of the time each element of the matrix was estimated as zero, for the benchmark banding of the correct blocks, Isoband, and thresholding. Banding in permuted order tends to pick a very large k and produces very few zeros, and the Ledoit-Wolf estimator is not sparse; they are omitted from the comparison.

Figure 3.3 shows the sparse structure of the estimators for the AR(1) block model $\Sigma_1(0.7, 0.8, 0.9)$. Note that, strictly speaking, the truth is not sparse, but it has many very small elements, which should be set to zero. In this case, Isomap does make a few mistakes in identifying the blocks (light-gray patches outside the blocks) but does very similar to benchmark banding within the blocks. Thresholding seems to be overall sparser – it has very few non-zeros outside the blocks and substantially



(a) Benchmark $B_k(\hat{\Sigma}_{BD})$ (b) Isomap $\hat{\Sigma}_I$

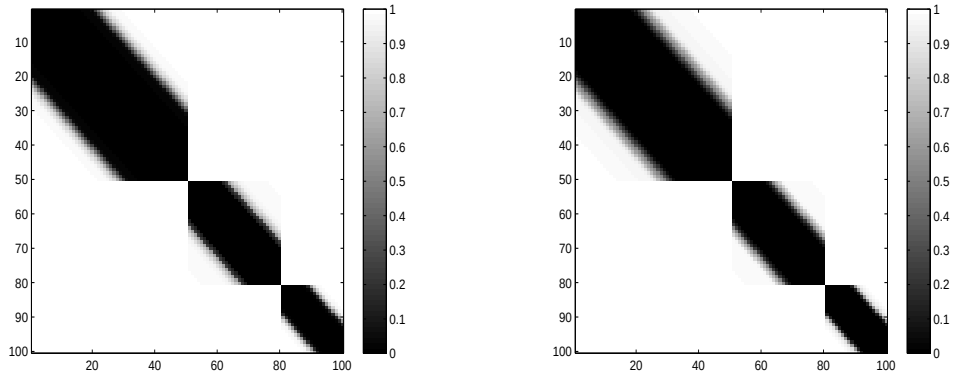


(c) Thresholding

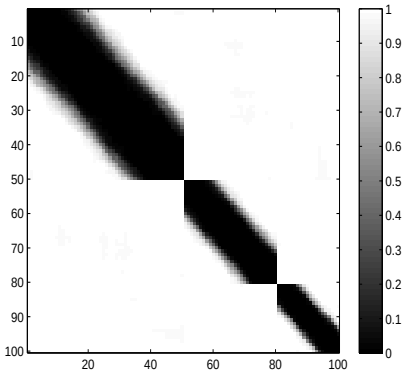
Figure 3.3: Heatmap of percentage of times out of 50 replications each element was estimated as zero for model $\Sigma_1(0.7, 0.8, 0.9)$. Black corresponds to 0%, white to 100%.

fewer non-zeros inside – but we know from Table 3.3 that it does not do as well as banding on estimation, and thus must be cutting off too many elements.

Results for the triangular block model $\Sigma_2(1/2, 1/2, 1/2)$ are shown in Figure 3.4. Here, the Isoband results are only slightly noisier than banding, and it identifies the blocks perfectly every time. Thresholding makes some errors outside the blocks, and again appears to cut off a little bit more than necessary inside the blocks. Overall, the Isomap does very well on picking up the two structural features we built into the simulated data – the blocks and the banded structure within each block.



(a) Benchmark $B_k(\hat{\Sigma}_{BD})$ (b) Isomap $\hat{\Sigma}_I$



(c) Thresholding

Figure 3.4: Heatmap of percentage of times out of 50 replications each element was estimated as zero for model $\Sigma_2(1/2, 1/2, 1/2)$. Black corresponds to 0%, white to 100%.

3.4 The Krzanowski measure for comparing eigenspaces

Before demonstrating the performance of Isoband on real data, a discussion of comparison measures for eigenvectors and/or eigenspaces is appropriate. We have been using the matrix l_2 norm for choosing k and comparing matrices, but may not always be interested in improving estimation of the entire covariance matrix. Consider the case where we are only interested in the first few eigenvectors rather than the entire covariance matrix, as is the case in PCA. We introduce some comparison measures and investigate their ability to help select tuning parameters for procedures such as banding for that setting.

3.4.1 Comparison measures

In order to determine what measures are appropriate to examine improvement in the leading eigenvector estimation and to perhaps also choose the value k for banding, we consider how values of k have been chosen in the literature when the goal was to improve estimation of the sample covariance matrix [5]. We have mentioned this cross-validation scheme previously in the Introduction, but here we discuss the details.

When the goal is to obtain the "best" estimate of the population covariance matrix and we are choosing k for simple banding, one may consider the risk function

$$R_1(k) = \|\Sigma - \hat{\Sigma}_k\| \tag{3.9}$$

where $\|\cdot\|$ denotes the matrix l_2 norm. The optimal value of $\hat{\Sigma}_k$ is the estimate that minimizes the risk function. However, we do not know the true covariance matrix, hence we must estimate the risk function, and examine whether or not it has minima at the same values of k that the population risk function does.

The cross-validation scheme of Bickel and Levina estimate the risk function as follows [5]. Consider repeatedly splitting the data into a training and test data set. For each split i, on the test data set, compute the sample covariance estimate, denoted $\hat{\Sigma}_i^{(1)}$, and on the training data set, compute all possible k banded sample covariance estimates, denoted $\hat{\Sigma}_{i,k}^{(2)}$. Then estimate the risk as

$$\hat{R}_1(k) = \frac{1}{N}\sum_{i=1}^{N} \|\hat{\Sigma}_i^{(1)} - \hat{\Sigma}_{i,k}^{(2)}\|, \tag{3.10}$$

where N is the number of splits, and choose k so that $\hat{R}_1(k)$ is minimized. In this approach, the test set is used to get an estimate of the truth, and the training data set is used to examine what possible regularized covariance matrices might look like.

This risk function has been found to behave similarly to the population version via simulations [5], and a theoretical justification has been provided in [4].

With the goal of comparing eigenvectors and improving estimation of the eigenvectors in mind, we can consider other measures besides the l_2 norm as the risk function, while still utilizing the general procedure outlined above with repeated data splits to choose k. One possible measure of comparison of eigenvectors is the cosine of the angle between them, analyzed theoretically by Johnstone and Lu [28]. Then k can be chosen to maximize the cosine of the angle between the vectors, or, in terms of minimizing a risk function, for just the first pair of eigenvectors choose k to minimize:

$$R_2(k) = 1 - e_1(\Sigma)' e_1(\hat{\Sigma}_k) \tag{3.11}$$

where $e_1(M)$ denotes the first eigenvector of a matrix M. Using the cross-validation scheme (let i count the number of splits), we can choose k to minimize the estimated risk function:

$$\hat{R}_2(k) = \frac{1}{N} \sum_{i=1}^{N} \left(1 - e_1(\hat{\Sigma}_i^{(1)})' e_1(\hat{\Sigma}_{i,k}^{(2)})\right), \tag{3.12}$$

This idea can be extended to deal with comparisons of multiple eigenvectors. However, we note that even when dealing with one eigenvector, sometimes the first sample and first population eigenvectors do not "match", because the order of estimated eigenvectors may not match the order of population eigenvectors, even if the principal eigenspaces are similar. Resolving these difficulties is possible with a matching algorithm; however, it adds computational complexity, and it is easier to use a measure of comparison that avoids this matching difficulty.

Measures of similarity have been derived for comparing principal components across several groups (or populations). Krzanowski [34] developed a measure of

similarity that can be used to compare the first m principal components from two or more groups. His measure also involves the cosines of the angles between the eigenvectors but does not suffer from a matching problem because all possible pairs are considered up to a point.

Following Krzanowski's setup, consider two covariance matrices Σ_1 and Σ_2. To compare the principal eigenspaces spanned by the first m PCs of each covariance matrix, let the lead eigenvectors for Σ_1 be denoted by $\{e_1, \ldots, e_m\}$, and the lead eigenvectors for Σ_2 by $\{f_1, \ldots, f_m\}$. Define a measure of similarity by

$$\mathrm{K}(m) = \sum_{i=1}^{m} \sum_{j=1}^{m} < e_i, f_j >^2 \ , \tag{3.13}$$

where the inner product $< e_i, f_j >$ gives the cosine of the angle between e_i and f_j. We call this measure of similarity the Krzanowski measure. If the two eigenspaces are exactly the same, $K(m)$ will take on its maximum value of m. Note that $K(p) = p$ always, and by definition, $K(0) \equiv 0$. Note that this measure examines all possible eigenvector pairings for the first m eigenvectors, and hence, this measure does not suffer from matching problems so long as m is chosen to be large enough.

We can use this measure in a risk function for choosing k for regularization. Let θ_{ij} represent the angle between the i^{th} population eigenvector and j^{th} sample eigenvector. Then a risk function can be defined as

$$R_3(k) = m - \sum_{i=1}^{m} \sum_{j=1}^{m} \cos^2(\theta_{ij}) = m - K(m), \tag{3.14}$$

where $K(m)$ is the Krzanowski similarity measure between Σ and $\hat{\Sigma}_k$. If we again divide the data into training and test data sets, where we compute all possible banded estimates on the training data set, and let $\hat{\theta}_{ij_k}$ represent the angle between the i^{th} eigenvector from the test data set and j^{th} eigenvector from the training data set for

the k-banded estimator, then we can approximate the risk function $R_3(k)$ by

$$\hat{R}_3(k) = \frac{1}{N}\sum_{i=1}^{N}\left(m - \sum_{i=1}^{m}\sum_{j=1}^{m}\cos^2(\hat{\theta}_{ij_k})\right) , \tag{3.15}$$

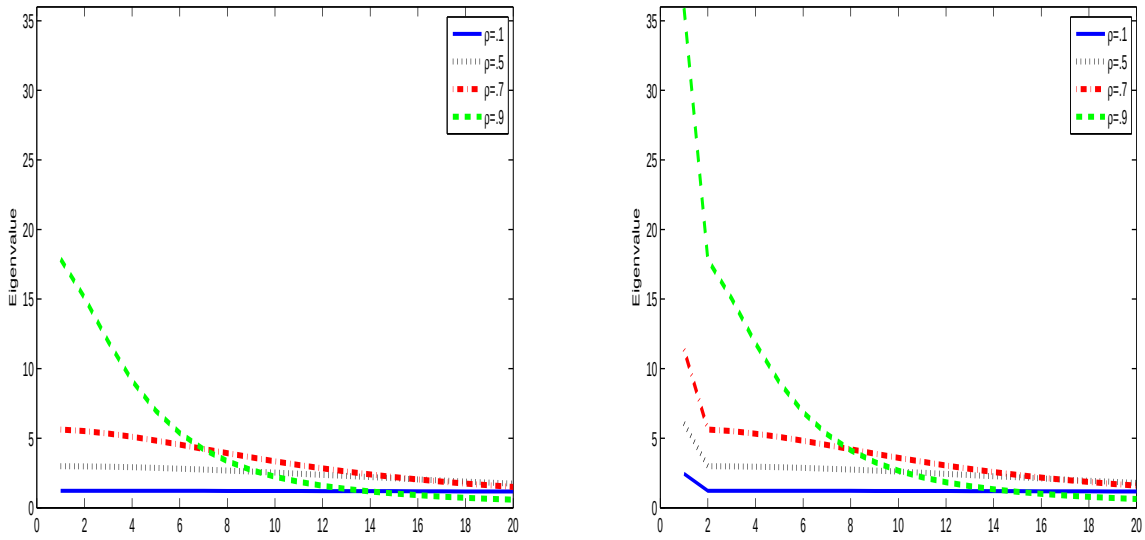
where N is the number of splits.

