Distance measures and whitening procedures for high dimensional data

Emily O'Riordan

School of Mathematics

Submitted in partial fulfillment of

the requirements for the degree of

Doctor of Philosophy

2023

Abstract

The need to effectively analyse high dimensional data is increasingly crucial to many fields as data collection and storage capabilities continue to grow. Working with high dimensional data is fraught with difficulties, making many data analysis methods inadvisable, unstable or entirely unavailable.

The Mahalanobis distance and data whitening are two methods that are integral to multivariate data analysis. These methods are reliant on the inverse of the covariance matrix, which is often non-existent or unstable in high dimensions. The methods that are currently used to circumvent singularity in the covariance matrix often impose structural assumptions on the data, which are not always appropriate or known.

In this thesis, three novel methods are proposed. Two of these methods are distance measures which measure the proximity of a point x to a set of points X. The simplicial distances find the average volume of all k-dimensional simplices between x and vertices of X. The minimal-variance distances aim to minimize the variance of the distances produced, while adhering to a constraint ensuring similar behaviour to the Mahalanobis distance. Finally, the minimal-variance whitening method is detailed. This is a method of data whitening, and is constructed by minimizing the total variation of the transformed data subject to a constraint.

All of these novel methods are shown to behave similarly to the Mahalanobis distances and data whitening methods that are used for full-rank data. Furthermore, unlike the methods that rely on the inverse covariance matrix, these new methods are well-defined for degenerate data and do not impose structural assumptions. This thesis explores the aims, constructions and limitations of these new methods, and offers many empirical examples and comparisons of their performances when used with high dimensional data.

Acknowledgements

Although a PhD can be a rather solitary undertaking, particularly during a global pandemic, this work would not have been possible without the support and encouragement of many people.

First and foremost, I would like to thank my academic supervisors. I will always be extremely grateful to have had Professor Jonathan Gillard as my mentor during these years. Professor Gillard has been a constant source of support and guidance, and has been extraordinary at putting my many academic anxieties to rest. My second supervisor, Professor Anatoly Zhigljavsky, has been fundamental in the production of the work that makes up this thesis, and ensured that no meeting was ever a boring one.

The PhD community within the School of Mathematics has been excellent. I made many friends who offered support, coffee and welcome distractions during my postgraduate studies here. Special thanks go to Dr Meg Scammell, Dr Nikoleta Glynatsi, Dr Henry Wilde, Dr Emily Williams, Dr Emma Aspland, Dr Chris Seaman and Sam Richardson. Nik and Henry are owed additional thanks for all their help with software development and office nerf fights.

To my friends outside of academia, thank you for being there during the highs and lows of this endeavour. In particular, to those in the university triathlon club, thank you for so much fun. Monday night track sessions in the pouring rain will forever hold a special place in my heart.

To Mum and Dad, thank you for being the best parents I could ever ask for. You've supported me through everything so wholeheartedly and in so many ways. I promise to get you that fridge one day. To Georgie and Jim, sorry for being your weird sister and thanks for always making me laugh. I love you all. To Millie, I miss you forever.

Ali, I will never be able to express my gratitude enough. Your support and love have been consistent, but your onion bhajis have not been. Thank you so much, here's to the next adventure.

Dissemination

A dissemination of the publications and presentations produced throughout this PhD project is detailed below.

Publications

The following manuscripts have been published, in order of publication:

- [85] J. Gillard, E. O'Riordan, and A. Zhigljavsky. **Simplicial and minimal-variance distances in multivariate data analysis**. *Journal of Statistical Theory and Practice*, 16(1):1-30, 2022. https://doi.org/10.1007/s42519-021-00227-7.
- [84] J. Gillard, E. O'Riordan, and A. Zhigljavsky. **Polynomial whitening for high-dimensional data**. *Computational Statistics*, 2022. https://doi.org/10.1007/s00180-022-01277-6.

Presentations

The following presentations were given during this PhD, in chronological order:

- **On simplicial distances with a view to applications in statistics**, *Cardiff OR/Stats Seminar*, Cardiff University. 2019.
- **On simplicial distances with a view to applications in statistics**, *NUMTA 2019*, University of Calabria, Italy. 2019.
- **Simplicial distances in high dimensional spaces (poster)**, *SIAM UKIE Annual Meeting*, University of Edinburgh. 2020.
- **Distance measures in clustering**, *Statistics in Big Data*, Cardiff University. 2020.

- **Simplicial distances in high dimensional spaces (poster)**, *TakeAIM Awards Ceremony, The Smith Institute*, Imperial College London. 2020.
- **Distances in correlated high dimensional spaces (poster)**, *Cardiff PGR Induction*, Cardiff University. 2020.
- **Polynomial whitening for high dimensional data**, *Cardiff 3 Minute Thesis*, Cardiff University. 2021.
- **Data workshop**, *Maths Support Service*, Cardiff University. 2022.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The ability to analyse and make sense of large amounts of data is fundamental to modern decision making processes. Data analysis is an umbrella term that can refer to any method of learning from data. This can range from simple investigations of data, such as the production of summary statistics, through to more advanced techniques often referred to as 'machine learning'. These methodologies are used in all manner of applications, including business intelligence [139], healthcare [230], security [52], insurance [40] and education [211], to name a few.

Many data analysis methods are reliant on measures of similarity between 'observations' in a dataset. A simple example of this could be measuring how similar different people ('observations') are based on some known properties ('variables') about them, such as their height, weight and age. There are many measures of similarity that are used in data analysis; in the case of numerical data these are referred to as 'distance measures'.

The rapid advancement of technology over the last few decades has seen exponential growth in computational power, computing efficiency and computer storage [200]. The ability to collect an abundance of data is therefore accessible to anyone. This may be in the form of 'big data', which is often used to refer to a lot of observations within a dataset [97], or high dimensional data, which refers to a large amount of variables in a dataset [124], or both simultaneously.

This thesis considers high dimensional data analysis methods. The ability to measure distance in high dimensional data is known to be problematic, which can in turn render many

data analysis methods unavailable. The research presented in this thesis addresses some of the issues found in measuring distances within high dimensional data, and proposes new methodology to be used. Furthermore, methods of pre-processing data for analysis can suffer from similar issues in high dimensional spaces, and so alternative techniques are suggested.

The rest of this chapter is organised as follows. Section 1.1 gives a brief summary of the motivation and aims of the research disseminated in this thesis. Section 1.2 details the structure of the thesis, and provides a short summary of each chapter that follows.

1.1 Thesis Objective

In multivariate data analysis, the Mahalanobis distance is often relied upon to measure proximity between observations in a dataset. The Mahalanobis distance transforms the data into a space where variables are uncorrelated and have unit variance. This removes the interference caused by interactions between the variables and accounts for issues caused by variables of differing scales. The Mahalanobis distance uses the inverse covariance matrix to perform such a transformation. However, in correlated data (where the distance measure is most useful), the covariance matrix is often singular, rendering the Mahalanobis distance unusable. The covariance matrix is also known to be singular in cases where the number of observations is less than the number of variables.

The method of data whitening is also relied upon heavily in multivariate data analysis as a pre-processing technique. Data whitening again transforms a dataset to a space with uncorrelated variables of unit variance, where further analysis can then be performed. The most common methods of whitening are also reliant on the inverse covariance matrix (specifically, the square root of the inverse covariance matrix), which is unavailable in correlated and (most) high dimensional data.

The objective of this thesis is to provide alternative methods that can account for correlations and varying scales in the same way that Mahalanobis distances and data whitening do, but that are available in the case of a singular covariance matrix. Two distance measures are proposed for this purpose, and one method of data whitening is given. An overview of these methods will be given in the next section.

1.2 Thesis Structure and Overview

This thesis contains six chapters, the first of which is this introductory chapter. An overview of the following chapters is given below.

- Chapter 2 provides a literature review on topics relevant to the research in this thesis. An introduction to distances measures is given, including the Mahalanobis distance which motivates much of the research performed in the rest of the thesis. The concept of data whitening is also outlined. A summary of some of the issues found in high dimensional data analysis is given, with specific focus on distance measures. Finally, an evaluation of estimators for the covariance matrix and its inverse is given, as such estimators are often used in Mahalanobis distances and data whitening methods.

- Chapter 3 discusses the simplicial distances, a multivariate distance measure first introduced in [192] and later built upon in [85] and this thesis. The metric uses the volumes of simplices to measure proximity, and can be tuned using parameters as to not suffer with degeneracy in the way that the Mahalanobis distance does.

- Chapter 4 introduces the minimal-variance distances, which were introduced in [85], having been inspired by trends seen when performing parameter experiments with the simplicial distances. This distance measure is found using an optimization function to minimize the variances of the distances produced, which helps to remove correlations in a dataset. A constraint is imposed to ensure similar behaviour to the Mahalanobis distance, should it exist.

- Chapter 5 further proposes the concept of minimal-variance whitening, having first been introduced in [84]. This method aims to transform a multivariate dataset to have variables with unit variance and no correlations between one another. This is again achieved using a constrained optimization method, and is usable in the case of degeneracy, unlike many popular methods of data whitening.

- Chapter 6 gives a summary of the research in this thesis and details future work that could be performed to build upon this work.

Chapter 2

Literature Review

This chapter aims to give an overview of the methods that have motivated the research detailed in this thesis. An introduction to several distance measures is given, as well as the reasons for their downfalls in high dimensional spaces. A similar analysis is then provided regarding data whitening methods for high dimensional data. Both of these topics suffer from the so-called 'curse of dimensionality', a term which encompasses the counterintuitive behaviour of data and data analysis methods in high dimensional spaces. A review of techniques proposed in the literature to evade this issue is given, with particular focus on the estimation of the covariance and inverse covariance matrix. The benefits and drawbacks of such techniques are also discussed. Overall, this literature review aims to illustrate the issues with distance measures and data whitening in high dimensional spaces, and motivates the need for the development of further methods to overcome such issues.

2.1 Introduction

High dimensional data is defined to be data with a large number of variables, where the definition of 'large' is dependent on the application, field and number of observations. The ubiquity of high dimensional data can be attributed to the continuous development of data collection and storage resources, as well as an ever increasing amount of computational power [200]. Typical examples of high dimensional datasets include microarray gene expression datasets [27, 90], where thousands of genes (variables) can be collected but

the number of tissue samples (observations) may only be in the range of tens. Financial datasets are also often high dimensional, as the number of features or returns per stock can be much higher than the number of stocks themselves [16]. Many other fields of research are now reliant on methods to deal with high dimensional data, including medical imaging [252], chemometrics [78], text classification [181] and astronomy [119], to name a few.

Classical statistical and data analysis methods often assume that the number of observations in a dataset is far greater than the number of variables [124], meaning there is enough information to accurately estimate the properties of the dataset. If the number of variables (or the dimensionality) of the dataset is similar to or greater than the number of observations, many of these classical methods are either incomputable (for example, simple linear regression [124]) or produce unreliable and sometimes nonsensical results [8]. The need for methodology to deal with high dimensional data has therefore rapidly increased as the presence of such data continues to grow.

Many multivariate data analysis methods are reliant on the ability to measure the proximity from one observation to another, including classification, clustering and outlier detection methods. The ℓ_p distances include the Euclidean and Manhattan distances, and are the most commonly applied distance measures in Euclidean space. The Mahalanobis distance is constructed specifically for multivariate datasets, and can account for correlations and variations in the scale of variables, the presence of which can often be problematic for ℓ_p distances.

Multivariate data analysis often requires that the data has been 'pre-processed' before being analyzed. Pre-processing is a broad term that covers many transformations of data, including dealing with missing variables or editing mistakes in data entries. Data whitening is a common pre-processing step which removes correlations between variables and standardizes all variables to have mean zero and unit variance. This can help to simplify many multivariate data analysis methods [130] and has been shown to improve the output of machine learning algorithms [112].