3.4.2 Model Settings

In order to examine the Krzanowski measure as a possible tuning measure for selecting the bandwidth k, we need to examine a model similar to the "spiked" covariance model, where only the first few eigenvectors are of interest. This model assumes that most of the population eigenvalues are 1, while a few are greater than 1 and well-separated from the rest. In the original spiked model by Paul [45], $\Sigma = \text{diag}(\lambda_1, \lambda_2, ..., \lambda_M, 1, ..., 1)$, with M eigenvalues greater than 1. However, in this model $k = 0$ is clearly the best choice for any measure to make since the true covariance is diagonal. Instead, we extend the idea of a "spiked" model to an auto-regressive (AR) setting.

The AR covariance structure Σ_1 defined in (3.8) has eigenvalues that are not well separated for most ρ. Let λ_1 be the largest eigenvalue of the unspiked AR(1) covariance matrix. In order to introduce a single spike, we set the variance of the first variable to be $q * \lambda_1$, where q is a multiplier greater than 1. We examined a range of values of q though the results presented here focus on $q = 2$. To introduce more than one spike, let the number of spikes be s, and the value of the multiplier for the smallest spike be q. Then, set the variances of the first s variables to be $q * \lambda_1$,$(q+1) * \lambda_1$,...,$(q+s-1) * \lambda_1$, respectively. With this method of spiking, the AR correlation decay structure is preserved, and the eigenvalues corresponding to each spike are well separated. Then, we sample observations from this model under the Gaussian assumption. Scree plots for the unspiked and spiked AR model with

$\rho = .1,.5,.7$, and .9 are shown in Figure 3.5; only the first 20 eigenvalues are shown for $p = 100$. The figure shows how the spiking procedure results in well-separated eigenvalues in the case of a single spike.



(a) Unspiked (b) Spiked

Figure 3.5: Scree plots of unspiked and spiked AR eigenvalues for first 20 eigenvectors for $\rho = .1,.5,.7$, and .9 for one spike

For all of the simulations, the number of observations in each sample was $n = 100$, the number of variables $p = 10$, 50, 100, and 200, and for the AR structure, $\rho = .1$, .5, .7, and .9. For each simulation, 100 data sets were drawn from the true model. When the cross-validation scheme was being used to choose k, simulations for $p = 10$ were run with 50 splits on each data set, and with 10 splits for $p > 10$. For the Krzanowski measure, the number of principal components compared (m) ranged from 1 to 20 taking values in (1,2,3,4,5,10,15,20). Other simulation details are discussed with the results.

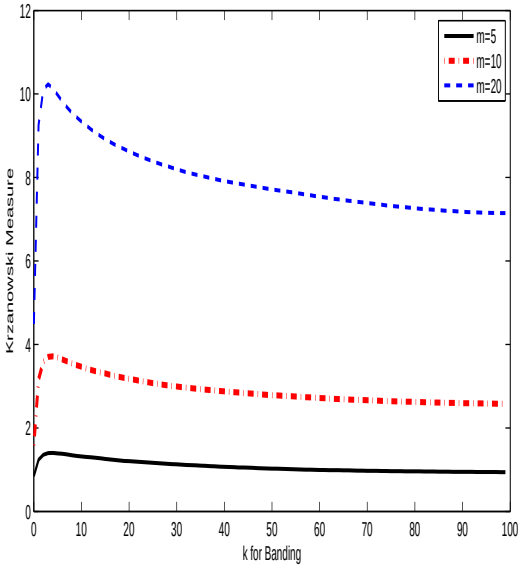
3.4.3 Results

We summarize our results for choosing k using the various risk functions, and examine the effects of the regularized estimators on the resulting eigenvectors and eigenvalues. Selected results are presented that are representative of all results.

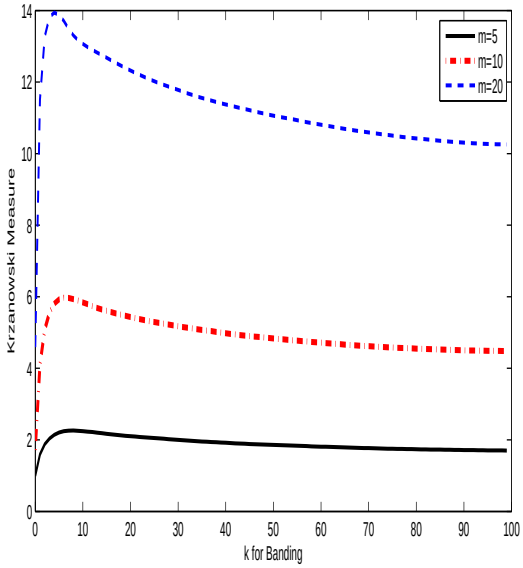
First, we investigated whether the three proposed risk functions based on matrix l_2 norm (Norm), cosine between first eigenvectors (Cos), and the Krzanowski measure ($K(m)$) result in different oracle values of k. That is, we computed the population version of the risk functions (3.9), (3.11), (3.14), and examined the optimal k chosen by each measure. The number of replications to compute the expectation for and number of observations n were both set at 100 for each combination of p and ρ. While the chosen values of k were similar and generally small, they were not exactly the same across the three measures.

In order to estimate the three risk functions from data, we implemented the cross-validation scheme described above and computed risk from (3.10), (3.12), (3.15), for 100 replications with $n = 100$ observations sampled for each combination of p and ρ. Figure 3.6 displays the average estimated value of the Krzanowski measure ($K(m)$ itself rather than the risk function (3.15)) versus the values of k for $p = 100$ and several values of ρ and m when there was one spike in the data. The locations of the maxima of the curve (which correspond to the optimal k) do change depending on ρ as expected due to the AR structure, but are fairly consistent across m, although the curve has local maxima at high m for high ρ. We fix $m = 5$ in the results that follow.

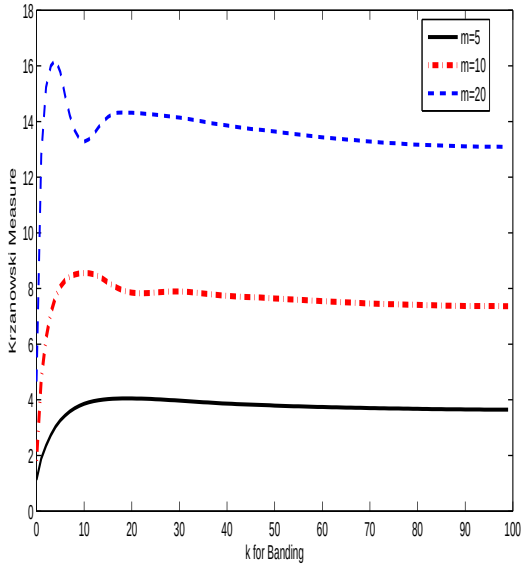
Figure 3.7 shows the true versus estimated average values of the Krzanowski measures for $p = 100$, $m = 5$, and values of $\rho = .5$ and .9. The estimated measure is less than the true measure in all cases (as expected), but the maxima occur at



(a) $\rho = .5$



(b) $\rho = .7$

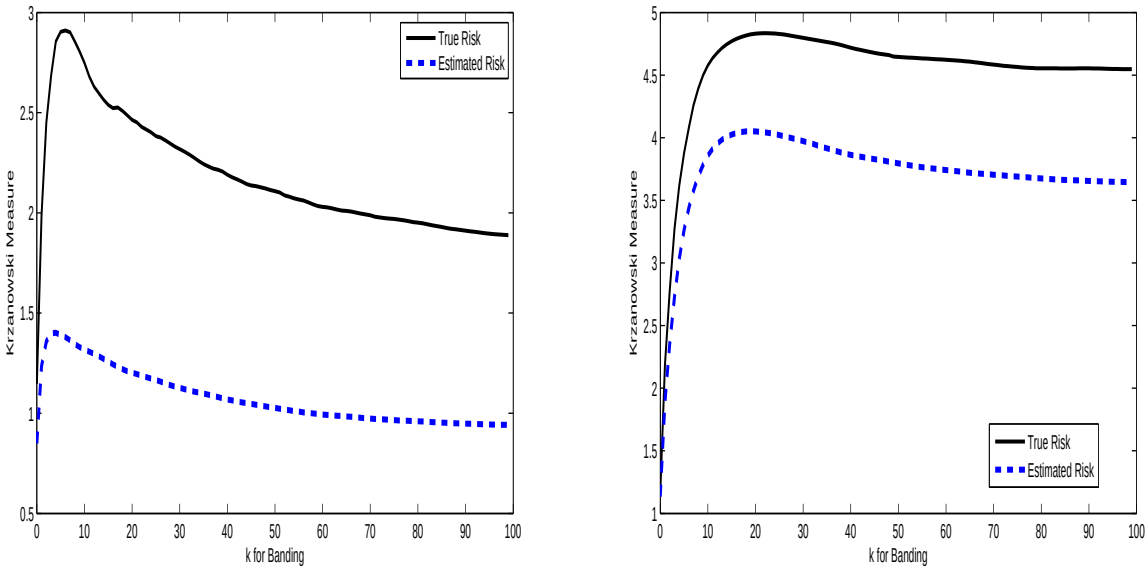


(c) $\rho = .9$

Figure 3.6: Average Krzanowski measure vs. k for AR Setting $n = p = 100$, for several values of m

roughly the same k values. This suggests the resampling scheme provides an adequate estimate of the risk based on the Krzanowski measure for the purposes of choosing k.

In order to determine which tuning measure is best for improving the estimation

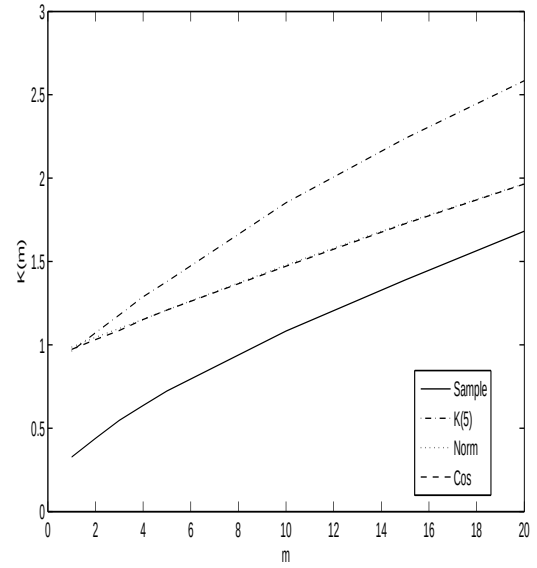


(a) $\rho = .5$ (b) $\rho = .9$

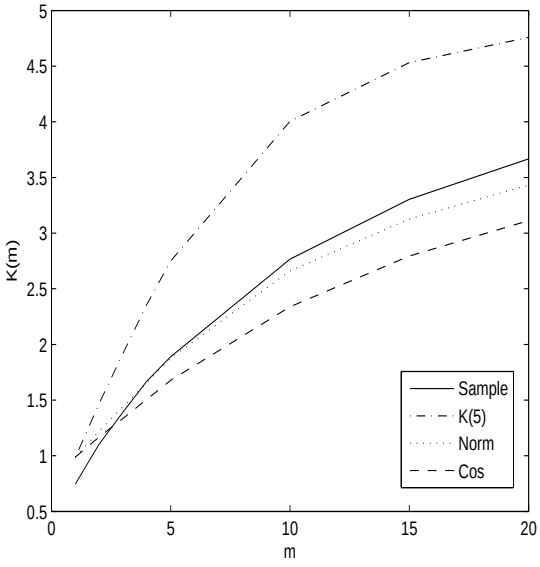
Figure 3.7: True vs. estimated average Krzanowski measures

of the leading eigenvectors, we computed the Krzanowski measure between the different banded estimators with k chosen using the estimated risk functions (3.10), (3.12), (3.15), and the true covariance. Results in Figure 3.8 show how each measure improved estimation of the first m eigenvectors, for $n = p = 100$. We used $m = 5$ in (3.15) although only one spike was introduced into the model. Figure 3.8 shows that the estimate based on the Krzanowski measure is the best at improving estimation of the leading eigenvectors. For completeness, we include a table of average l_2 norms over the 100 replications with each estimator compared to the true covariance. Table 3.4 confirms that all the banded estimates improve on the sample covariance in terms of the l_2 norm for this "spiked" AR model with one spike. Moreover, for all but the smallest value of ρ, the Krzanowski measure provides better tuning than the norm measure even when performance is measured by the norm loss.

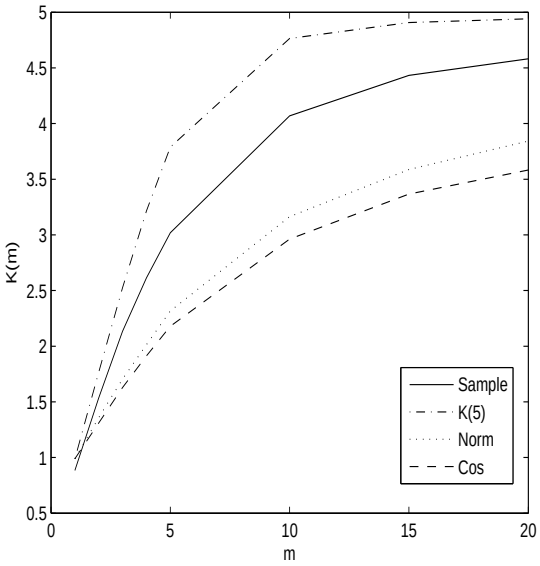
Future work will need to further examine the impact of m on the chosen values of k. In conclusion, the Krzanowski measure is a suitable measure for comparing



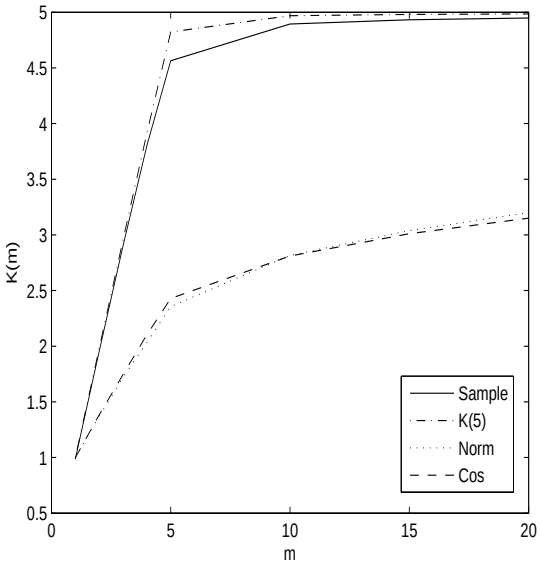
(a) $\rho = .1$



(b) $\rho = .5$



(c) $\rho = .7$



(d) $\rho = .9$

Figure 3.8: Average $K(m)$ vs. m for the banded estimates based on each tuning measure and the sample covariance compared to the true covariance over 100 replications with $n = p = 100$

principal components, and could have applications choosing tuning parameters in settings where a comparison of eigenvectors is of primary interest.

Table 3.4: Average l_2 norm loss over 100 replications for each tuning measure.

ρ	Sample	$K(5)$	Cos	Norm
.1	2.94	.60	.57	.53
.5	3.76	1.23	1.62	1.47
.7	4.79	2.10	3.40	3.33
.9	9.17	6.26	8.87	8.26

3.5 Protein consumption example

In this section, we illustrate advantages of discovering structure on a dataset that contains data on per capita protein consumption from nine different protein sources in 25 European countries, which was analyzed via principal component analysis in [16]. All variables are measured in grams per capita per day, so there is no need to standardize, and we perform PCA on the covariance rather than the correlation matrix. The nine protein sources are meat (grazing animals), pork and poultry, eggs, dairy, fish, cereals, starchy foods, pulses and nuts and oil-seeds, and fruits and vegetables. While the dimension of the data ($p = 9$) is moderate, the low sample size ($n = 25$) makes regularization necessary. This situation is exactly what our estimator is designed for: it is clear that some variables are "closer" than others (e.g., one might think that meat and pork are closer than dairy and fish), but it is not obvious a priori how the variables should be ordered.

To compare the estimators, we compute the principal components from the sample covariance matrix, Isoband, and the thresholded covariance matrix. Note that the Ledoit-Wolf's estimator does not change the eigenvectors, and its principal components are identical to those of the sample covariance. The Isomap reordered the variables as eggs, meat, starchy foods, pork and poultry, cereals, dairy, pulses and nuts and oil-seeds, fish, and fruits and vegetables. Banding in this order chose to keep only three sub-diagonals, introducing 30 zeros in the covariance matrix. The thresh-

olded covariance had 35 zeros; notably, covariances between fruits and vegetables and all other variables were set to 0 (which accounts for 16 of the zeros).

To compare the differences in principal eigenspaces, we use the Krzanowski measure [34]. Table 3.5 gives the values of the Krzanowski measure for $m = 1, .., 9$ for Isoband and thresholding, both compared to the sample covariance PCs. A gain of about 1 between $K(m-1)$ and $K(m)$ indicates that the m-th PCs are very similar; a gain close to zero indicates that they are different. Table 3.5 shows that there is a big difference in the 1st PC between the sample covariance and Isoband. Moreover, $K(m)$ does not come close to m until $m = p$, which means that the 1st Isoband PC is different from all the first eight PCs the sample covariance has found. For thresholding, on the other hand, the first four PCs are very similar to the sample, and the differences only appear in the 5th PC, which accounts for only about 2% of the total variation (Table 3.6).

Table 3.5: Krzanowski measure $K(m)$: principal eigenspaces of Isoband and thresholding compared to the sample covariance.

m	1	2	3	4	5	6	7	8	9
Isoband	.17	1.04	2.14	2.94	3.48	4.45	5.45	7.11	9
Thresholding	.999	1.91	2.90	3.88	4.06	4.81	6.29	7.32	9

Table 3.6: Percentage of variance explained by the nine PCs.

m	1	2	3	4	5	6	7	8	9
Sample	72.27	14.10	6.72	3.87	1.76	1.11	.73	.33	.11
Isoband	69.30	14.76	6.16	4.82	1.79	1.44	.94	.70	.08
Thresholding	69.76	15.36	6.24	4.05	2.34	1.48	.63	.15	.01

Examining the loadings in Table 3.7 shows that the first principal component for the sample covariance and thresholding is essentially the difference between cereals and dairy, whereas the first PC from Isoband is the difference between cereals and fish. The second PC involves more variables, but the sample and thresholding both

Table 3.7: Loadings for the first two PCs.

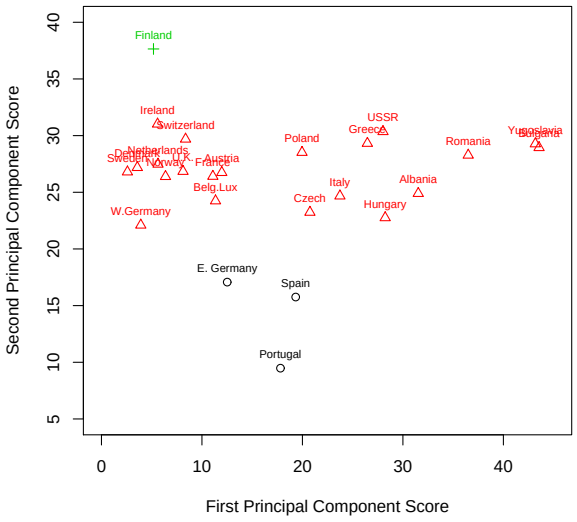
Estimator	Meat	Pork/Poul.	Eggs	Dairy	Fish	Cereal	Starch	Pulses	Fruits/Veg.
	First Principal Component								
Sample	-.15	-.13	-.067	**-.43**	-.13	**.86**	-.067	.11	.020
Thresholding	-.15	-.11	-.067	**-.42**	-.12	**.87**	-.057	.11	0
Isoband	-.01	-.11	-.065	-.13	**.87**	**-.42**	.11	-.13	.014
	Second Principal Component								
Sample	.14	-.05	.008	**.84**	-.27	**.40**	-.078	-.06	-.17
Thresholding	.16	-.28	.006	**.82**	-.30	**.34**	-.10	-.03	0
Isoband	.025	**.30**	.047	.051	-.34	**-.85**	.08	.19	.17

still put a large weight on dairy, and Isoband does not.

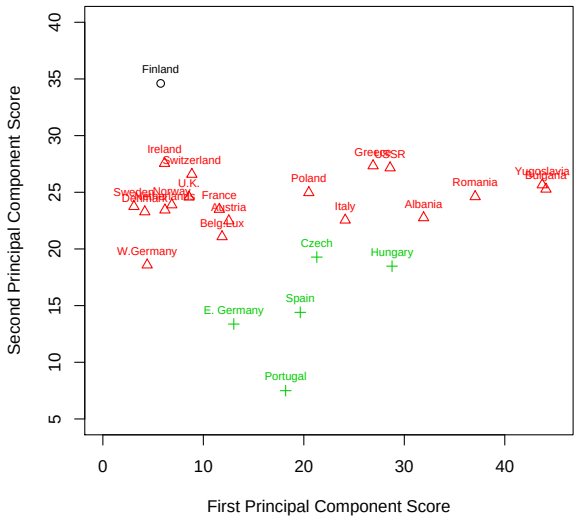
To further illustrate the differences between the principal components, we projected the data onto the first two principal components and applied agglomerative clustering (bottom-up) via the agnes algorithm [32] using Euclidean distances between projected data points as dissimilarities. Results for three clusters are shown in Figure 3.9. Agnes solutions are hierarchical; at the first split, it divides the data into two clusters, and then further splits one of the clusters into two.

Briefly, we make some notes about the principal component scores. The principal component scores for the sample covariance are uncorrelated, by definition of principal components analysis. The scores based on thresholding appear uncorrelated due to the similarity between the principal components from the sample covariance and thresholding. However, the principal component scores for Isoband are positively correlated. This can happen whenever the principal components are modified for better interpretability, in particular, when small loadings are thresholded to zero.