The Mahalanobis distance and data whitening are two of many data science techniques that are reliant on the inverse of the covariance matrix. The covariance matrix describes the spread of variables in a dataset, as well as how the variables interact with one another.

In low dimensional applications, the covariance matrix of a dataset is well estimated by the sample covariance matrix. However, this is often not the case when the dimensionality of the dataset grows [23]. Even if the sample covariance matrix is a good estimator, the inversion of a high dimensional covariance matrix can be extremely costly. It is common for the sample covariance matrix to be singular in high dimensions [235], caused by highly correlated variables or the presence of more variables than observations. In such cases, methods that are reliant on the inverse covariance matrix are unavailable. To overcome this, many estimators for the covariance matrix and its inverse have been produced for specific use with high dimensional data.

This literature review is structured as follows. In Section 2.2 some multivariate distance measures are defined, with a focus on the ℓ_p distances and the Mahalanobis distance. Section 2.3 outlines the method and applications of whitening a multivariate dataset. Section 2.4 delves further into the issues that arise from working with high dimensional datasets, including the counterintuitive nature of high dimensional geometry and the impact this has on measures of proximity between high dimensional objects. Finally, Section 2.5 details the reliance on the covariance matrix and its inverse in multivariate analysis, and gives a summary of methods to estimate these matrices in high dimensions.

2.2 Distance measures

A distance measure is a function that outputs a measure of proximity between two objects. In many contexts, these objects will be points within a set. Let $x, y, z \in X$ be three points in a set of points X. A function $f : X \times X \to \mathbb{R}$ is called a distance measure if the following three axioms hold:

1. $f(x,y) = 0 \Leftrightarrow x = y$,
2. $f(x,y) = f(y,x)$,
3. $f(x,z) \leq f(x,y) + f(y,z)$ (triangle inequality).

Many multivariate data analysis methods are reliant on the ability to accurately measure the distances between observations in a dataset. Examples of methods that depend on distance measures include the following:

- The K-means clustering algorithm [160] is an unsupervised learning algorithm that seeks to partition data into K groups, known as clusters. K clusters are formed using an initialization method [53], and the proximity from every point in the dataset is measured to the mean of each cluster using the squared Euclidean distance. Each point is then assigned to its closest cluster. The clusters are re-evalutated, the means of the new clusters are calculated, and the algorithm repeats until the means of each cluster converge (i.e. the means do not change with each new iteration). See Algorithm 1 in Chapter 3 for an algorithmic representation, or [7, 120, 221] for references on the method.

- The K-nearest neighbours (K-NN) algorithm [13, 80] is a supervised learning method used to classify observations. As the algorithm is a supervised method, the observations in the dataset must have pre-assigned class labels. The algorithm is then used to find the class labels of any new unlabelled observation, called the query point. The proximities from the query point to all other points in the dataset are calculated (usually using the Euclidean distance), and the points that return the K smallest distances are called the K nearest neighbours of the query point. The label of the query point is then taken to be the mode of the labels of the K nearest neighbours. For more information, see [3, 228].

- Several outlier detection methods are reliant on measuring the proximity of points to the rest of the dataset. Examples include K-NN based methods [17, 195] and other density based methods, such as DBSCAN [72] and Local Outlier Factor [43].

The distance measure chosen in such applications can have an influential effect on the outcome of the method [3, 212], and so making an informed and appropriate choice is integral to the success of data analysis methods.

2.2.1 ℓ_p distances

Many multivariate data analysis methods use the Euclidean distance by default to measure the distance between two points in Euclidean space. For points $x = (x_1, \ldots, x_d)$, $y = (y_1, \ldots, y_d) \in \mathbb{R}^d$, the Euclidean distance is defined as

$$D_{\mathrm{E}}(x, y) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}.$$

The Euclidean distance is a specific case of the so-called ℓ_p distances, which are defined as

$$D_p(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p \right)^{1/p}$$

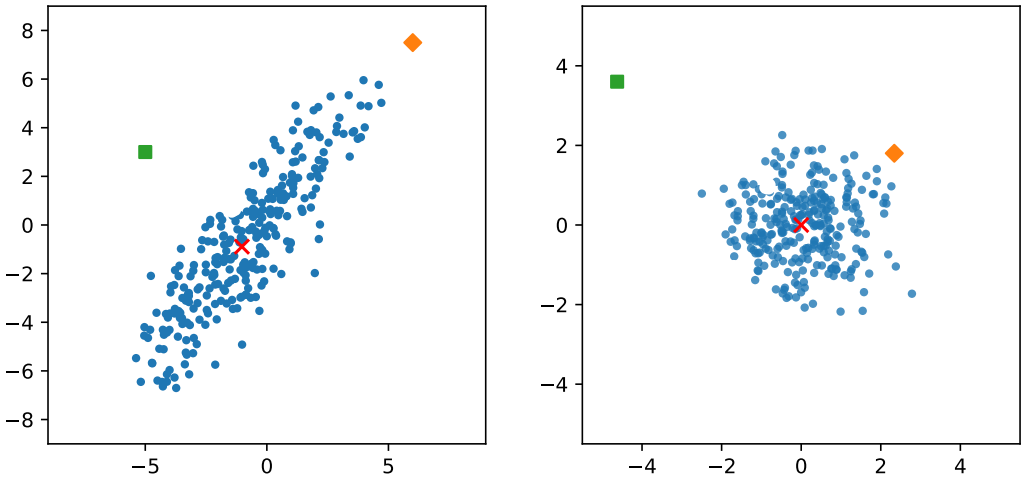
for a given parameter p. The Euclidean distance uses $p = 2$. Other common distances of this form include the Manhattan distance, which uses $p = 1$, and the ℓ_∞ distance which returns the maximum difference between two coordinates of x and y. Fractional ℓ_p metrics use a parameter $0 < p < 1$, but these are not well defined distance measures as they do not satisfy the triangle inequality [135].

The ℓ_p distances work well for relatively low dimensional datasets, where the dimensions within the dataset are roughly on the same scale. However, if variables are measured on different scales, or if there are correlations between the variables, the Euclidean distance (and other ℓ_p distances) can produce some misleading results. As an example, consider Figure 2.1a, which shows the plot of 300 observations generated from a 2-dimensional Gaussian distribution with covariance matrix $\Sigma = \left(\begin{smallmatrix} 5 & 8 \\ 8 & 10 \end{smallmatrix}\right)$. The mean of this set of points is shown by the red cross. Two new points are added to the dataset, shown by the orange diamond and green square. If one of these points were to be identified as an outlier by eye, the green square would be the obvious choice. However, using the Euclidean distance to measure the distance from the mean of the dataset to the orange diamond gives a distance of 14.89, and the distance to the green square is 8.03. Therefore, using the Euclidean distance, the orange diamond would be labelled as the outlier before the green square.

This downfall of the Euclidean distance (and other ℓ_p distances) is due to the fact it only accounts for the two points being measured, and does not take the distribution of any wider data into consideration. If data is high dimensional or more complex than the data given in Figure 2.1a, the need for a reliable distance measure is imperative.

2.2.2 Mahalanobis distance

The Mahalanobis distance was first introduced by P.C. Mahalanobis in 1936 [163]. Let $X \in \mathbb{R}^{d \times N}$ be a set of d-dimensional points with empirical mean vector $\mu \in \mathbb{R}^d$ and empirical covariance matrix Σ. The Mahalanobis distance from a point $x \in \mathbb{R}^d$ to the set X is



(a) Original correlated data

(b) Transformed data

Figure 2.1: (a) A set of 300 points generated from a Gaussian distribution with correlations (blue points), with empirical mean shown by the red cross. Two further points have been added (orange diamond and green square). (b) The same dataset after being decorrelated and standardized by Mahalanobis whitening.

defined as follows:

$$D_M(x, X) = \sqrt{(x-\mu)^\top \Sigma^{-1} (x-\mu)}. \tag{2.1}$$

This can also be viewed as the distance from x to the point μ with respect to the set X. The distance measure is therefore not limited to finding distances between points and distributions; the point μ can be replaced in Equation (2.1) by another point $y \in \mathbb{R}^d$ to find the Mahalanobis distance between x and y with respect to the set X. If the set X has covariance matrix $\Sigma = I$, where I is the d-dimensional identity matrix, the Mahalanobis distance reduces to the Euclidean distance.

By using the inverse of the covariance matrix, the Mahalanobis distance removes correlations in the data and scales every variable to have unit variance. This transforms the data from its elliptical shape, like the data in Figure 2.1a, to have a spherical shape, as seen in Figure 2.1b. The Euclidean distance is then used to find the distances in the transformed space. Using the example in Figure 2.1, the Mahalanobis distance from the mean of the blue points to the orange diamond is 4.18, and is 8.29 to the green square, meaning the green square would be labelled as the outlier as hoped.

The Mahalanobis distance is clearly an extremely valuable measure in multivariate data

analysis, and has many applications including cluster analysis [87, 260], outlier detection [193], text classification [218], Bayesian inference [10, 67] and image processing [187, 269]. A wide range of fields benefit from implementing the Mahalanobis distance, with applications in finance [55, 224], ecology [73], genetics [165, 178] and other medical fields [58, 225], to name a few.

The main attraction of the Mahalanobis distance is its ability to intuitively measure distances in elliptically-distributed data. Data is elliptically distributed when correlations are present in the data, meaning the Mahalanobis distance is particularly relied upon in such settings. However, correlations in a dataset are often the cause of singularity (or near-singularity) in the covariance matrix, meaning the inverse of the covariance matrix is either nonexistent or unstable. Therefore, in the cases where the Mahalanobis distance is needed the most, it is often unavailable. The covariance matrix of a dataset is also singular when the number of variables is greater than the number of observations, as is common in many modern datasets. In such circumstances, it is common to use the Moore-Penrose pseudoinverse of the covariance matrix, or find some alternative way to compute the covariance matrix. Both of these methods will be discussed in due course.

2.3 Data whitening

Data whitening is a method of transforming a dataset to have decorrelated and standardized variables. Fully decorrelated data possesses a diagonal covariance matrix, and standardized data has unit variance for each variable [107]. In the case of non-degenerate data, the covariance matrix of a whitened dataset is the identity matrix. By removing the elliptic structure of the data through such whitening transformations, more interesting and complex structures can be uncovered that may have previously been hidden in correlations, such as clusters, outliers or sparsity [115, 156]. Furthermore, the orthogonality of whitened variables can improve the computational time and performance of many statistical and machine learning methods [112, 133, 145, 249, 275].

Several methods of whitening are outlined in [130], including methods that are scale-invariant and methods aimed at improving dimension reduction. The most commonly used whitening technique is Mahalanobis whitening. Let X, Σ and μ be defined as in

Section 2.2.2. The dataset X is transformed by Mahalanobis whitening as follows:

$$X_{\Sigma^{-1/2}} = \Sigma^{-1/2}(X - \mu).$$

This method is popularly used to whiten data before performing many classical multivariate analysis processes. The transformed data $X_{\Sigma^{-1/2}}$ has zero-valued mean and the $d \times d$ identity matrix I as the covariance matrix.

The success of Mahalanobis whitening depends on the ability to compute $\Sigma^{-1/2}$ in a way that is both accurate and stable. However, it is common for big, high dimensional data to be close to degeneracy or of low-rank [235], yielding unstable computations of $\Sigma^{-1/2}$. Numerous examples of this problem are observed in fields such as recommender systems [155, 273], finance [22], medicine [206], genomics [255], and social networks [157]. These issues also arise in generalized mixture models [261], multiple regression [99, 104], adaptive algorithms [24], and linear discriminant analysis [265]. This is because variables often possess (approximate) linear dependencies, resulting in a covariance matrix Σ that is singular, or very close to singularity. As such, the inverse of the covariance matrix does not exist, or is at least unstable, making it inadvisable or impossible to calculate $\Sigma^{-1/2}$.