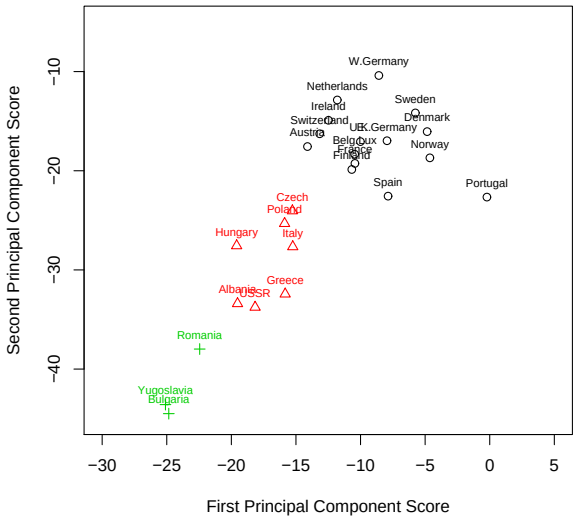
The clustering for the sample covariance matrix is very similar to thresholding. Both essentially cluster only on the second PC, and both separate out Finland (sample at the second split and thresholding at the first). The only difference is that Czechoslovakia and Hungary are split off from the biggest cluster in thresholding and combined with East Germany, Spain and Portugal. Geographically and cul-



(a) Sample



(b) Thresholding



(c) Isoband

Figure 3.9: Agnes clustering of the protein consumption data in the space of the first two PCs. The first split separates circles from everything else; the second split separates triangles from pluses.

turally, these clusters are difficult to explain. The Isoband clustering is completely different. All three clusters are substantial in size and both the first and second PCs have discriminating information. The biggest cluster is clearly Western Europe;

the second split separates South-Eastern Europe (Yugoslavia, Romania, and Bulgaria) from the rest. While it is possible that other clustering methods may have obtained somewhat different clustering results, this example serves as an illustration of a meaningful reordering of the variables resulting in more meaningful principal components.

3.6 Discussion

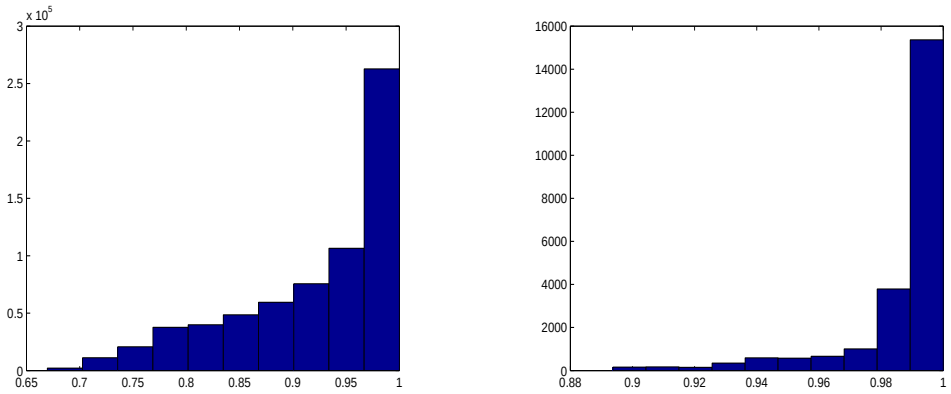
While in this chapter we have concentrated on regularizing the covariance matrix itself after recovering an ordering, there are many methods for regularizing the concentration matrix (the inverse of the covariance) that depend on variable ordering, for example, banding [60, 5] or adaptive banding [37] of the Cholesky factor of the inverse. Since these methods also rely on an ordering that places correlated variables close together, our methodology for finding an ordering is equally applicable to regularizing the concentration matrix. An alternative would be to use partial correlations as a measure of similarity, since they are directly the entries of the concentration matrix, but this is not feasible in high dimensions because partial correlations, unlike regular (marginal) correlations, cannot be estimated reliably.

Another extension of our method is to spatial data, where one might project onto two or three dimensions instead of one to find a variable structure that is most suitable for, e.g., modeling by a Markov random field. Since the theoretical results on banding generalize to any metric on the variable indexes, not just the distance on the line, one would gain the same advantages from regularization, but a 2- or 3-dimensional structure may be more meaningful in some applications.

Also, note that the computational cost of finding a one-dimensional ordering is simply the cost of computing the leading eigenvector of a $p \times p$ matrix. This can

be done efficiently and quickly for p on the order of several thousand, and the computation cost is negligible compared to permutation-invariant methods that require semi-definite programming algorithms.

Finally, we return to our Raman spectroscopy application and consider the effect of applying PCA to the Isoband estimate rather than the sample covariance to extract the pure component spectra. We examined a data set where the primary interest was extracting the spectra for normal and damaged bone. The most prominent difference in the pure component spectra is a peak shift for the phosphate component between 957 and 962 cm^{-1}, although shifts are present at other wavenumbers. On both the full data set ($p = 815$) and a subset of variables chosen to cover the main peak shifts ($p = 151$), Isoband found only one block and picked $k = p$, and thus the resulting extracted spectra are identical to those extracted via the sample covariance. The principal reason for this is that the covariance matrix in the Raman spectroscopy application is not sparse in the sense that Isoband is designed for; the measurements across different wavelengths are highly correlated (see Figure 3.10). Similarly, for thresholding the cross-validation chooses such a low threshold that no values are actually set to zero. For this data, the smallest correlation in the data set was .6696 and for the partial $p = 151$ subset, it was .8937. Alternative estimators for Raman spectroscopy will need to be investigated that do not rely on sparsity in the matrix but instead capitalize on the low intrinsic dimension of the data and the underlying linear structure of the problem.



(a) Full data set

(b) Partial data set

Figure 3.10: Histograms of correlations for full $p = 815$ and partial $p = 151$ Raman data sets.

CHAPTER IV

Improving Nearest Neighbor Graph Construction

4.1 Background on nearest neighbor graphs

Graph construction is a building block of many statistical techniques. In dimension reduction, the Isomap method relies on a NN graph to estimate geodesic distances between points to uncover non-linear structure in the data [55]. In machine learning (most notably semi-supervised learning), NN graphs can be used to label the unlabeled data points [63]. Graphs are also a fundamental part of studying networks, including social, computer, and biological networks [43].

Consider a graph where edges between the vertices (which may or may not have weights) reflect similarity of the vertices. One such graph is a fully connected graph, where every vertex has an edge to every other vertex with weights to reflect similarity between vertices. However, these graphs are not practical in some applications as they have a high computational cost. Alternatives to fully connected graphs are sparse graphs including k-NN graphs, ϵ-NN graphs, tanh-weighted graphs, and exp-weighted graphs. In k-NN graphs, each point is connected by an edge (directed or undirected) to its k nearest neighbors under some dissimilarity d. These graphs may not be connected, particularly with small k, which we capitalized on in our Isoband methodology to discover blocks. An alternative is to have edges between vertices x

and y if $d(x, y) \leq \epsilon$ for some ϵ, which is referred to as the ϵ-NN graph. The choice of ϵ, however, depends on the problem and can be challenging. The tanh-weighted graphs and exp-weighted graphs are constructed with weight functions that can set edge weights to zero based on the value of the dissimilarity between vertices, and hence, not add an edge between x and y if the dissimilarity is large. In semi-supervised learning, k-NN graphs have been found to perform well compared to these other graphs [63]. Similarly, k-NN graphs worked well for our Isoband methodology, so here we focus on improving construction of undirected k-NN graphs.

In general, there are no restrictions on the dissimilarity function d. However, for many methods including the Isomap, it is natural to have non-negative dissimilarities, $d(x, x) = 0$, and $d(x, y) = d(y, x)$, but the triangle inequality does not need to be satisfied, i.e., d does not need to be a metric. However, note that even with symmetric dissimilarities, $d(x, y) = d(y, x)$ does not imply that x is one of the k nearest neighbors of y if y is one of the k nearest neighbors of x.

Based on our experience with Isoband, we have seen that noise in the data can lead to unstable k-NN relationships. This occurred with block diagonal covariances, where the blocks with weaker internal correlations were merged incorrectly about 20 percent of the time. Additionally, particularly if the data are not sampled uniformly, the choice of constant k for all data points can force erroneous nearest neighbor connections into the graph. These short-circuits (erroneous nearest neighbors connections) and issues with instability in k-NN graphs call for investigating robust methods for constructing the graphs.

In this chapter, we explore modifications to k-NN graph construction with an aim to improve block detection in our Isoband method. We discuss existing graph perturbation methods in Section 4.2. Section 4.3 discusses two new proposals aimed

at improving graph stability, based on the bootstrap and on local smoothing. Simulation results on block diagonal covariance models demonstrating the improvement in Isoband are given in Section 4.4. Section 4.5 shows the results of applying the bootstrap and smoothing graph perturbation methods to the protein consumption data from Section 3.5, and Section 4.6 demonstrates results on a gene expression data set. We conclude with discussion and future work in Section 4.7.

4.2 Existing methods for graph perturbation

One area where graph perturbation has been used to improve robustness is the study of networks, where researchers aim to uncover community structure in large-scale networks but the number and size of communities in the network is not known a priori [43]. In this field, researchers have sought a measure for quantifying the strength of discovered community structure. It seems plausible to conclude that if the number of edges between groups is sufficiently small, or the number of edges within a group is sufficiently large, compared to what would be expected from a random graph, the community structure found is meaningful. This idea was quantified by the concept of *modularity*. Modularity is defined to be (up to a multiplicative constant) the number of edges falling within groups minus the expected number in a network where the expected degree of vertices matches the starting network but with edges placed at random. This random graph is called the standard configuration model [43]. However, researchers have found that some networks with high modularity do not have strong community structure, and hence, high modularity is only a necessary but not a sufficient condition for strong community structure [31].

Instead of maximizing modularity when looking for community structure, Karrer et al. [31] proposed to test community structure by studying the robustness of

the community assignments to perturbations of the graph. The procedure used to perturb the graph and test robustness is as follows:

1. For a given network with n vertices and m edges, compute the degree of each vertex k_i, $i = 1, ..., n$.
2. Compute the expected number of edges between vertex i and j under the so-called standard configuration model as

$$e_{ij} = \frac{k_i k_j}{2m}. \tag{4.1}$$

3. For each edge in the original network, with probability α, remove that edge and add a new edge between vertex i and j with probability e_{ij}/m.

Constructing a perturbed network in this fashion results in the same expected degree of vertices as in the starting network. Varying α allows different levels of perturbation of the network: $\alpha = 0$ implies that no edges are perturbed, while for $\alpha = 1$ the process will generate a fully random graph with the same expected degree of each vertex. Note that in the application in [31], edges are not weighted.

A modification of this perturbation method is necessary to apply it to a weighted graph based on correlations for use in Isoband. The original graph is constructed using k nearest neighbors defined by the correlation-based dissimilarity measure. Then its adjacency matrix is perturbed T times according to the algorithm we outlined above. The adjacency matrices of the T perturbed graphs are then combined into a final adjacency matrix, which has an edge between variables i and j if there is an edge between i and j in more than cT of the T graphs, where $0 < c < 1$ is a tuning parameter. The weight is set equal to the original weight if the edge is included. We refer to this method as the edge perturbation method since it works by directly perturbing the edges.