Despite this, it has been demonstrated that applying a whitening transformation prior to data analysis methods, such as clustering [268], dimension reduction [209] or outlier detection [93], often results in better empirical results. Theoretically, Mahalanobis whitening underpins weighted least squares [207], PCA [116, 125], canonical correlation analysis [96] and most of the array of classic multivariate statistics methods [156, 166]. Crucially, decorrelated and standardized data greatly simplifies both theoretical and practical multivariate data analysis [6, 14, 54, 169, 231, 262].

Recent literature has shown that whitening can also be used to improve the training of neural networks [111]. Often, normalization is used in such training rather than whitening, due to ill-conditioned problems [161] and the great expense of computing a large inverse square root covariance matrix [118], despite whitening being preferable if it is possible [112].

As with the Mahalanobis distance discussed in Section 2.2.2, many methods attempt to circumvent the aforementioned problems by use of the square root of the Moore-Penrose

pseudoinverse Σ^{-} to Σ or some other estimator for the covariance matrix. Section 2.5 will discuss these estimators in more detail.

2.4 High dimensional data

As previously mentioned, many data analysis processes assume that the number of observations in a dataset is much greater than the number of variables. Classical theoretical results, such as the law of large numbers and the central limit theorem, are reliant on this assumption [124]. However, the advancements in data collection, storage capabilities and computational power over the last few decades have given rise to new formats of data that may not satisfy this assumption. It is commonly agreed that high dimensional data is defined to be a collection of observations with a large number of variables (or dimensions). Despite this, there is not an explicit definition of how many variables this is, or how it relates to the number of observations. Let d be the number of dimensions, and let N be the number of observations in a dataset. A few of the varied definitions of high dimensional data are as follows:

- Data with more than 3 variables [153],
- Data with more than 10 variables [20],
- Data with $d \geq N$ [217],
- Data with d much larger than N (often orders of magnitude larger) [44].

There are many other definitions available throughout the literature [245, 247]. High dimensional data, no matter the precise definition used, brings with it phenomena and complications that are not present in low dimensional spaces. For dimensions as low as $d = 5$, the geometry of Euclidean space begins to behave counterintuitively, as will be discussed in Section 2.4.1.

The 'curse of dimensionality' is a term coined by Richard E. Bellman [28] that is often used to encompass the issues caused by working in high dimensional spaces that are not present in three-or-lower dimensional spaces. These issues span many fields, including optimization [190], data mining [238] and machine learning models [30, 63]. The specific issues relating to distance measures will be covered in Section 2.4.2.

2.4.1 High dimensional geometry

High dimensional space is characterised by geometrical properties that are very different to the properties of two and three dimensional spaces. Some examples of these counter-intuitive properties are given below:

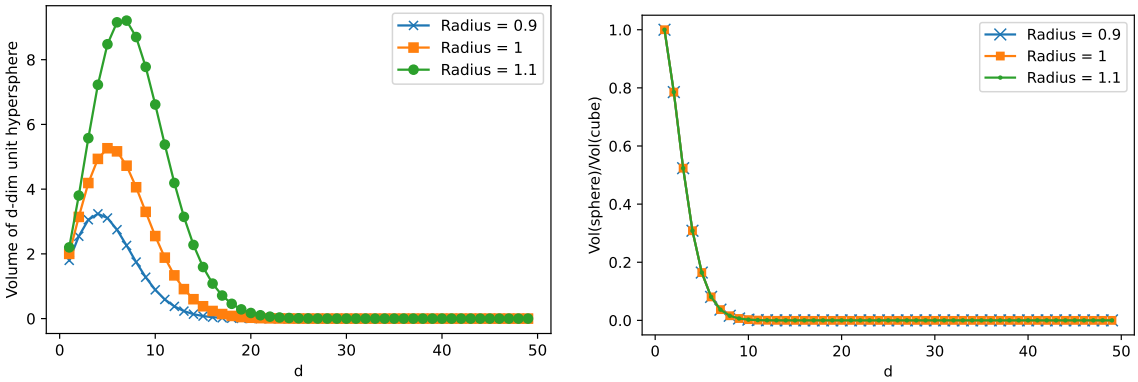
- Most of the volume of a high dimensional object is located near the surface of the object [38]. For example, in the unit hypersphere in d dimensions (for large d), most of the volume is located in a small annulus near the edge of the sphere. This implies that most points will have close-to-unit length, see [238] for more intuition on this.

- Most of the volume of the high dimensional unit hypersphere is also concentrated near its 'equator', for any equator chosen [38]. Therefore, for two random points generated on the d-dimensional unit hypersphere, there is a high probability that the two points are orthogonal (or nearly orthogonal). This can be generalized to N points, where there is high probability that the N points will be pairwise orthogonal [91].

- Let $S_{d,r}$ be the d-dimensional hypersphere with radius r. As $d \to \infty$, $\text{Vol}(S_{d,1}) \to 0$; that is, the volume of the d-dimensional unit hypersphere goes to zero. Figure 2.2a illustrates this phenomenon for hyperspheres with radii equal to 0.9, 1 and 1.1. Zimek et al. [274] explain that this should be viewed as the ratio of the volume of the hypersphere to the volume of the hypercube with side lengths 1 (the volume of which will always be 1). This makes the values plotted in Figure 2.2a unitless, allowing for comparisons between volumes of different dimensions.

- Let $C_{d,r}$ be the d-dimensional hypercube with radius r (i.e. edge length $2r$). Then, as $d \to \infty$, $\text{Vol}(S_{d,r})/\text{Vol}(C_{d,r}) \to 0$. That is, the ratio of the volume of a hypersphere with radius r to the volume of a hypercube with radius r tends towards zero as d increases, as demonstrated in Figure 2.2b. This indicates that most of the volume of a hypercube is located in its corners [238, 239]. See Figure 2.4 of [38] for a visualization of this.

Properties like those given above serve as a warning for high dimensional data analysis: methods that perform well in low dimensional spaces are not guaranteed to perform sim-



(a) Vol($S_{d,r}$) (b) Vol($S_{d,r}$)/Vol($C_{d,r}$)

Figure 2.2: For $r = 0.9, 1, 1.1$: (a) The volume of d-dimensional hyperspheres with radius r as d increases. (b) The ratio of the volume of d dimensional unit hyperspheres with radius r over the volume of the d-dimensional hypercube with radius r as d increases.

ilarly in high dimensional spaces. The aforementioned properties imply that high dimensional space is mostly empty, suggesting that high dimensional data is often embedded in lower dimensional subspaces. There are many implications for measures of proximity between points in high dimensional data, some of which will be highlighted in the next section.

2.4.2 Distance measures in high dimensional data

As discussed in Section 2.2, many multivariate data analysis methods are heavily reliant on the ability to measure the distance between observations in a dataset. For high dimensional data, this is even more significant, as the ability to visualize and understand the data becomes more difficult as the dimension d grows. However, measures of proximity are highly susceptible to the curse of dimensionality. Some examples of the effects of high dimensionality on distance measures are given below.

Diminishing relative contrast For a distance measure D, let $D_{\min}(d)$ and $D_{\max}(d)$ be the minimum and maximum distances measured between any two points in a d-dimensional dataset (with no assumptions on the distributions of this dataset). Define the relative contrast of the distance as

$$R(d) = \frac{D_{\max}(d) - D_{\min}(d)}{D_{\min(d)}}.$$

Beyer et al. [33] showed that as d increases, the expected relative contrast of distances tends towards zero:

$$\lim_{d\to\infty} E\left(R(d)\right) \to 0.$$

That is, as the dimension increases, the difference between the minimum and the maximum distances measured in a dataset becomes smaller and smaller. This phenomenon is often known as distance functions losing their usefulness, or the 'concentration effect', and is acknowledged in many fields of research [4, 12, 39, 59, 128, 134, 136].

Hinneburg et al. [103] show that ℓ_p distances with lower values of p seem to be better at handling this concentration effect. For example, the ℓ_1 distance often produces a higher relative contrast than the ℓ_2 distance in high dimensions, making the distance measure more 'useful'. Encouraged by this trend, Aggarwal et al. [5] show that using fractional ℓ_p metrics with $0 < p < 1$ can counter the concentration effect even more, making these potentially more useful in higher dimensional spaces. However, it has been shown that this is only applicable to uniformly distributed data [81], and that the performance of different norms are very much data dependent [175]. Furthermore, using $0 < p < 1$ gives a metric that does not satisfy the triangle inequality, so fractional ℓ_p metrics are not formal distance measures by the definition given in Section 2.2. This metric can therefore not be used with any methods that assume the triangle inequality holds [71, 138].

Furthermore, for the Euclidean distance (and therefore likely other ℓ_p distances like the Manhattan distance), it has been shown that the decreasing relative contrast is only applicable when the variables are independent and identically distributed [69], and the effect is less extreme for data with correlations, or data made up of a mix of distributions (such as clustered data or Gaussian mixture models) [33, 108, 274].

Irrelevant attributes As dimensionality increases, the possiblity for irrelevant attributes grows. The presence of irrelevant attributes decreases the signal-to-noise ratio of a dataset, which reduces the separability of points, masking the relevant variables and worsening the relative contrast of distances [31, 274].

However, if all attributes are relevant to the data (i.e. they all help to characterise clusterings or distributions), the separability of data is likely to grow in a way that will not suffer due to the concentration effect [108]. Relevant attributes are likely have correlations be-

tween each other, which is shown in [69] to be an important characteristic to avoid the concentration effect. The effect of relevant versus irrelevant attributes has also been noted in [32] and [59].

Measuring data with d > N In some settings, high dimensional data specifically refers to data with a greater number of dimensions than observations. This type of data is often known as high dimension low sample size (HDLSS) data, and is commonly found in fields such as genetics [92], chemometrics [95] and image analysis [264]. HDLSS data has implications for many types of data analysis. For example, many machine learning models struggle with overfitting [159, 208] in such circumstances.

Within the context of distance measures, HDLSS data causes issues for the Mahalanobis distance as there is not enough data present to accurately estimate the covariance matrix using the sample covariance matrix. Other methods for estimating the covariance matrix in such circumstances will be discussed in Section 2.5.2. Even if the covariance matrix can be calculated reliably, the inversion required to use the Mahalanobis distance can also cause issues: inverting a covariance matrix of size $d \times d$ becomes costly as d increases, and the estimated covariance matrix (not necessarily the classical sample covariance matrix) is likely to be singular. In the next section, the estimation of covariance matrices and inverse covariance matrices in high dimensions will be considered.

2.5 Covariance and inverse covariance matrices

The covariance matrix of a d-dimensional dataset is a $d \times d$ matrix that characterises the variance and covariance of the variables. The variance of the ith variable is given by the ith diagonal entry in the covariance matrix. The covariance between variable i and variable j is given by the (i, j)th entry of the covariance matrix. The covariance matrix is also known as the variance-covariance matrix, and is denoted here by Σ.

The covariance matrix is used for many data analysis methods, such as optimization [210] and dimension reduction [123, 172], with applications in fields such as economics [148], image processing [177] and chemical analysis [213]. As already discussed, it is fundamental to the implementation of both the Mahalanobis distance and data whitening.

2.5.1 The sample covariance matrix

The population covariance matrix Σ is the 'true' covariance matrix, if the whole population of the data is known. If data is generated synthetically from the normal distribution, the covariance matrix used to generate the data is the population covariance matrix. It is not common to know the population covariance matrix, and so the sample covariance matrix $\tilde{\Sigma}$ is often used to estimate it. For a dataset $X = \{x_1, \ldots, x_N\} \in \mathbb{R}^{d \times N}$ with sample mean $\tilde{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i \in \mathbb{R}^d$, the sample covariance matrix is typically defined as follows:

$$\tilde{\Sigma} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \tilde{\mu})(x_i - \tilde{\mu})^\top.$$

The denominator $N-1$ is due to the reliance on the estimated sample mean $\tilde{\mu}$. If the population mean is known and used, this can be swapped for a denominator of N. The sample covariance matrix is an unbiased and efficient estimator in well-conditioned data (that is, data that is full rank and has $d << N$) [15]. However, for HDLSS data, the sample covariance matrix is shown to no longer estimate the population covariance matrix well [23, 48]. Furthermore, the sample covariance matrix is singular in HDLSS settings, meaning it is not possible to calculate the Mahalanobis distance or perform data whitening. There are two options to alleviate this issue: find another estimate of the covariance matrix, or find an alternative to the inverse of the covariance matrix. The former of these options will be considered in the next section, and the latter in Section 2.5.5.

2.5.2 Alternative covariance matrix estimates

In order to circumvent the issues with the sample covariance matrix in high dimensions, many alternative methods of calculating or regularizing covariance matrices have been suggested. These methods often impose structural assumptions on the covariance matrix, and take advantage of such structures. A few of these methods are outlined below.

Banding In many settings, such as those considering spatial and temporal data, it is appropriate to assume that if the indices of variable i and variable j are far apart, the covariance between variable i and variable j is negligible. That is, for a covariance matrix $\Sigma = \left(\sigma_{ij}\right)_{i,j=1}^{d}$ and some threshold t, it is assumed that the following is likely to hold:

$$|i - j| > t \implies \sigma_{ij} = 0.$$

Such matrices are called 'bandable covariance matrices', and the banded estimator given by Bickel and Levina [35] is:

$$\hat{\Sigma}_{ij} = \begin{cases} \tilde{\sigma}_{ij} & |i-j| \leq t, \\ 0 & \text{otherwise} \end{cases}$$

where $\tilde{\Sigma} = \left(\tilde{\sigma}_{ij}\right)_{i,j=1}^{d}$ is the sample covariance matrix. The performance of the banded estimator is highly dependent on the choice of the threshold value t, which is often found via cross-validation [189].

Tapering Building on the idea of bandable covariance matrices, Cai et al. [50] proposed the tapering of the off-diagonals of the sample covariance matrix, where some natural ordering of the variables is assumed. Their suggestion is dependent on a parameter h, with $1 \leq h \leq d$, such that $\hat{\Sigma} = \left(w_{ij}\tilde{\sigma}_{ij}\right)_{i,j=1}^{d}$ with weights w_{ij} defined as:

$$w_{ij} = \begin{cases} 1 & |i-j| \leq h/2, \\ 2 - \frac{2|i-j|}{h} & h/2 < |i-j| < h, \\ 0 & \text{otherwise.} \end{cases}$$

The authors show that this estimator has good empirical performance and often outperforms the banded covariance matrix estimator. Methods of choosing an appropriate parameter h for tapering estimators have also been studied [49, 50].

Many extensions on the idea of banded and tapered off-diagonals to estimate the covariance matrix have been proposed, see Chapter 2.1 of [141] and Chapter 6 of [189] for further details.

Thresholding Whereas banding and tapering assume some natural order in the variables to inform adjustments to the covariance matrix, the technique known as 'thresholding' does not require ordered variables. Instead, it is assumed that the covariance matrix is sparsely populated (i.e. that the majority of the off-diagonal elements of Σ are zero or close to zero). Rather than amending the entries of the sample covariance matrix based on the distance of the indices $|i-j|$, thresholding methods instead consider the absolute

value of the entry $|\tilde{\sigma}_{ij}|$ in the sample covariance matrix itself:

$$\hat{\Sigma}_{ij} = \begin{cases} \tilde{\sigma}_{ij} & |\tilde{\sigma}_{ij}| \geq g, \\ 0 & \text{otherwise}, \end{cases} \tag{2.2}$$

where g is a user-defined threshold. This estimator was suggested by Bickel and Levina [34] and Karoui [70], and is invariant to permutations of the variables. In [202] a method is introduced using more generalized thresholding functions, rather than a hard threshold as in Equation (2.2). An adaptive thresholding method is suggested in [46] in which each individual entry $\tilde{\sigma}_{ij}$ has its own threshold g_{ij}, rather than a universal threshold g for all entries of the sample covariance matrix. This leads to a more flexible estimator but requires further parameter estimation.

Dimension reduction It is common to use dimension reduction techniques when working in high dimensions to make the data more manageable, and then perform data analysis (including the calculation of the covariance matrix) on this new, lower dimensional dataset. Methods can be divided into two categories: feature selection and feature extraction. Feature selection is the method of choosing a subset of the original variables and creating a new dataset from these variables. This method is simple and cheap, but often requires domain-specific understanding of the data and risks losing a lot of information. Feature extraction is typically more involved, as it transforms the data to a new, lower dimensional set of variables while retaining as much information as possible.

Principal component analysis (PCA) is the most commonly used dimension reduction technique. The data is projected to a lower dimensional space (with the new dimension set by the user) where singularity and low rank are no longer an issue. However, in high dimensions this is not always straightforward, as PCA is calculated using the singular value decomposition (SVD) of the covariance matrix of the original data. Firstly, the SVD can be extremely computationally expensive as dimensions increase [136]. Secondly, PCA is often inconsistent in high dimensions due to the reliance on the sample covariance matrix [9, 184].

Shrinkage When $d > N$, it is known that the eigenvalues of the sample covariance matrix $\tilde{\Sigma}$ do not approximate the eigenvalues of the population covariance matrix Σ well

[219]. Specifically, the largest eigenvalue of $\tilde{\Sigma}$ is often larger than the largest eigenvalue of Σ, and the smallest eigenvalue of $\tilde{\Sigma}$ is smaller than the smallest eigenvalue of Σ. See Figure 2.2 of [189] for empirical evidence of this phenomenon. To counteract this, 'shrinkage' methods were introduced to pull the eigenvalues closer towards some central value(s).

Linear shrinkage estimators are of the form

$$\hat{\Sigma} = \gamma_1 T + \gamma_2 \tilde{\Sigma},$$

where $\tilde{\Sigma}$ is the sample covariance matrix, T is a $d \times d$ matrix called the shrinkage target matrix, and γ_1, γ_2 are parameters to be found, known as the 'shrinkage coefficients'. The shrinkage target matrix T is typically a well-conditioned matrix.

The Ledoit-Wolf shrinkage estimator [149] is defined as follows:

$$\hat{\Sigma}_{LW} = \gamma_1 I + \gamma_2 \tilde{\Sigma} = \frac{\beta^2}{\alpha^2 + \beta^2} \nu I + \frac{\alpha^2}{\alpha^2 + \beta^2} \tilde{\Sigma}, \tag{2.3}$$

where $\alpha^2 = \|\Sigma - \mu I\|_F^2$, $\beta^2 = E\left[\|\tilde{\Sigma} - \Sigma\|_F^2\right]$ and $\nu = \text{trace}\left(\tilde{\Sigma}\right)/d$. Here, $\|\cdot\|_F$ denotes the Frobenius norm. The estimator given in Equation (2.3) is proven to be the linear combination of the identity matrix I and the sample covariance matrix $\tilde{\Sigma}$ that fulfills the following optimization problem:

$$\min_{\gamma_1, \gamma_2} E\left[\|\hat{\Sigma} - \Sigma\|_F^2\right] \quad \text{s.t.} \quad \hat{\Sigma} = \gamma_1 I + \gamma_2 \tilde{\Sigma}.$$

Ledoit and Wolf [149] also suggests estimators for α, β and μ. These estimators are reliant on some structural assumptions of Σ. The estimator given by (2.3) is shown to be a consistent and successful estimator for the population covariance matrix, and is positive-definite. It has also inspired many other linear shrinkage estimators. Cross-validation techniques to estimate parameters have also been suggested [246]. Various alternative shrinkage target matrices have been proposed in the literature, including diagonal matrices [205], the single-factor matrix [147], the constant correlation model [148] and multi-target matrices [143]. Linear shrinkage models are highly dependent on the choice of the target matrix, which imposes some assumed structure on Σ. Shrinkage estimators also do not affect the eigenvectors of the covariance matrix. It has been shown that the sample eigenvectors are not consistent when d is large [122], meaning shrinkage estimators may not be appropriate to use with any methods relying on eigenvectors, such as PCA [154].

Many non-linear shrinkage estimators have also been proposed to circumvent the issue of choosing an appropriate target matrix. These methods often begin with the spectral decomposition of the sample covariance matrix: $\tilde{\Sigma} = P\tilde{\Lambda}P^\top$, where P is the matrix of eigenvectors and $\tilde{\Lambda}$ is the diagonal matrix with the eigenvalues of $\tilde{\Sigma}$ as its diagonal entries. The estimator is then of the form

$$\hat{\Sigma} = P\Delta P^\top,$$

where Δ is a diagonal matrix whose entries are shrunken values of the eigenvalues in $\tilde{\Lambda}$. It is shown in [149] that linear shrinkage estimators can be written in this form, and in the linear case all eigenvalues have the same shrinkage function applied to them. In non-linear shrinkage, each eigenvalue can be adjusted individually. This produces an estimator that is at least as good as a linear shrinkage estimator [150]. For some examples of non-linear shrinkage, see [1, 146, 254]. For more information on shrinkage estimators, see [150], Chapter 4.1 of [189] or Chapter 3 of [141] for reviews on common methods.

Many further covariance estimators have been proposed in the literature that are not included here, such as modified Cholesky decomposition [110], factor-based models [74, 75], alternating projection models [102], Newton-type methods [194] and penalized methods [197]. For recent surveys on high dimensional covariance estimation, see [48, 141, 189].

2.5.3 The inverse covariance matrix

A square matrix $A \in \mathbb{R}^{d\times d}$ is non-singular if there exists a matrix $B \in \mathbb{R}^{d\times d}$ such that

$$AB = BA = I$$

where I is the $d \times d$ identity matrix. If such a matrix B exists, it is unique and is called the inverse of the matrix A, denoted A^{-1}.

The inverse covariance matrix is relied upon for many applications, including linear discriminant analysis [117] and optimization methods [222], as well as for the Mahalanobis distance and data whitening. If the dataset has many correlations, some low-rank structure or has more variables than observations, it is likely that the sample covariance matrix is singular, rendering many of these data analysis methods unusable. In such cases, there are three options. Firstly, one can estimate the covariance matrix using the sample covariance

matrix, and find the 'generalized inverse' of this estimate, which will be discussed shortly. Alternatively, one can use one of the alternative covariance matrix estimates, as detailed in Section 2.5.2, and invert this if it is nonsingular. Finally, one can directly estimate the inverse covariance matrix, which will be considered in Section 2.5.5.

Generalized inverse of a matrix If a matrix A is singular, the inverse of the matrix A^{-1} does not exist. In such cases, a generalized inverse of the singular matrix can be used. A generalized inverse of a $d \times d$ matrix A is a $d \times d$ matrix A^g which satisfies $AA^gA = A$ [243]. If the matrix A is not singular, the generalized inverse is exactly the inverse $A^g = A^{-1}$ [236].