Another graph perturbation method was introduced by Carreira-Perpiñán and Zemel [9], in the context of perturbing minimal spanning trees (MST). Each data point x_i is perturbed by adding Gaussian noise with mean zero and standard deviation $s_i = \beta\gamma_i$ where γ_i is the average distance to the k nearest neighbors of x_i and $\beta \in [0, 1]$ is a tuning parameter. A MST is constructed from the perturbed data set (note that this tree is unweighted), and the process is repeated on T perturbed versions of the data ($T = 20$ was used in [9]). The final tree has a weight on each edge equal to the number of times that edge appears in the T perturbed trees [9]. One possible extension when starting with a weighted graph is to retain edges in the final graph if those edges appeared in more than cT of the perturbed graphs, and keep original weights. This method was designed to use with Euclidean distances between observations, but in our context adding something on the scale of the correlations directly to the variable vectors does not make sense. Instead, we propose a new method in the next section that makes use of neighborhood information.

Another perturbation method is simply to add noise to all data points (regardless of neighbors), or to all the weights. A technique of this type was studied in [6], but a detailed discussion of how the noise was added was omitted. We implement a "noise" method, where noise is added to the dissimilarity matrix before graph construction. The noise added to each dissimilarity is Gaussian with variance γ^2, then the dissimilarity matrix is checked to be sure all dissimilarities remain non-negative (set to 0 if negative after noise added) and to enforce $d(x, x) = 0\ \forall x$. A k-NN graph is constructed after the noise has been added and the entire process is repeated T times. The final adjacency matrix for the "noise" method retains edges that appear in more than cT of the repetitions and keeps original weights for those edges.

In the next section, we introduce two new perturbation methods motivated by the Isoband application.

4.3 New perturbation proposals

The idea of the local noise model of [9] is to perturb each data point in a way that depends on its average distance to nearest neighbors. However, our nearest neighbors are variables as measured by correlations, not the data points. So, instead of perturbing the data points, we will perturb the variables, and in essence, "smooth" rather than add noise. First, we standardize the variables. Then, using our dissimilarity $1 - |\hat{\rho}_{ij}|$, we find the k nearest neighbors of each standardized variable vector, X_i, which is n by 1. Then, assign a new value $\tilde{X}_i$ to variable i as

$$\tilde{X}_i = (1-\delta)X_i + \frac{\delta}{k}\sum_{j=1}^{k} X_{j(i)}, \tag{4.2}$$

where $X_{j(i)}$ is the j^{th} nearest neighbor of X_i. The k-NN graph is then constructed using the modified vectors $\tilde{X}$. Note the value of the tuning parameter δ here is pre-determined. A variation would be to sample δ uniformly from $[0, 1]$ and combine multiple perturbed graphs as before. Finally, we set the edge weights in the final graph to the original correlations. We refer to this method as the local smoothing method where the principle of local smoothing is analogous to the approach we take in Section 2.3 to deal with high levels of noise.

Finally, we introduce a new graph perturbation method based on the bootstrap, where we resample the original data set with replacement T times and recompute the k-NN graph for each bootstrapped sample using new correlations. The final graph is aggregated over the T bootstrapped samples by keeping edges which appeared in more than cT of the bootstrapped graphs, and using the original weights for those edges.

4.4 Simulation results

The four graph perturbation methods: the edge perturbation, noise, local smoothing, and bootstrap methods were used to obtain different NN graphs prior to applying Isomap on various block diagonal designs (details below) with $T = 100$, $c = .5$, $\gamma = .001$, $\alpha = .1$, and $\delta = .5$. The values of c and T are kept constant for the bootstrap, the edge perturbation method, and the noise method for easier comparison. However, one could easily change either c or T (or both) for each method as desired. The choice of tuning parameters is discussed in Section 4.4.2 below. For all simulations, $n = 100$ and $p = 100$ or 200 as in Chapter III, with block sizes (50,30,20) and (100,60,40), with 100 replications for each setting. Recall from Chapter III, the two types of bandable covariance structures we use are:

$$\Sigma_1(\rho) \,:\, \sigma_{ij} = \rho^{|i-j|}\,, \qquad \Sigma_2(m) \,:\, \sigma_{ij} = \left(1 - \frac{|i-j|}{m+1}\right)_+, \tag{4.3}$$

which we concatenate as blocks. We use notation $\Sigma_1(0.7, 0.8, 0.9)$ to refer to a model with three AR(1) blocks with values of ρ of 0.7, 0.8, and 0.9, and size of the blocks as described above. Similarly, $\Sigma_2(1/2, 1/2, 1/2)$ has three triangular blocks, and each m is half of the corresponding block size. Block detection for block diagonal designs with Σ_2 was already good, but there was room for improvement for Σ_1.

4.4.1 Results for estimating block-diagonal structure

We compare average l_2 norm losses (Table 4.1) and average number of blocks detected (Table 4.2) for the four graph perturbation methods, as well as the sample covariance and $B_k(\hat{\Sigma}_{BD})$ (best estimator with correct ordering and with banding applied to known blocks), along with the Isoband without any perturbations. From the results in Tables 4.1 and 4.2, it is clear that the bootstrap and local smoothing outperform the other estimators, including Isoband itself. The noise method and

the edge perturbation method offer no improvement over Isoband. Because of the poor performance of the edge perturbation method and noise method, while we discuss choice of tuning parameters for them, we do not include them in our real data examples which follow in Sections 4.5 and 4.6.

Table 4.1: Average operator norm loss over 100 replications for block-diagonal covariance models. Block sizes are 50, 30, 20 for $p = 100$ and 100, 60, 40 for $p = 200$.

Setting	p	Sample	Banding	Isoband	Local Sm.	Bootstrap	Noise	Edge Pert.
Σ_1	100	4.17	1.72	1.85	1.74	1.71	1.87	1.84
(.7,.7,.7)	200	6.61	1.82	3.80	1.90	1.84	2.20	2.24
Σ_1	100	4.73	2.52	4.40	2.54	2.51	2.60	2.57
(.7,.8,.9)	200	7.76	3.47	3.58	3.51	3.49	3.63	3.62
Σ_2	100	6.42	4.25	4.25	4.21	4.28	4.24	4.25
$(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$	200	13.48	8.87	8.92	8.99	8.91	8.97	8.87
Σ_2	100	5.05	2.28	2.27	2.28	2.28	2.27	2.28
$(\frac{1}{10}, \frac{1}{3}, \frac{3}{4})$	200	10.34	4.57	4.54	4.55	4.61	4.54	4.56

Table 4.2: Average number of blocks found over 100 replications for block-diagonal covariance models. Block sizes are 50, 30, 20 for $p = 100$ and 100, 60, 40 for $p = 200$.

Setting	p	Isoband	Local Sm.	Bootstrap	Noise	Edge Pert.
Σ_1	100	2.13	2.88	2.95	2.14	2.13
(.7,.7,.7)	200	1.94	2.75	2.97	1.94	1.94
Σ_1	100	2.85	2.95	3	2.86	2.85
(.7,.8,.9)	200	2.81	2.95	3	2.81	2.81
Σ_2	100	3	3	3	3	3
$(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$	200	3	3	3	3	3
Σ_2	100	3	3	3	3	3
$(\frac{1}{10}, \frac{1}{3}, \frac{3}{4})$	200	3	3	3	3	3

4.4.2 Choice of tuning parameters

In the simulations the tuning parameters we used were: $c = .5$ for the fraction of perturbed graphs that an edge needed to appear in to be kept in the final graph (shared between the bootstrap, noise, and edge perturbation methods), $\gamma = .001$ for the standard deviation in the noise method, $\alpha = .1$ for the percentage of edges to remove and replace for the edge perturbation method, and $\delta = .5$ to govern the amount of smoothing for the local smoothing method. In the simulations, $T = 100$

was used for the number of bootstraps and perturbed graphs (shared between the bootstrap, noise, and edge perturbation methods).

Detailed investigations on the $\Sigma_1(.7, .8, .9)$ setting with p=100 reveal the following for the tuning parameters for each method. Local smoothing seems to work best for lower values of δ (.1-.5) and does not seem sensitive to the choice in that range. At higher values (.7 or .9), however, it begins to find more blocks than are actually present. The bootstrap does not seem sensitive to the choice of c if c is in the range from $.3 - .9$. When c is less than .3, the bootstrap does not offer any improvement over Isoband in terms of improving block detection. For edge perturbation, best results are for c in (.1-.7) and low α (.05 or .1). When c is greater than .7, this method finds too many blocks (i.e., too many edges are dropped from the original graph), and also finds too many blocks if $\alpha > .1$ (.2 or .3). Finally, for the noise method, choice of c does not affect performance if the amount of noise added is low ($\gamma = .0001$ or .001). Higher values of γ (.01 or .1) cause the method to find too many blocks at any level of c. Even at optimal settings, however, the noise method and edge perturbation method cannot compete with the bootstrap and local smoothing. Based on these results, we selected $c = .5$ for the bootstrap, noise method and edge perturbation. For method specific tuning parameters, we chose $\gamma = .001$ for the noise method, $\alpha = .1$ for edge perturbation, and $\delta = .5$ for local smoothing.

4.5 Protein consumption example

Recall the protein consumption data discussed in Section 3.5 with 25 countries and nine variables. Previously, we compared the sample covariance, thresholding, and Isoband on this data by applying the covariance estimators and examining their principal components by looking at clustering solutions on the space of the first two

principal components. Here we investigate how much the results are affected if the graph is perturbed by the bootstrap and the local smoothing method. We omit the other two perturbation methods because they were not competitive in simulations.

The variable order that was found by Isoband in Section 3.5 was eggs, meat, starchy foods, pork and poultry, cereals, dairy, pulses and nuts and oil-seeds, fish, and fruits and vegetables. The bootstrap was applied with $c = .5$ and with 1000 bootstrap replications and produced a variable order of eggs, starchy foods, pork and poultry, meat, cereals, pulses and nuts and oil-seeds, dairy, fish, and fruits and vegetables. With the bootstrap ordering, banding chose to retain one more diagonal than Isoband. The local smoothing method was applied with $\delta = .5$ and banding applied after local smoothing chose to keep two more diagonals than Isoband. The variable ordering recovered for local smoothing was dairy, meat, fish, pulses and nuts and oil-seeds, starchy foods, cereals, eggs, pork and poultry, and fruits and vegetables. The changes in the order suggest that we may see some changes to the principal components. Note that no blocks were found using Isoband, the bootstrap, or local smoothing (which distinguishes these results from thresholding, which isolated fruits and vegetables from the other variables). We now examine the principal components and clustering solutions for each estimator.

As before, we use the Krzanowski measure to compare the principal components of the bootstrap and local smoothing to Isoband with results in Table 4.3. We see that there are slight differences between these new principal components and those found by Isoband. However, values for $m = 2$ are very close so we do not expect the clustering solutions to be very different from the Isoband solution.

Clustering results are shown in Figure 4.1 for thresholding, Isoband, the bootstrap, and local smoothing. In each case we clustered the data into three clusters using the

Table 4.3: Krzanowski measure $K(m)$: principal eigenspaces of the bootstrap and local smoothing compared to Isoband.