There are many different types of generalized inverses, with different conditions and aims. Roger Penrose showed that for every matrix $A \in \mathbb{R}^{d \times m}$, there exists a unique matrix A^g satisfying the four following conditions, known as the Penrose conditions [186]:

1. $AA^gA = A$
2. $A^gAA^g = A^g$
3. $(AA^g)^\star = AA^g$
4. $(A^gA)^\star = A^gA$

where $A^\star$ denotes the conjugate transpose of A. If a matrix satisfies the first condition only, it is called a generalized inverse of A. If the first two conditions are satisfied by A^g, it is called a reflexive generalized inverse of A. The unique matrix that satisfies all four of the Penrose conditions is called the Moore-Penrose inverse, the Moore-Penrose pseudoinverse, or simply the pseudoinverse of the matrix A, and is often denoted by A^- [176, 186]. This will be discussed further in the next section.

Many other generalized inverses exist, including the following:

- The right-sided inverse is a matrix $A_R^{-1} \in \mathbb{R}^{d \times d}$ that satisfies $AA_R^{-1} = I$ [227]. The left-sided inverse of a matrix is defined analogously.
- A constrained generalized inverse is found by solving a system of linear equations, with the added constraint that the solution is in a given subspace. An example is the Bott-Duffin inverse of A [41, 65].

- The Drazin inverse A^D [66, 272] is often categorized as a generalized inverse, despite not always fulfilling the condition $AA^DA = A$.

See [29] for an extensive overview of alternative generalized inverse matrices.

2.5.4 The Moore-Penrose pseudoinverse

The Moore-Penrose (MP) pseudoinverse A^- is discussed in more depth here, thanks to its uniqueness and popularity in the literature. There are many different ways of computing the MP pseudoinverse of a matrix. In [215], seven fast methods of computation are discussed and the numerical stability of these methods is highlighted, as well as their computational complexity.

Construction of the Moore-Penrose pseudoinverse The MP pseudoinverse is classically computed through the singular value decomposition (SVD). For the matrix A, let $A = USV^\star$ be the SVD, where U is a unitary matrix, S is a diagonal matrix with the singular values $\{s_1, s_2, \ldots, s_d\}$ of A on the diagonal, and V is a unitary matrix. If the entries of A are all real, U and V are real, orthogonal matrices. Then the MP pseudoinverse is given by $A^- = VS^-U^\star$. The diagonal matrix S^- is calculated by taking the reciprocal of all nonzero elements in S, and leaving all zero elements in place:

$$S = \begin{pmatrix} s_1 & 0 & \ldots & 0 \\ 0 & s_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & s_d \end{pmatrix}, \quad S^-_{ii} = \begin{cases} 1/s_i & s_i \neq 0, \\ 0 & s_i = 0. \end{cases} \tag{2.4}$$

Computing the MP pseudoinverse via the SVD is relatively simple, accurate and numerically stable in a lot of cases. However, the cost of computing the SVD can be prohibitive for large matrices [215]. Furthermore, floating point computation can make it hard to know which singular values are very small, and which are exactly zero, creating the need to set a tolerance.

Another method of computing the MP pseudoinverse is via the rank decomposition of the $d \times d$ matrix A with rank $r \leq d$ [29, Theorem 5]. The rank decomposition is written $A = BC$, where B is a $d \times r$ matrix and C is a $r \times d$ matrix, both of rank r. Then the MP

pseudoinverse inverse is calculated as

$$A^- = (BC)^- = C^\top (CC^\top)^{-1} (B^\top B)^{-1} B^\top.$$

Much like the SVD method, this method is hindered by the time-intensive process of finding the rank decomposition of A and the inverses of $CC^\top$ and $B^\top B$. Furthermore, computing the inverses of the matrix products $CC^\top$ and $B^\top B$ can often introduce numerical issues, leading to severe loss of accuracy.

The QR decomposition can be used to calculate the MP pseudoinverse and can be found in a number of ways. The Gram Schmidt method can be used and is cheap to implement, but is an unstable algorithm [83]. Using Householder reflections [236] to calculate the QR decomposition is much more stable, but is computationally expensive and not parallelizable. The QR decomposition can be quicker to compute than the SVD, but can sometimes underestimate the rank of a matrix [215]. Many other methods of calculating the MP pseudoinverse have been suggested, including iterative methods [188], numerical methods [129] and methods using Cholesky factorization [60], to name a few.

Drawbacks of the Moore-Penrose pseudoinverse There are clearly many benefits to using the MP pseudoinverse, such as its uniqueness [236], generalization to the true inverse in full-rank cases [45] and that the MP pseudoinverse of the MP pseudoinverse is the original matrix: $(A^-)^- = A$. The MP pseudoinverse is also free from assumptions about the structure of the matrix A. These benefits, plus the simplicity of implementation, have seen the MP pseudoinverse used for many applications, including classification [248], clustering [140] and dimension reduction [232]. It is also used across many fields, such as image processing [57], medical imaging [216], analysis of genomic data [223] and financial applications [152], to name a few.

The MP pseudoinverse is constructed by finding the reciprocal of nonzero eigenvalues, and letting the inverse of any zero eigenvalues be equal to zero, see (2.4). By using *all* nonzero eigenvalues, the MP pseudoinverse can be adversely affected by very small eigenvalues. It can be difficult to differentiate between very small nonzero eigenvalues and zero eigenvalues, requiring a user-set threshold for eigenvalues to be classed as zero eigenvalues. Furthermore, any minor changes in the small eigenvalues of the matrix A cause huge changes in A^-, making the MP pseudoinverse very sensitive.

Consider the population covariance matrix Σ and the sample covariance matrix $\tilde{\Sigma}$ of a dataset X. In classical $d < N$ cases, the small eigenvalues of $\tilde{\Sigma}$ usually relate to noise and so the relevant variables in X can be removed, or the eigenvalue changed to zero. As discussed in Section 2.5.2, the sample covariance matrix in $d \geq N$ cases is not a consistent estimator. In these HDLSS circumstances, the small nonzero eigenvalues of $\tilde{\Sigma}$ do not necessarily relate to noisy variables, and the eigenvalues in Σ that correspond to these small eigenvalues in $\tilde{\Sigma}$ may not be small. Therefore, removing these variables from X or changing the eigenvalues to zero can cause a loss of information. It is shown that this, combined with the MP pseudoinverse, can cause inconsistency in applications such as regression and classification [196, 205]. See [109] for a study into the error caused in machine learning methods through the use of the MP pseudoinverse.

2.5.5 Alternative inverse covariance matrix estimates

As discussed previously, to find the inverse of the covariance matrix, it is common to find an estimator of the covariance matrix $\hat{\Sigma}$, and then invert this estimator to find $\hat{\Sigma}^{-1}$. If $\hat{\Sigma}$ is not invertible, one may need to use the MP pseudoinverse from the previous section, or some other method of inversion [101] to estimate the inverse covariance matrix. However, in high dimensions, this approach may not be ideal. As the dimension d of the covariance matrix increases, inversion of $\hat{\Sigma}$ becomes more time-intensive. Furthermore, any error in the estimator $\hat{\Sigma}$ may be greatly amplified by its inversion. Instead, a number of estimators have been proposed to find the inverse covariance matrix directly. These methods rely on assumptions on the structure of the data, and take advantage of such structures.

Banding If there is a natural ordering of the variables, a banding technique can be used to estimate the inverse covariance matrix, much like the method used to estimate the covariance matrix in Section 2.5.2. Wu and Pourahmadi [257] introduced the method of banding using the modified Cholesky decomposition for the inverse covariance matrix, which has been shown to be a consistent estimator [35, 258]. Further estimators of this type have since been proposed, see [25, 154, 151, 259].

Sparsity It is often assumed that the inverse covariance matrix should be a sparse matrix. Even if variables are correlated with one another, sparsity is common in the inverse

covariance matrix. If $\Sigma^{-1}_{ij} = 0$, the ith and jth variables are conditionally independent, given the other variables [82]. Imposing sparsity assumptions is closely related to the practice of setting entries of the inverse covariance matrix to zero, known as 'covariance selection' [64].

There are several proposed methods for estimating the inverse covariance matrix using sparsity assumptions. Let $\hat{\Omega}$ denote an estimator of the inverse covariance matrix. Penalized likelihood methods are amongst the most common estimation methods, and are often of the form:

$$\hat{\Omega} = \arg\min_{\Omega} \left\{ \text{trace}\left(\hat{\Sigma}\Omega\right) - \log|\Omega| + \sum_{i \neq j} P_w\left(|\Omega_{ij}|\right) \right\},$$

where $\hat{\Sigma}$ is the sample covariance matrix, and $P_w\left(|\Omega_{ij}|\right)$ is a penalty function on the off-diagonals of the matrix Ω with weights w. The ℓ_1 penalty $P_w(x) = w|x|$ is commonly used [82, 201, 267]. However, this approach is computationally very intensive, particularly in high dimensions [154].

If every column of the inverse covariance matrix is assumed to be sparse, column-by-column estimation can be used. Meinhausen and Bühlmann [173] propose using lasso regression techniques to estimate every column of the inverse covariance matrix, and several other similar methods have since been produced [47, 266]. Although these methods are less computationally intensive, they do require a stricter sparsity assumption than many other sparse inverse covariance matrix estimates.

For more information on estimators of the inverse covariance matrix that rely on sparsity assumptions, see [75, 142]. Estimating the inverse covariance matrix is clearly extremely reliant on structural assumptions of the matrix, which may not always be appropriate for the given application. If used incorrectly, sparsity and banding based methods can produce very inconsistent estimators of the true inverse covariance matrix.

2.6 Chapter summary

This chapter has summarized methods of navigating the curse of dimensionality in relation to distance measures, whitening and covariance matrix estimation. It is shown that the geometry of high dimensional spaces leads to problems in using classical methods that are often relied upon in low dimensional spaces, such as the ℓ_p distances.

The motivation for using the Mahalanobis distance in multivariate data is illustrated in Section 2.2.2. The Mahalanobis distance is most commonly required in data with correlated variables. However, it is exactly this property that often makes the Mahalanobis distance unusable, as correlations make the covariance matrix of the data singular and thus not invertible. The sample covariance matrix is also singular in the case of HDLSS data, which is increasingly common in modern data analysis. The same issues arise in the application of data whitening, which is discussed in Section 2.3.

The sample covariance matrix is a popular and unbiased estimator for the covariance matrix in low dimensions. However, when the dimensionality of the dataset outstrips the number of observations, the sample covariance matrix is no longer a good estimator for the population covariance matrix. In Section 2.5.2, a number of alternative estimators for the covariance matrix are given. Many of these estimators are specifically constructed for the HDLSS case, and rely on assumptions such as ordered variables or sparsity.

Section 2.5.4 discusses the famous Moore-Penrose pseudoinverse, which calculates a generalized inverse of a singular matrix. This method is commonly used due to its ease and good results in lower dimensions. However, it is shown that in high dimensions it can be difficult to compute and can cause a loss of information.

When working in high dimensions, the practice of finding an estimator for the covariance matrix and then inverting it may not be appropriate or practical. Firstly, the process of inverting a large dimensional matrix is extremely time consuming. Secondly, any errors in the estimator of the covariance matrix will be amplified by the inversion of the matrix. As such, many authors have proposed methods to approximate the inverse of the covariance matrix directly from the data itself, as explored in Section 2.5.5. These methods are also reliant on structural assumptions, again including variable ordering and sparsity.

The aim of this literature review is to show the circular issue of requiring methods to account for correlations and singularity in datasets, but not being able to use these methods because of such correlations and singularity. The alternative estimators proposed in the literature are often heavily reliant on structural assumptions of the data, which may not always be appropriate. These issues form the motivation for the research presented in this thesis. The methods proposed in Chapters 3, 4 and 5 are successful in the cases of correlations and singularity, without imposing structural assumptions on the dataset.

Chapter 3

Simplicial Distances

The research presented in this chapter forms part of a publication I have co-authored [85] entitled **Simplicial and Minimal-Variance Distances in Multivariate Data Analysis**, published in *Journal of Statistical Theory and Practice*, available at https://doi.org/10.1007/s42519-021-00227-7 .

The differences between the published manuscript and the contents of this chapter are:

- This chapter does not include research on minimal-variance distances (which is given in Chapter 4);
- The manuscript compares the simplicial distances to the minimal-variance distances in their efficiency and through numerical examples. These comparisons will be included in Chapter 4 of this thesis, after introducing the minimal-variance distances;
- This chapter gives details of the distribution of the simplicial distances (Section 3.4), which are not included in the manuscript;
- Additional empirical examples relating solely to the simplicial distances are given in this chapter.

The aims of the research presented in this chapter are:

- To produce a distance measure which performs similarly to the Mahalanobis distance in multivariate data, but without the issues of degeneracy faced by the Mahalanobis distance, as detailed in the literature review (see Section 2.2.2);

- To produce a non-model based method of measuring proximity. The Mahalanobis distance relies on an estimator of the covariance matrix (which often imposes assumptions on the data, see Section 2.5 of the literature review). The simplicial distances are based on simplices formed by the data, so do not require such estimators or assumptions;
- To highlight how the simplicial distances can be used to form an ℓ_1 version of the Mahalanobis distance, as ℓ_1 distances are shown to be more useful in some circumstances in high dimensional data analysis (see Section 2.4.2);
- To illustrate the spectrum of metrics created by the simplicial distances, and to give recommendations for parameter choices and application uses.

3.1 Introduction

Pronzato et al. [192] introduced a family of distances called the simplicial distances. These distances are parameterized by a value k, an integer which takes value between 1 and d, where d is the dimension of the dataset being measured. The distances can be used to find the proximity of a point $x \in \mathbb{R}^d$ to a set of points $X = \{x_1, x_2, \ldots, x_N\} \in \mathbb{R}^{d \times N}$. The distance is calculated as follows: compute the volumes of all possible k-dimensional simplices formed by the point x and all points from the set X, and raise the average volume to a user-defined exponent $\delta > 0$ to give the k-simplicial distance between x and X. Figure 3.1 gives a visualization of three-dimensional simplices formed by points within three-dimensional space.

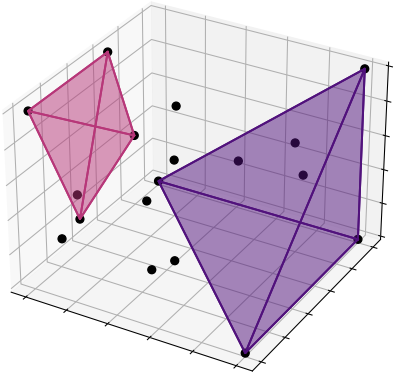


Figure 3.1: Examples of three-dimensional simplices in three-dimensional space.

For general values of $\delta > 0$, when $k = 1$ the simplicial distance between the point x and the set X is proportional to the ℓ_δ-distances in $\mathbb{R}^d$ between x and the mean of the set of points X. Therefore, when $\delta = 2$, using $k = 1$ gives distances proportional to the squared Euclidean distances. It can be shown that using $k = d$ with $\delta = 2$ gives distances proportional to the squared Mahalanobis distance (if the inverse of the covariance matrix exists) [192].

Other choices of the parameters k and δ will be discussed in detail in Section 3.3 and can be used to alter distances to make them more appropriate for specific purposes. Throughout this chapter, the terms 'k-simplicial distances' and 'simplicial distances' are used interchangeably, but with the former often referring to the distance using a specific value of k, and the latter referring to the distances more generally.

This rest of this chapter is structured as follows. Section 3.2 describes the methods of calculating the simplicial distances. Section 3.3 discusses the impact of using different parameters for the simplicial distance; namely the degree parameter k and the exponent parameter δ. The distribution of the simplicial distance with $\delta = 2$ is considered in Section 3.4. Finally, some applications of the distance measure are given in Section 3.5, including outlier detection, clustering and data whitening.

3.2 Constructing the simplicial distances

The simplicial distance was initially proposed by Pronzato et al. [192]. The authors describe two methods of calculating the distances, both of which will be outlined (with some modifications) in this section. In particular, Section 3.2.1 describes the formal construction of the distance through volumes of simplices, as previously described. A normalization constant has been introduced which was not present in the aforementioned paper. Section 3.2.2 outlines an alternative, faster method for construction when $\delta = 2$, using elementary symmetric functions and polynomial methods. Section 3.2.3 gives a novel approach to the calculation of simplicial distances using sampling methods, with the purpose of improving computation time for the distance when found directly using simplex volumes.

3.2.1 Construction through simplex volumes

Let $X = \{x_1, \ldots, x_N\} \in \mathbb{R}^{d \times N}$ be a set of N points in d dimensions, with no assumptions on how this set of points has been generated. The sample mean and covariance matrix associated with the set X are defined as follows:

$$\mu = \frac{1}{N}\sum_{j=1}^{N} x_j, \quad \Sigma = \frac{1}{N}\sum_{j=1}^{N}(x_j - \mu)(x_j - \mu)^\top, \tag{3.1}$$

where the biased covariance matrix is defined here to simplify some later calculations.

The k-simplicial distance between a point $x \in \mathbb{R}^d$ and the set X is defined as the average volume of all possible k-dimensional simplices formed by x and any k points in X, raised to a given power $\delta > 0$. Let $r \leq d$ be the intrinsic dimension of the data set X, which is the rank of X when X is considered as a $d \times N$ matrix. The volumes of all k-dimensional simplices are zero for $k > r$, and so it makes no sense to use $k > r$.

Let $\mathscr{V}_k(x, z_1, \ldots, z_k)$ be the volume of a k-dimensional simplex with vertices $x \in \mathbb{R}^d$ and $z_1, \ldots, z_k \in \mathbb{R}^d$. Following Theorem 4 of [192], this volume can be computed by

$$\mathscr{V}_k(x, z_1, \ldots, z_k) = \frac{1}{k!}\left|\det(Z^\top Z)\right|^{1/2},$$

where $\det(A)$ is the determinant of the matrix A and $|a|$ is the absolute value of the scalar a. Z is the $d \times k$ matrix with columns $[(z_1 - x)\ (z_2 - x)\ \ldots\ (z_k - x)]$. Let

$$J = \left\{(j_1, \ldots, j_k) \in \{1, \ldots, N\}^k \mid j_1 < \ldots < j_k\right\} \tag{3.2}$$

be the set of all ordered k-combinations of the indices in $\{1, \ldots, N\}$. Define

$$P_{k,\delta}(x, X) = \frac{1}{\binom{N}{k}} \sum_{(j_1, \ldots, j_k) \in J} \mathscr{V}_k^{\delta}(x, x_{j_1}, \ldots, x_{j_k}), \tag{3.3}$$

which is the average volume of all k-dimensional simplices created by the point x and points in X, raised to the power of a user-defined scalar $\delta > 0$. For given $\delta > 0$ and $1 \leq k \leq r$, the centre of the set X (that is, the k-simplicial multidimensional median [179]) is defined as

$$\overline{\mu}_{k,\delta} = \arg\min_{x \in \mathbb{R}^d} P_{k,\delta}(x, X),$$

which may not be uniquely defined [276]. The k-simplicial outlyingness function is then defined by

$$O_{k,\delta}(x, X) = \frac{P_{k,\delta}(x, X)}{P_{k,\delta}(\overline{\mu}_{k,\delta}, X)} - 1. \tag{3.4}$$

The function (3.4) is non-negative, has value 0 at the centre of the sample and is unitless; which are the required properties that an outlyingness function must possess [250]. For any $\delta > 0$, the k-simplicial distance (here to the power of δ) from a point x to the dataset X is

$$\rho_{k,\delta}^{\delta}(x, X) = c_{k,\delta} O_{k,\delta}(x, X) = c_{k,\delta}\left(\frac{P_{k,\delta}(x, X)}{P_{k,\delta}(\overline{\mu}_{k,\delta}, X)} - 1\right), \tag{3.5}$$

where the constant $c_{k,\delta}$ is chosen so that

$$\frac{1}{N}\sum_{j=1}^{N} \rho_{k,\delta}^{\delta}(x_j, X) = 1\,. \tag{3.6}$$

The normalization in Equation (3.6) is introduced to ensure consistency of the simplicial distances for different values of k. It is shown in Section 3.2.2 that for $\delta = 2$ and all $k \leq r$, the normalization constant takes value $c_{k,2} = 1/k$. In the cases where $\delta \neq 2$, the constants $c_{k,\delta}$ are found numerically from Equation (3.6).

For $\delta = 2$ and any eligible k, the centre of the set X is given by the sample mean $\overline{\mu}_{k,\delta} = \mu$ [192, Theorem 5]. Then, similarly to Section 3.1 of [192],

$$\frac{1}{N}\sum_{j=1}^{N} P_{k,2}(x_j, X) = (k+1) P_{k,2}(\mu, X)\,. \tag{3.7}$$

The squared k-simplicial distance (of order $\delta = 2$) from the point x to the dataset X is then defined as

$$\rho_{k,2}^{2}(x, X) = \frac{1}{k} O_{k,2}(x, X) = \frac{1}{k}\left(\frac{P_{k,2}(x, X)}{P_{k,2}(\mu, X)} - 1\right). \tag{3.8}$$

The difference between Equation (3.8) and the corresponding definition in [192, Equation 17] is the introduction of the normalizing constant $1/k$, which provides consistency of the distances for different k in the sense that Equation (3.6) holds for $\delta = 2$ and all $k = 1, 2 \ldots, r$. The equality in Equation (3.6) with $\delta = 2$ directly follows from Equation (3.7).

Evaluation of the distances given by Equation (3.8) can be computationally time-consuming when performed 'directly' by the empirical calculation of the volumes of all $\binom{N}{k}$ simplices. In Section 3.2.2, an alternative polynomial method for use with $\delta = 2$ is given, which is much faster. Section 3.2.3 then outlines a sampling approach to reduce computation time when calculating the distance using the volumes of the simplices directly.

3.2.2 Construction through elementary symmetric functions

Let $\Lambda = \{\lambda_1, \ldots, \lambda_d\}$ be the set of eigenvalues of the matrix Σ defined in (3.1). The elementary symmetric function of degree $k \leq d$ associated with the set Λ is given by

$$e_k(\Lambda) = \sum_{1 \leq i_1 < i_2 < \ldots < i_k \leq d} \lambda_{i_1} \ldots \lambda_{i_k},$$

with $e_0(\Lambda) = 1$. If $k > r = \text{rank}(X)$ then $e_k(\Lambda) = 0$ and the k-simplicial distance is always 0. For $k \leq r$, define the function

$$q_k(\Sigma) = \sum_{i=0}^{k-1} (-1)^i e_{k-i-1}(\Lambda)\Sigma^i \tag{3.9}$$

and the associated matrix

$$S_k = \frac{q_k(\Sigma)}{e_k(\Lambda)}.$$

As follows from Section 3.2 of [192], for any $k \leq r$, the distance defined in Equation (3.8) can be written as:

$$\rho^2_{k,2}(x, X) = \frac{1}{k}(x-\mu)^\top S_k (x-\mu). \tag{3.10}$$

Note that the $d \times d$ matrices S_k, $k = 1, \ldots, r$, are polynomials in the covariance matrix Σ.