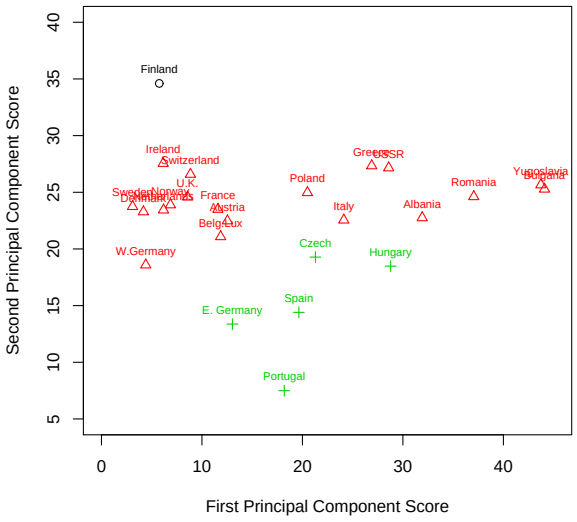
m	1	2	3	4	5	6	7	8	9
Bootstrap	.9998	1.98	2.97	3.92	4.65	4.85	6.16	7.03	9
Local smoothing	.9993	1.90	2.91	3.84	4.46	4.91	6.56	7.34	9

agnes hierarchical clustering [32]. Recall the thresholding results were very similar to the sample covariance results, with one cluster containing a single country (Finland) (see Figure 3.9). The bootstrap solution is almost identical to the Isoband solution, although the middle cluster for the bootstrap is a little tighter. For local smoothing, the changes to the variable ordering and keeping an additional diagonal are enough to change the clustering since Portugal is far enough from the other countries to be set aside as its own cluster. It appears that the Isoband and the bootstrap solutions are the best with the additional advantage of tighter clusters provided by the bootstrap.

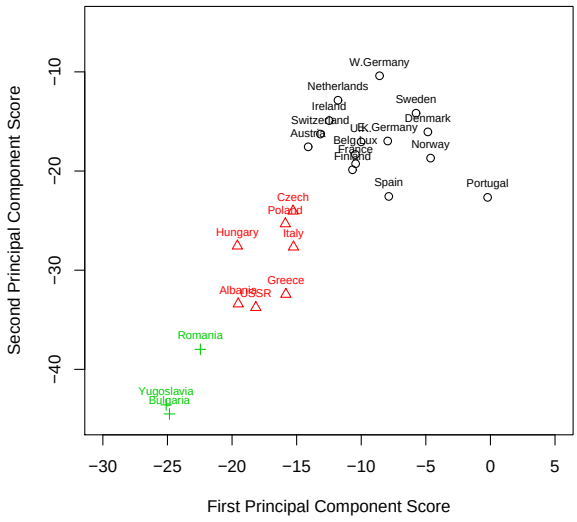
Although this example shows that the bootstrap may improve over Isoband, there are no blocks detected in this example. Next, we consider a larger gene expression data set, where blocks of variables are present.

4.6 Gene expression example

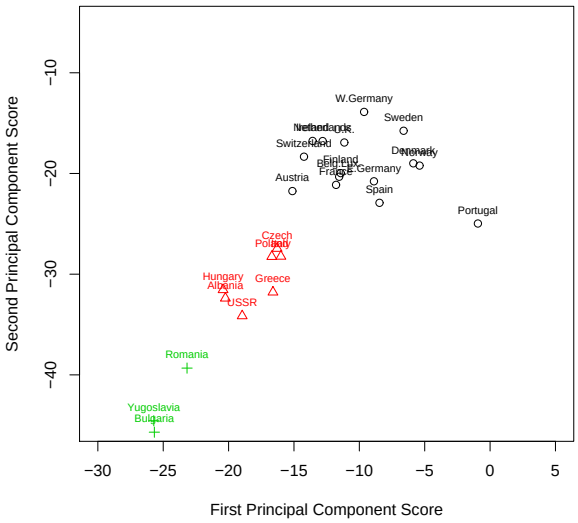
In this section, we show the advantages of improving graph construction in the context of finding blocks of variables in a covariance matrix for a gene expression data set. The full data set contains 2308 gene expression profiles measured on $n = 64$ samples which included four classes of small round blue cell tumors of childhood [33]. There were 23 samples from the Ewing family of tumors (EWS), 8 from Burkitt lymphoma, a subset of non-Hodgkin lymphoma (BL-NHL), 12 from neuroblastoma (NB), and 21 from rhabdomyosarcoma (RWS). For our analysis, we select a subset of the genes which carry the most discriminative information, as measured by the F-statistics computed using all four tissue classes. Specifically, the F-statistic is



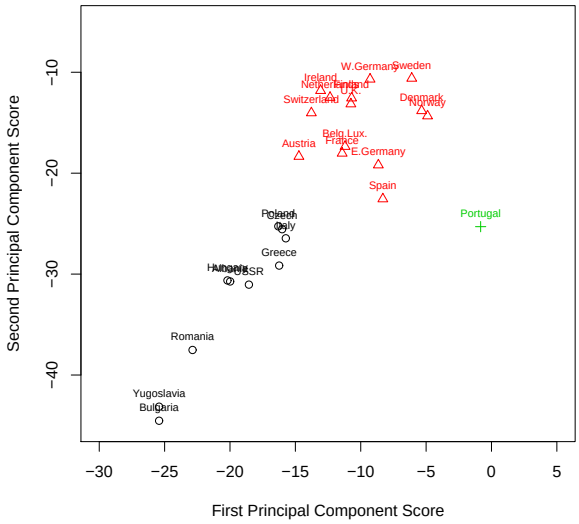
(a) Thresholding



(b) Isoband



(c) Bootstrap



(d) Local Smoothing

Figure 4.1: Agnes clustering of the protein consumption data in the space of the first two PCs after graph perturbation methods have been applied.

computed as

$$F = \frac{\frac{1}{k-1}\sum_{i=1}^{k} n_i(\bar{x}_i - \bar{x})^2}{\frac{1}{n-k}\sum_{i=1}^{k}(n_i - 1)\hat{\sigma}_i^2}, \tag{4.4}$$

where $k = 4$ is the number of groups, $n = 64$ is the total number of samples, n_i is the number in each class (given above), $\bar{x}_i$ and $\hat{\sigma}_i^2$ are the sample mean and variance of class i, and the overall mean is $\bar{x}$. We examined subsets of size 50, 100, 250, 500, and 1000 based on the F-statistics, but only discuss results for the subset of 50 here, since visualization of correlation matrices is easier with small p. Similar results were found for the other subsets.

With a reduced data set of 50 gene profiles across the 64 samples, we applied thresholding, Isoband, the bootstrap with 100 bootstrap replications and $c = .5$, and local smoothing with $\delta = .5$. Thresholding chooses a very small threshold and thresholds no correlations, and thus is identical to the sample covariance. Isoband finds three blocks of sizes 32, 11, and 7, and retains 4, 10, and 7 diagonals in each respectively. The bootstrap finds five blocks of sizes 24, 11, 7, 7, and 1, and chooses not to band anything. Local smoothing finds four blocks of sizes 24, 11, 8, and 7, retains only 2 diagonals in the first block, and does no banding in the smaller blocks. Even though the block sizes are similar for local smoothing and the bootstrap, there are some differences in the orderings as shown in Figure 4.2. Note that the size 24 blocks for the bootstrap and local smoothing do not contain the exact same genes.

One way to compare the estimators is to compare the blocks found to the four tissue classes. In Figure 4.2, columns are sorted by tissue class (labeled with class abbreviations: EWS, BL-NHL, NB, and RWS). The rows are the 50 genes sorted in the order found by each method with the largest blocks starting at the bottom of the figure. The gene expression values have been standardized for each gene; white corresponds to the largest standardized positive expression values, while the largest (in magnitude) standardized negative expression values are black. For the sample, the genes were simply sorted by hierarchical clustering (average linkage)

using the dissimilarity $d(i,j) = 1 - |\hat{\rho}_{ij}|$, where $\hat{\rho}$ is the sample correlation between variable i and j. For the other methods, genes are labeled based on their ordering in hierarchical clustering for ease of comparison.

Figure 4.2 shows that hierarchical clustering groups genes based on positive expression levels which correspond to the tissue classes, and there are clearly blocks of genes that correspond to each class. The Isoband plot appears different from the others due to the large block of 32 genes at the bottom of the figure, which covers classes BL-NHL and EWS. This is due to Isoband detecting the strong negative expression values for genes 43-50 for class BL-NHL as strong negative correlations with genes 1-24, and therefore merging the two blocks and retaining strong negative correlations in banding. It may be more appropriate here to use a dissimilarity of the form $1 - \hat{\rho}_{ij}$ rather than $1 - |\hat{\rho}_{ij}|$. The other two blocks still correspond to genes 36-42 and genes 25-35; however, the ordering within blocks is not exactly the same as the hierarchical solution. Note the changes to ordering are most significant in the first block since Isoband only retains 4 diagonals in that block compared to 10 out of 11 and 7 out of 7 diagonals in the other blocks.

Turning our attention to the bootstrap ordering, we see a new feature in the first block of genes (size 24, bottom of figure), because the bootstrap detected the large negative correlations between gene 50 and genes 9 and 20 and included gene 50 in that block. Gene 50 has large negative expression values for class BL-NHL rather than large positive ones, but that still means the first bootstrap block distinguishes class BL-NHL from the other classes. The next three blocks are genes 25-35, 36-42, and 43-49, just as in the Isoband solution, except with gene 50 missing from the last block, and each block corresponds to high positive expression levels for one of the remaining tissue classes. Again, the ordering within blocks is slightly different from

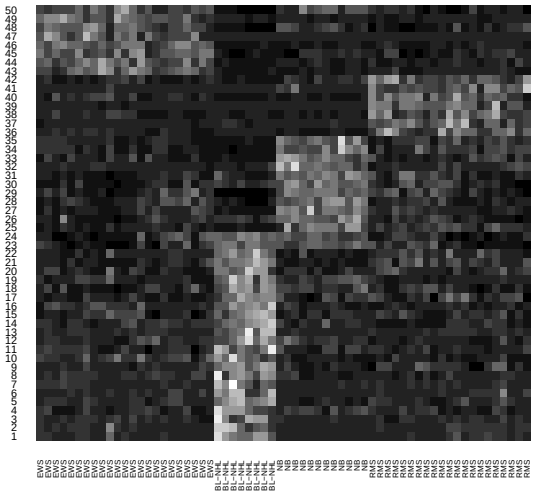
both the hierarchical solution and Isoband. The bootstrap isolated gene 11 (no other genes remained nearest neighbors to it in more than 50 % of the perturbed graphs) and it has high positive expression values for class BL-NHL.

Finally, we consider the local smoothing ordering where the blocks of genes found by local smoothing are genes 1-24, 25-35, 36-42, and 43-50. These blocks correspond to high positive expression levels for one tissue class each; the blocks in the figure are ordered to match the hierarchical clustering solution. Note that for the large block (bottom 24 genes of figure) the order of variables is not the same as in any of the other methods. This is significant because local smoothing bands this block more than both Isoband and the bootstrap, so genes far apart in this block have their correlations set to zero.