Since $S_1 = I/\text{trace}(\Sigma)$, where I is the $d \times d$ identity matrix, for $k = 1$ the squared simplicial distance in Equation (3.10) is equal to the squared Euclidean distance over the trace of the covariance matrix Σ of the data X:

$$\rho^2_{1,2}(x, X) = (x-\mu)^\top S_1 (x-\mu) = (x-\mu)^\top \frac{q_1(\Sigma)}{e_1(\Lambda)}(x-\mu) = \frac{(x-\mu)^\top (x-\mu)}{\text{trace}(\Sigma)}.$$

Section 3.1 of [192] shows that when $k = d$ and Σ is invertible, $S_d = \Sigma^{-1}$. That is, the squared simplicial distance (3.10) is equal to the squared Mahalanobis distance over d:

$$\rho^2_{d,2}(x, X) = \frac{1}{d}(x-\mu)^\top S_d (x-\mu) = \frac{1}{d}(x-\mu)^\top \Sigma^{-1} (x-\mu).$$

The following theorem compares the variance of the squared Euclidean distance, Mahalanobis distance and simplicial distance with $k = 2$ and $\delta = 2$.

Theorem 1. *Assume $X = \{x_1, \ldots, x_N\} \sim \mathcal{N}_d(\mu, \Sigma)$. That is, let X be a set of N normally distributed d-dimensional points with sample mean μ and sample covariance matrix Σ, as*

defined in (3.1). *Let* $\Lambda = \{\lambda_1, \ldots, \lambda_d\}$ *be the set of eigenvalues of the matrix* Σ. *Assume* $\text{rank}(X) = r \leq d$ *and let* $x \in X$. *Then*

$$Var\left(\rho^2_{r,2}(x, X)\right) \leq Var\left(\rho^2_{2,2}(x, X)\right) < Var\left(\rho^2_{1,2}(x, X)\right),$$

where $\rho^2_{k,2}(x, X)$ *is the squared k-simplicial distance between the point x and set X as defined in Equation* (3.5) *with* $\delta = 2$.

Proof. The simplicial distance between a point x and set X with $k = 2$ and $\delta = 2$ can be written as

$$\rho^2_{2,2}(x, X) = (x-\mu)^\top \frac{S_k}{k}(x-\mu) = (x-\mu)^\top \frac{S_2}{2}(x-\mu) = (x-\mu)^\top \frac{q_2(\Sigma)}{2e_2(\Lambda)}(x-\mu)$$

with $q_2(\Sigma) = e_1(\Lambda)I - \Sigma$, from Equation (3.9). From (A.1) in Appendix A.2, the variance of the simplicial distance with $k = 2$ and $\delta = 2$ is given by

$$\text{Var}\left(\rho^2_{2,2}(x, X)\right) = 2\text{trace}\left(\left[\frac{S_2}{2}\Sigma\right]^2\right) = \frac{\text{trace}\left(\Sigma^2(e_1(\Lambda)I - \Sigma)^2\right)}{2e_2(\Lambda)^2}, \tag{3.11}$$

Let $\eta_i = \sum_{j\neq i}\lambda_j = \sum_{j=1}^r \lambda_j - \lambda_i = e_1(\Lambda) - \lambda_i$. Consider the second-order elementary symmetric polynomial:

$$e_2(\Lambda) = \sum_{i<j}\lambda_i\lambda_j = \frac{1}{2}\sum_{i\neq j}\lambda_i\lambda_j = \frac{1}{2}\left(\sum_{i=1}^r\sum_{j=1}^r \lambda_i\lambda_j - \sum_{i=1}^r \lambda_i^2\right) = \frac{1}{2}\sum_{j=1}^r \lambda_j\eta_j\,.$$

Then Equation (3.11) can be re-written as:

$$\text{Var}\left(\rho^2_{2,2}(x, X)\right) = \frac{\text{trace}\left(\Sigma^2(e_1(\Lambda)I - \Sigma)^2\right)}{2e_2(\Lambda)^2} = \frac{\sum_{j=1}^d \lambda_j^2(e_1(\Lambda) - \lambda_j)^2}{2(\frac{1}{2}\sum_{j=1}^d \lambda_j\eta_j)^2} = \frac{2\sum_{j=1}^r \lambda_j^2\eta_j^2}{(\sum_{j=1}^r \lambda_j\eta_j)^2}. \tag{3.12}$$

Using (A.2) from Appendix A.3, the variance of the distance with $k = 1$ and $\delta = 2$ is:

$$\text{Var}\left(\rho^2_{1,2}(x, X)\right) = \frac{2\sum_{j=1}^r \lambda_j^2}{(\sum_{j=1}^r \lambda_j)^2}. \tag{3.13}$$

Consider the denominator in Equation (3.13). By the Cauchy-Schwartz inequality,

$$\left(\sum_{j=1}^r \lambda_j\right)^2 = \left(\sum_{j=1}^r 1\cdot\lambda_j\right)^2 \leq \sum_{j=1}^r 1^2 \sum_{j=1}^r \lambda_j^2 = r\sum_{j=1}^r \lambda_j^2,$$

and so it follows that

$$\text{Var}\left(\rho^2_{1,2}(x, X)\right) = \frac{2\sum_{j=1}^r \lambda_j^2}{(\sum_{j=1}^r \lambda_j)^2} \geq \frac{2\sum_{j=1}^r \lambda_j^2}{r\sum_{j=1}^r \lambda_j^2} = \frac{2}{r} = \text{Var}\left(\rho^2_{r,2}(x, X)\right).$$

Again using the Cauchy-Schwartz inequality, the denominator in Equation (3.12) has the inequality

$$\left(\sum_{j=1}^{r} \lambda_j \eta_j\right)^2 \leq r \sum_{j=1}^{r} \lambda_j^2 \eta_j^2,$$

and therefore

$$\mathrm{Var}\left(\rho_{2,2}^2(x,X)\right) = \frac{2\sum_{j=1}^{r} \lambda_j^2 \eta_j^2}{(\sum_{j=1}^{r} \lambda_j \eta_j)^2} \geq \frac{2\sum_{j=1}^{r} \lambda_j^2 \eta_j^2}{r\sum_{j=1}^{r} \lambda_j^2 \eta_j^2} = \frac{2}{r} = \mathrm{Var}\left(\rho_{r,2}^2(x,X)\right).$$

It remains to show that

$$\mathrm{Var}\left(\rho_{1,2}^2(x,X)\right) = \frac{2\sum_{j=1}^{r} \lambda_j^2}{\left(\sum_{j=1}^{r} \lambda_j\right)^2} \geq \frac{2\sum_{j=1}^{r} \lambda_j^2 \eta_j^2}{(\sum_{j=1}^{r} \lambda_j \eta_j)^2} = \mathrm{Var}\left(\rho_{2,2}^2(x,X)\right). \tag{3.14}$$

The validity of the inequality in (3.14) does not depend on the change $\lambda_i \to c\lambda_i$ for all i and for any constant $c > 0$. Therefore, it is viable to choose $\lambda_1, \lambda_2, \ldots, \lambda_r$ such that $\sum_{i=1}^{r} \lambda_i = 1$. Equation (3.14) can then be expressed as moments of a random variable ξ concentrated on $[0,1]$ having values λ_i with probabilities λ_i.

Let $\tau_j = E\left[\xi^j\right]$. Using this notation, consider the following properties:

$$\begin{aligned}
&\sum_{j=1}^{r} \lambda_j = E\left[\xi^0\right] = 1,\\
&\sum_{j=1}^{r} \lambda_j^2 = E\left[\xi\right] = \tau_1,\\
&\sum_{j=1}^{r} \lambda_j \eta_j = \sum_{j=1}^{r} \lambda_j (1-\lambda_j) = 1 - E\left[\xi\right] = 1 - \tau_1,\\
&\sum_{j=1}^{r} \lambda_j^2 \eta_j^2 = \sum_{j=1}^{r} \lambda_j^2 (1-\lambda_j)^2 = E\left[\xi\right] - 2E\left[\xi^2\right] + E\left[\xi^3\right]\\
&\qquad\qquad = \tau_1 - 2\tau_2 + \tau_3.
\end{aligned}$$

Using these properties the inequality in (3.14) has the form

$$\tau_1 \geq \frac{\tau_1 - 2\tau_2 + \tau_3}{(1-\tau_1)^2}.$$

Rearranging gives $\tau_1^3 + 2\tau_2 - 2\tau_1^2 - \tau_3 \geq 0$, which is true for all probability measures on $[0, 1]$. □

Theorem 1 helps to prove the intuition that as k increases, the variance of the simplicial distances between points within the set X and the set X itself decreases. This intuition will be illustrated and discussed further in Section 3.3.1.

3.2.3 Computation using sampling methods

Section 3.2.2 introduced a fast polynomial method to produce the simplicial distances between x and X when using the parameter $\delta = 2$. When using $\delta \neq 2$, the distances must be computed using the methods outlined in Section 3.2.1: by averaging the volumes of all $\binom{N}{k}$ simplices formed with x and $X \in \mathbb{R}^{d \times N}$. This can be extremely computationally intensive, particularly for large d and large N.

To circumvent this problem and reduce computation time, the volumes of a subset of the simplices can be found, rather than all possible simplices. The size of the sample of simplices depends on the user's wish for precision versus time improvement. This size does not have to be large to achieve practically accurate approximations, which will be demonstrated in the examples that follow, where less than 0.05% of all possible simplices are used when use $k = 3$ is considered, and less than 0.0004% when using $k = 4$.

Let J be the set of all ordered k-combinations of the indices in $\{1, \ldots, N\}$, as defined in Equation (3.2). To compute the simplicial distances, the values of $P_{k,\delta}(x, X)$ defined in Equation (3.3) must be computed. The method of approximating these values using sampling is as follows.

For any sampling percentage $\gamma \in [0, 1]$, create $J^{(\gamma)}$, a subset of J of size $|J^{(\gamma)}| = \left\lceil \gamma \times \binom{N}{k} \right\rceil$. Approximate (3.3) by finding the volumes of the simplices indexed by the indices in $J^{(\gamma)}$ and x:

$$P_{k,\delta,\gamma}(x, X) = \frac{1}{|J^{(\gamma)}|} \sum_{(j_1,\ldots,j_k) \in J^{(\gamma)}} \mathscr{V}_k^{\delta}(x, x_{j_1}, \ldots, x_{j_k}).$$

A simple but efficient way of constructing $J^{(\gamma)}$ consists of taking random samples of size k without replacement from the set $\{1, 2, \ldots, N\}$, see [37]. This reduces computation time dramatically, and in examples that follow it is clear that this method of sampling is effective in producing results extremely close to those of the 'full' distance measure, in which the volumes of all available simplices are calculated.