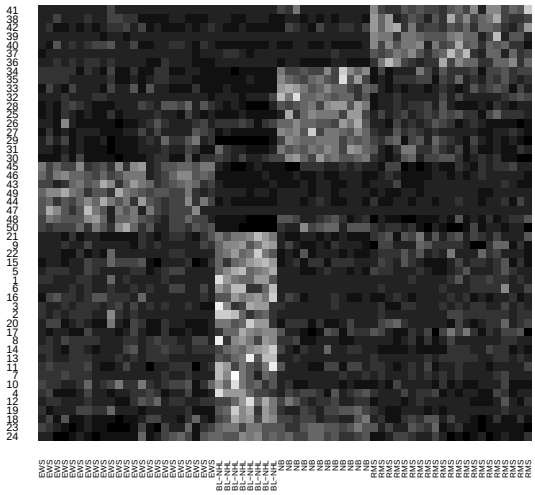
We also examined the number of edges in each graph. Isoband (with the traditional 3-NN graph) has 104 edges between the 50 variables. Local smoothing results in a graph with 99 edges and the bootstrap in a graph with 84 edges, so it does seem likely that the perturbation methods remove some erroneous edges.

When looking at the estimated matrices themselves, we will display the correlation matrices rather than the covariance matrices. To enhance the display of the correlation matrices, the variables were ordered according to a hierarchical clustering algorithm. All estimated correlation matrices were reordered in this order for ease of comparison and plotted as heatmaps in Figure 4.3. Thresholding is omitted since it is the same as the sample covariance.

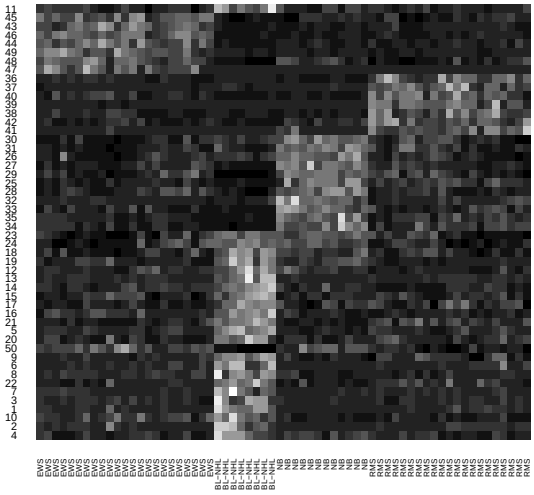
Figure 4.3 shows that Isoband retains many of the variables with strong positive correlations and a few negative ones, while the bootstrap results in even stronger blocks and a few additional negative correlations being retained. Local smoothing produces more zeros in the largest block, and zeros out all the negative correlations.



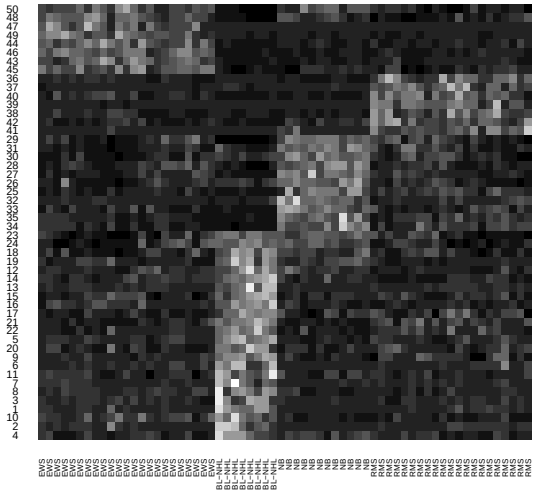
(a) Sample



(b) Isoband



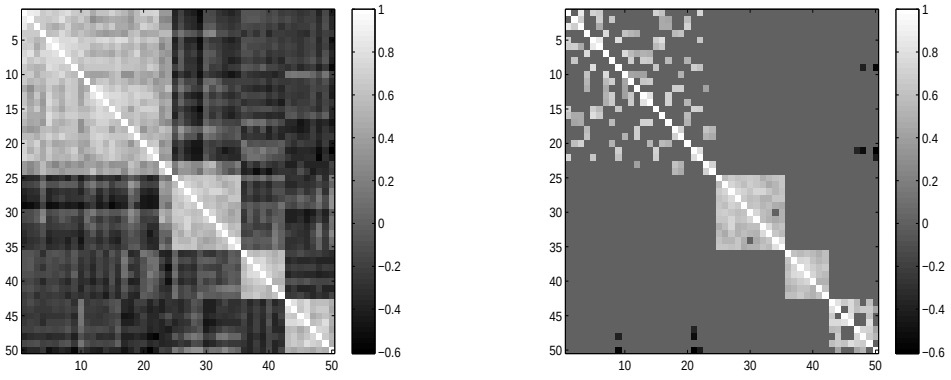
(c) Bootstrap



(d) Local Smoothing

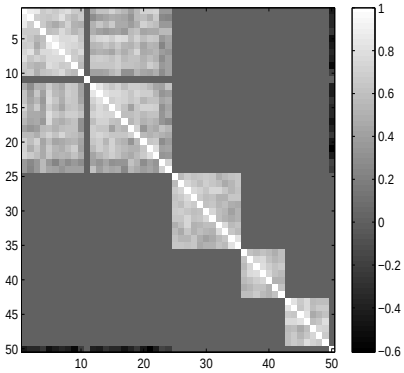
Figure 4.2: Heatmap of Khan data, columns are sorted by tissue class, rows are genes by block order for each estimator or hierarchical clustering for sample and labeled based on position in hierarchical clustering.

Both new methods result in blocks of genes which can distinguish the tissue classes, which is an improvement over Isoband. The bootstrap retains negative correlations while local smoothing does not, and is less sparse than local smoothing.

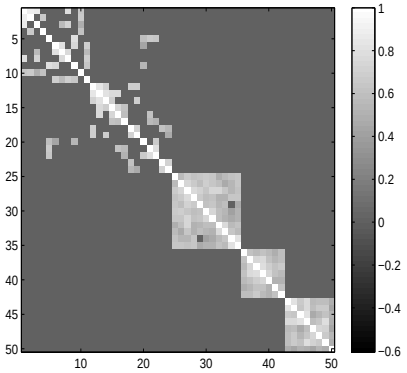


(a) Sample

(b) Isoband



(c) Bootstrap



(d) Local Smoothing

Figure 4.3: Heatmap of Khan data estimated correlation matrices for different covariance estimators. Black corresponds to correlations near -.6, white to correlations of 1. Variables were clustered via hierarchical clustering for visual scanning of blocks.

Overall, it appears that both perturbation methods result in "cleaner" block-diagonal estimates, which in this case clearly corresponds to the class structure in the data.

4.7 Discussion and future work

In this chapter we have concentrated on improving k-NN graph construction for use in Isoband and have shown the promise of two new methods – the bootstrap and local smoothing. Note the computational cost for local smoothing is much lower than that of the bootstrap. One issue we will address in future work is the effect of perturbation on k-NN graphs for larger k. We conjecture that perturbation will still

help eliminate spurious edges. Another extension is to ϵ-NN graphs. It is important to note that the bootstrap works well in our application because the k-NN graph is constructed on variables rather than observations. If the vertices of the graph were the observations themselves, resampling with replacement would result in some vertices missing and some replicated; a different resampling mechanism may make more sense in this case. Local smoothing, on the other hand, is straightforward to apply to a NN graph based on the observations. More sophisticated forms of smoothing will be investigated as well.

We also plan to investigate a rigorous procedure for choosing the tuning parameter based on some type of cross-validation. Ultimately, the details of the methodology will need to be adapted to the application at hand, but the general idea of graph perturbation to improve robustness of NN graphs is applicable to a wide range of statistical methods.

APPENDIX

Appendix

The description of the experiments and equipment was provided by the Morris lab, and is included for completeness and as needed for reference for future work.

A.1 Raman instrumentation

Raman spectra were collected using two different systems: a Raman microprobe optimized for collection in the near-infrared (NIR) [58] and a purpose-built, visible Raman microscope [18]. Briefly, the NIR system consists of an epi-illumination microscope frame (Olympus, BH-2) and a 400 mW 785 nm laser (Invictus, Kaiser Optical Systems, Inc.). The laser light is line-focused through a Powell lens (Stocker Yale) and into a 20x/0.75 NA Fluar objective (Carl Zeiss, Inc). For the visible Raman system, a research grade microscope (Nikon E600) and a 2 W 532 nm laser (Spectra Physics, Millenia II) were used. The circular beam profile of the Millenia II laser is reshaped into a line using a Powell lens and focused through a 4x/0.20 NA infinity-corrected objective (Nikon). For visible Raman hyperspectral imaging, a single axis scanning mirror (64240H, Cambridge Technology, Inc., Cambridge, MA) was used [18]. A LabVIEW (National Instruments, Austin, TX) program controlled the mirror's position by adjusting the voltage sent to the mirror control board through a 12-bit digital-analog converter. The mirror could be positioned to approximately $\pm 0.2\ \mu m$ with a setting of 1-3 ms. Raman scatters from both systems were collected using an $f/1.8$ axial transmissive spectrograph (Kaiser, HoloSpec). NIR and visible Raman scatter were detected using a back-thinned, deep depletion 1024×128 pixel

CCD camera (Andor Technology) or an 512 × 512 pixel electron-multiplying CCD camera (iXon Andor Technology), respectively. The spectral axis was calibrated (pixel to wavenumbers) using emission lines from a neon or argon discharge lamp. Curvature corrections and data analysis were performed in Matlab 6.1 using built-in and locally-written scripts.

A.2 Chemical components (dissimilar spectra)

Raman spectra of bovine bone, polyethylene, polystyrene, Teflon, Delrin, and PMMA were acquired using the visible Raman system. All spectra were collected using an acquisition time of 10 seconds and within the $700 - 1600$ cm^{-1} spectral range. Bovine bone specimens were obtained from a local abattoir and sectioned into $5 \times 10 \times 2$ mm blocks under constant irrigation using a diamond wheel saw. The sections were rinsed with calcium-buffered saline solution to remove any blood residues, and stored at -30^oC until required.

A.3 Fractured mouse bone (similar spectra)

Raman spectra of the fractured bone specimen embedded in PMMA were collected using the NIR Raman microprobe. A series of spectra were taken in parallel with the fracture from the edge at 100 μm intervals. Spectra were collected using a 7 minute integration time to ensure good s/n ratios. To prepare the fractured mouse bone specimens, a heavy rounded blade was dropped onto the tibia of a 10 month old wild-type mouse. Fractured mouse tibias were harvested according to a protocol approved by the University of Michigan Institutional Committee on Use and Care of Animals. The specimens were embedded in PMMA, sectioned, and polished to reveal the fractured ends of the bone.

A.4 PMMA curing

Koldmount (Vernon & Bishoff, Albany, NY) is a two-part acrylic resin commonly used to embed biological specimens for microscopy and archival preservation. The solid component (poly(methyl methacrylate) plus benzoyl peroxide as an initiator) and the liquid (methyl methacrylate monomer plus N,N-dimethyl-p-toluidine as an initiator) are mixed; the reaction proceeds quickly to produce a translucent (highly scattering) material. Koldmount powder (2.3 g) and Koldmount liquid (1.5 mL) were mixed together using vendor-supplied protocols. The mixture was stirred briefly at ambient temperature and immediately poured into a polystyrene cuvette. A second batch of Koldmount was prepared after 5 minutes, mixed and poured into the same cuvette, on top of the partially cured material. The fresh mixture was allowed to cure for 3 minutes (the minimum at which it no longer flows as a liquid). The cuvette was then turned on its side and placed on the microscope stage for Raman imaging. The reaction continued during the imaging.

Transects (30 in all) for the Raman image were collected on the 532-nm system with 500 mW excitation power and 4 seconds acquisition time. The spectra – initially 512 spectral values by 390 spatial pixels, 30 exposures altogether – were binned spatially to improve the s/n ratio. This gave a data set consisting of 3900 spectra (130×30), each with 512 values in the spectral dimension ($800 - 1500$ cm^{-1}).

A.5 Bone image

Visible hyperspectral imaging of mouse bone specimens embedded in PMMA were performed using the scanning mirror described previously [18], measured over the range $800 - 1500$ cm^{-1} (512 spectral values). Spectra were acquired with an integration time of 4 seconds per line; the excitation power was 300 mW. The image

comprised of 30 lines with a reduced pixel size of 300×50.

BIBLIOGRAPHY

Bibliography

[1] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. In *Foundations of Computer Science (FOCS 2002)*, pages 238–247, 2002.

[2] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in NIPS*, volume 14, Cambridge, MA, 2002. MIT Press.

[3] P.J. Bickel and E. Levina. Some theory for fisher's linear discriminant function, "naive bayes", and some alternatives when there are many more variables than observations. *Bernoulli*, 10(6):989–1010, 2004.

[4] P.J. Bickel and E. Levina. Covariance regularization by thresholding. Technical Report 774, UC Berkeley, 2007.

[5] P.J. Bickel and E. Levina. Regularized estimation of large covariance matrices. *Annals of Statistics*, 36(1):199–227, 2008.

[6] A. Blum, J. Lafferty, M.R. Rwebangira, and R. Reddy. Semi-supervised learning using randomized mincuts. *ICML-04*, 2004.

[7] I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, New York, 2005.

[8] B.O. Budevska, S.T. Sum, and T.J. Jones. Application of multivariate curve resolution for analysis of FT-IR microscospectroscopic images of in situ plant tissue. *Appl. Spectrosc.*, 57(2):124–131, 2003.

[9] M.Á. Carreira-Perpiñán and R.S. Zemel. Proximity graphs for clustering and manifold learning. In *Advances in NIPS*, volume 17, pages 225–232, Cambridge, MA, 2004. MIT Press.

[10] A. d'Aspremont, O. Banerjee, and L. El Ghaoui. First-order methods for sparse covariance selection. *SIAM Journal on Matrix Analysis and its Applications*, 30(1):56–66, 2008.

[11] E.D. Demaine and N. Immorlica. Correlation clustering with partial information. In *Proceedings of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2003)*, Princeton, New Jersey, 2003.

[12] D.K. Dey and C. Srinivasan. Estimation of a covariance matrix under Stein's loss. *Annals of Statistics*, 13(4):1581–1591, 1985.

[13] D.L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *PNAS*, 100(10):5591–5596, 2003.

[14] J. Fan, Y. Fan, and J. Lv. High dimensional covariance matrix estimation using a factor model. *Journal of Econometrics*, 2008. To appear.

[15] R. Furrer and T. Bengtsson. Estimation of high-dimensional prior and posterior covariance matrices in kalman filter variants. *Journal of Multivariate Analysis*, 98(2):227–255, 2006.

[16] K. R. Gabriel. Biplot display of multivariate matrices for inspection of data and diagnosis. In V. Barnett, editor, *Interpreting Multivariate Data*, pages 147–173. John Wiley and Sons, New York, 1981.

[17] S. Geman. A limit theorem for the norm of random matrices. *Annals of Probability*, 8:252–261, 1980.

[18] K. Golcuk, G. Mandair, A. Callender, N. Sahar, D. Kohn, and M.D. Morris. Is photobleaching necessary for Raman imaging of bone tissue using a green laser? *Biochim. Biophys. Acta*, 1758(7):868–873, 2006.

[19] L.R. Haff. Empirical bayes estimation of the multivariate normal covariance matrix. *Annals of Statistics*, 8(3):586–597, 1980.

[20] J.F. Hair, R.E. Anderson, R.L. Tatham, and W.C. Black. *Multivariate Data Analysis*. Prentice Hall, New Jersey, 1998.

[21] F.J. Harris. On the use of windows for harmonic analysis with the discrete Fourier transform. *Proceedings of the IEEE*, 66(1):51–83, 1978.

[22] D. Hoyle and M. Rattray. Limiting form of the sample covariance eigenspectrum in pca and kernel pca. *Advances in Neural Information Processing Systems*, 16, 2003.

[23] D. Hoyle and M. Rattray. Principal component analysis eigenvalue spectra from data with symmetry breaking structure. *Physical Review E*, 69, 2004.

[24] J. Huang, N. Liu, M. Pourahmadi, and L. Liu. Covariance selection and estimation via penalized normal likelihood. *Biometrika*, 93(1):85–98, 2006.

[25] J.H. Jiang, Y. Liang, and Y. Ozaki. Self-modeling curve resolution (SMCR): Principles, techniques, and applications. *Appl. Spectrosc. Rev.*, 37(3):321–345, 2002.

[26] J.H. Jiang, Y. Liang, and Y. Ozaki. Principles and methodologies in self-modeling curve resolution. *Chemom. Intell. Lab. Syst.*, 71(1):1–12, 2004.

[27] I. Johnstone. On the distribution of the largest eigenvalue in principal components analysis. *The Annals of Statistics*, 29(2):295–327, 2001.

[28] I. Johnstone and A. Lu. Sparse principal components analysis. *JASA*, 2004. Tentatively accepted.

[29] N. El Karoui. Operator norm consistent estimation of large dimensional sparse covariance matrices. *Annals of Statistics*, 2007a. To appear.

[30] N. El Karoui. Tracy-Widom limit for the largest eigenvalue of a large class of complex sample covariance matrices. *Annals of Probability*, 35(2):663–714, 2007b.

[31] B. Karrer, E. Levina, and M.E.J. Newman. Robustness of community structure networks. *Physical Review E*, 2007. To appear.

[32] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York, 1990.

[33] J. Khan, J.S. Wei, M. Ringner, L.H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C Peterson, and P.S. Meltzer. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7(6):673–679, 2001.

[34] W. Krzanowski. Between-groups comparison of principal components. *JASA*, 74(367):703–707, 1979.

[35] O. Ledoit and M. Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88:365–411, 2003.

[36] E. Levina and P.J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in NIPS*, volume 17, Cambridge, MA, 2005. MIT Press.

[37] E. Levina, A.J. Rothman, and J. Zhu. Sparse estimation of large covariance matrices via a nested lasso penalty. *Annals of Applied Statistics*, 2(1):245–263, 2008.

[38] E. Levina, A.S. Wagaman, A.F. Callender, G.S. Mandair, and M.D. Morris. Estimating the number of pure chemical components in a mixture by maximum likelihood. *J. Chemom.*, 21(1-2):24–34, 2007.

[39] E.R. Malinowski. Statistical F-tests for abstract factor analysis and target testing. *J. Chemom.*, 3(1):49–60, 1988.

[40] E.R. Malinowski. Abstract factor analysis of data with multiple sources of error and a modified Faber-Kowalski F-test. *J. Chemom.*, 13(2):69–81, 1999.

[41] V.A. Marcenko and L.A. Pastur. Distributions of eigenvalues of some sets of random matrices. *Math. USSR-Sb*, 1:507–536, 1967.

[42] M.D. Morris, N.J. Crane, L.E. Gomez, and M.A. Ignelzi. Compatibility of staining protocols for bone tissue with Raman imaging. *Calcif. Tissue Int.*, 74(1):86–94, 2003.

[43] M.E.J. Newman. Modularity and community structure in networks. *PNAS*, 103(23):8577–8582, 2006.

[44] M. Otto. *Chemometrics*. Wiley-Vch Verlag, Germany, 1999.

[45] D. Paul. Asymptotics of the leading sample eigenvalues for a spiked covariance model. *Statistica Sinica*, 2007. To appear.

[46] M. Pourahmadi. Joint mean-covariance models with applications to longitudinal data: unconstrained parameterisation. *Biometrika*, 86:677–690, 1999.

[47] A.J. Rothman, P.J. Bickel, E. Levina, and J. Zhu. Sparse permutation invariant covariance estimation. Technical Report 467, University of Michigan, 2007. Tentatively accepted by the *Electronic Journal of Statistics*.

[48] S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by local linear embedding. *Science*, 290(5500):2323–2326, 2000.

[49] S. Sasic and D.A. Clark. Defining a strategy for chemical imaging of industrial pharmaceutical samples on Raman line-mapping and global illumination instruments. *Appl. Spectrosc.*, 60(5):494–502, 2006.

[50] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. and Mach. Intel.*, 22(8):888–905, 2000.

[51] D.A. Skoog, F.J. Holler, and S.R. Crouch. *Principles of Instrumental Analysis*. Thomson Brooks/Cole, California, 2007.

[52] C. Stein. Estimation of a covariance matrix. 1975. Rietz Lecture, 39th Annual Meeting IMS.

[53] J.D. Storey and R. Tibshirani. Statistical significance for genome-wide studies. *PNAS*, 100(16):9440–9445, 2003.

[54] C.P. Tarnowski, M.A. Ignelzi, W. Wang, J.M. Taboas, S.A. Goldstein, and M.D. Morris. Earliest mineral and matrix changes in force-induced musculoskeletal disease as revealed by Raman microspectroscopic imaging. *J. Bone Miner. Res.*, 19(1):64–71, 2004.

[55] J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[56] J.A. Timlin, A. Carden, M.D. Morris, J.F. Bonadio, C.E. Hoffer, K.M. Kozloff, and S.A. Goldstein. Spatial distribution of phosphate species in mature and newly generated mammalian bone by hyperspectral Raman imaging. *J. Biomed. Optics*, 4(1):28–34, 1999.

[57] J.A. Timlin, A. Carden, M.D. Morris, R.M. Rajachar, and D.H. Kohn. Raman spectroscopic imaging markers for fatigue-related microdamage in bovine bone. *Anal. Chem.*, 72(10):2229–2236, 2000.

[58] E. Widjaja, N. Crane, T.C. Chen, M.D. Morris, M.A. Ignelzi, and B.R. McCreadie. Band-target entropy minimization (BTEM) applied to hyperspectral Raman image data. *Appl. Spectrosc.*, 57(11):1353–1362, 2003.

[59] E. Widjaja and M. Garland. Pure component spectral reconstruction from mixture data using SVD, global entropy minimization, and simulated annealing. Numerical investigations of admissible objective functions using a synthetic 7-spectra data set. *J. Comput. Chem.*, 23(9):911–919, 2002.

[60] W.B. Wu and M. Pourahmadi. Nonparametric estimation of large covariance matrices of longitudinal data. *Biometrika*, 90:831–844, 2003.

[61] M. Yuan and Y. Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

[62] Z. Zhang and H. Zha. Local linear smoothing for nonlinear manifold learning. Technical report, Penn State University, 2003. CSE-03-003.

[63] X. Zhu. *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, 2005.