Three numerical examples are given to illustrate the sampling method and its performance. In each of the examples, $N = 500$ points are generated from a d-dimensional multivariate normal distribution, with zero mean and diagonal covariance matrix. Table 3.1 gives the values of d and the eigenvalues of the covariance matrix used to generate the datasets. The sample covariance matrix is used when computing the distances, and so

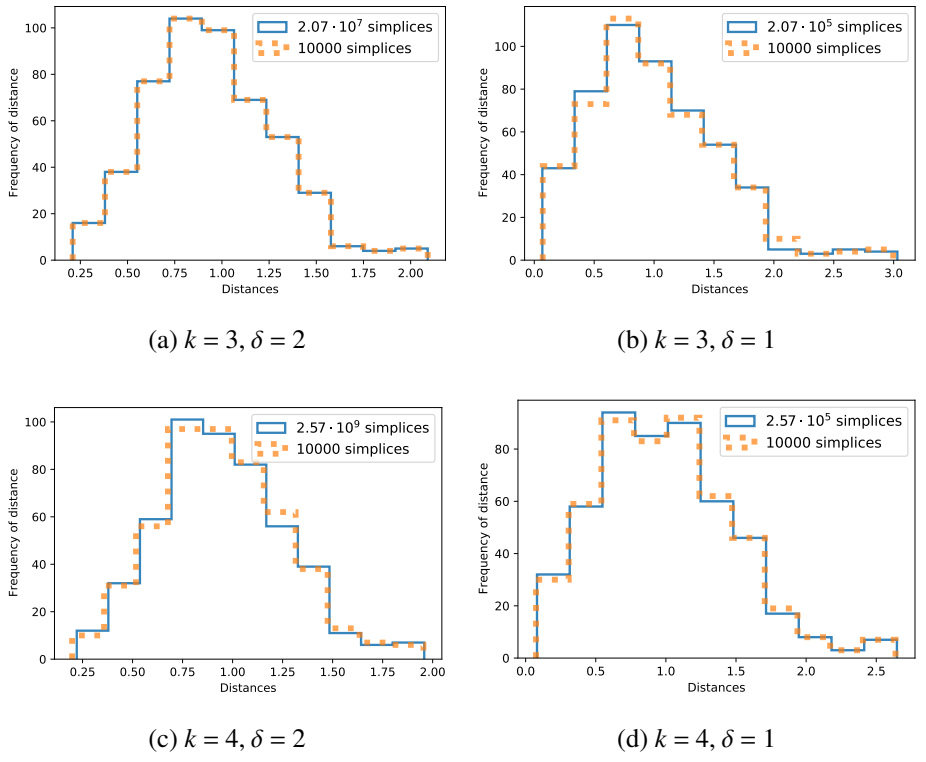
the true eigenvalues of the matrix will differ slightly from those given in Table 3.1. The parameters $k = 3$ and $k = 4$ are used in the examples that follow, as it will later be shown that using relatively low values of k often produces results not dissimilar to higher degree polynomials (see Section 3.3.1).

Λ	d	Eigenvalues
Λ_A	10	$[100, 4, 3, 2, 1] + [0.0001] * 4 + [0]$
Λ_B	50	$[100, 10] + [1] * 5 + [0.0001] * 33 + [0] * 10$
Λ_C	50	$[100, 100] + [1] * 10 + [0.00001] * 10 + [0] * 28$

Table 3.1: Details of datasets generated to be used in sampling examples. For each dataset, $N = 500$ points are generated from the d-dimensional Gaussian distribution with zero mean and diagonal covariance matrix, with the eigenvalues in the table on the diagonal (written in Python notation).

For each of the datasets, the distances from all points in the dataset X to the sample mean of the dataset μ are calculated, using the full sample of simplices (where possible) and then a smaller sample using 10,000 simplices. For $\delta = 2$, the 'full' distance is found using the fast polynomial method described in Section 3.2.2. For $\delta = 1$, however, the 'full' distance requires the computation of $\binom{500}{k}$ simplices, which gets large very quicky. Instead, a subsample of 10,000 simplices is compared to a larger sample of simplices (1% of the total amount of simplices in the $k = 3$ case, 0.01% in the $k = 4$ case). The effect that the different sample sizes have on the distribution of distances can be seen by investigating moments and histograms of distances.

Example 1: $\Lambda = \Lambda_A$. Histograms of the simplicial distances with and without using sampling are given in Figure 3.2 for $\delta = 2$ and $\delta = 1$. These histograms show that the distribution of distances produced using a sample of simplices is extremely similar to the distribution produced using all possible simplices (or a much larger sample, when considering $\delta = 1$ here). This indicates that sampling can be used to effectively reduce computation time without significantly changing the output of the distance measure. Tables 3.2a and 3.2b also demonstrate this through summary statistics for $k = 3$ and $k = 4$ respectively.



(a) $k = 3, \delta = 2$

(b) $k = 3, \delta = 1$

(c) $k = 4, \delta = 2$

(d) $k = 4, \delta = 1$

Figure 3.2: Histograms to compare the distribution of the simplicial distances from all points to the mean for eigenvalues Λ_A with different parameters k and δ for different sampling amounts. The blue solid line is the full (or larger) sample of simplices, the orange dotted line is the smaller sample of simplices.

	$\delta = 2$		$\delta = 1$	
# simplices	$2.07 \cdot 10^7$	10^4	$2.07 \cdot 10^5$	10^4
Mean	1.00	1.00	1.00	1.00
Variance	0.11	0.11	0.27	0.28
Skewness	0.49	0.49	0.81	0.81
Kurtosis	0.31	0.31	0.85	0.87

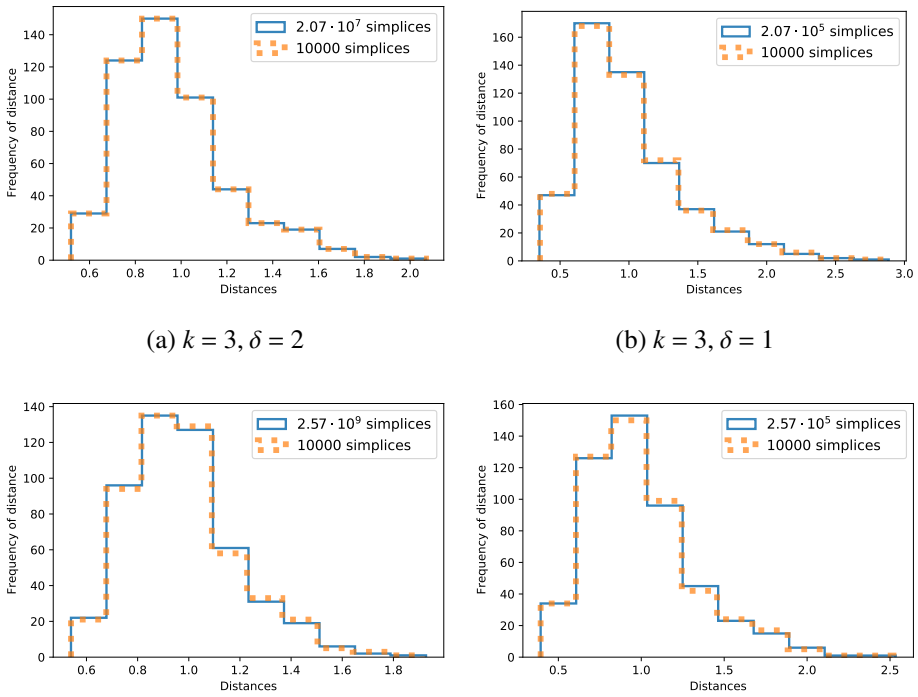
(a) $k = 3$

	$\delta = 2$		$\delta = 1$	
# simplices	$2.57 \cdot 10^9$	10^4	$2.57 \cdot 10^5$	10^4
Mean	1.00	1.00	1.00	1.00
Variance	0.10	0.10	0.23	0.23
Skewness	0.38	0.40	0.61	0.60
Kurtosis	0.10	0.12	0.34	0.34

(b) $k = 4$

Table 3.2: Summary statistics of the distances when (a) $k = 3$, (b) $k = 4$ with eigenvalues Λ_A. Table headers indicate the value of δ used. The second row gives the number of simplices sampled.

Example 2: $\Lambda = \Lambda_B$**.** Figure 3.3 and Tables 3.3a and 3.3b show again that using a low number of simplices produces distances that are mostly the same as the full distance measure. This example illustrates that the sampling method is effective even in cases with a lot of small and zero eigenvalues.



(a) $k = 3, \delta = 2$ (b) $k = 3, \delta = 1$

(c) $k = 4, \delta = 2$ (d) $k = 4, \delta = 1$

Figure 3.3: Histograms to compare the distribution of the simplicial distances from all points to the mean for eigenvalues Λ_B with different parameters k and δ for different sampling amounts. The blue solid line is the full (or larger) sample of simplices, the orange dotted line is the smaller subsample of simplices.

	$\delta = 2$		$\delta = 1$	
# simplices	$2.07 \cdot 10^7$	10^4	$2.07 \cdot 10^5$	10^4
Mean	1.00	1.00	1.00	1.00
Variance	0.06	0.06	0.16	0.16
Skewness	1.08	1.08	1.29	1.29
Kurtosis	1.42	1.42	2.00	1.97

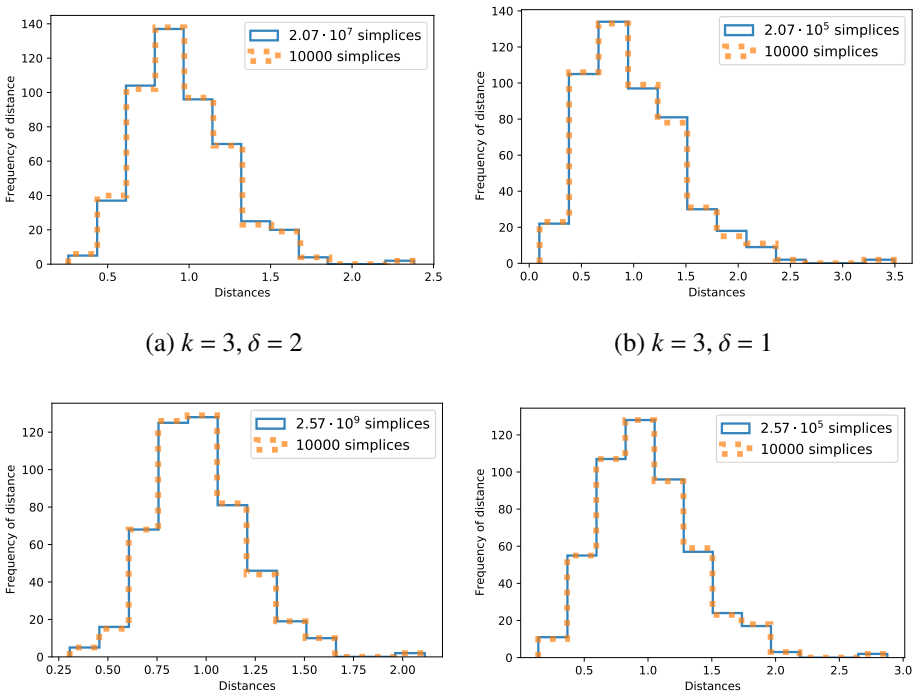
(a) $k = 3$

	$\delta = 2$		$\delta = 1$	
# simplices	$2.57 \cdot 10^9$	10^4	$2.57 \cdot 10^5$	10^4
Mean	1.00	1.00	1.00	1.00
Variance	0.04	0.04	0.11	0.11
Skewness	0.91	0.91	1.09	1.08
Kurtosis	1.20	1.16	1.58	1.52

(b) $k = 4$

Table 3.3: Summary statistics of the distances when (a) $k = 3$, (b) $k = 4$ with eigenvalues Λ_B. Table headers indicate the value of δ used. The second row gives the number of simplices sampled.

Example 3: $\Lambda = \Lambda_C$**.** The third dataset considered here is 50-dimensional with many small and zero eigenvalues. Figure 3.4 and Tables 3.4a and 3.4b show that small and zero eigenvalues do not affect the success of the sampling method, as the distribution of the distances measured are very close.



(a) $k = 3, \delta = 2$

(b) $k = 3, \delta = 1$

(c) $k = 4, \delta = 2$

(d) $k = 4, \delta = 1$

Figure 3.4: Histograms to compare the distribution of the simplicial distances from all points to the mean for eigenvalues Λ_C with different parameters k and δ for different sampling amounts. The blue solid line is the full (or larger) sample of simplices, the orange dotted line is the smaller sample of simplices.

	$\delta = 2$		$\delta = 1$	
# simplices	$2.07 \cdot 10^7$	10^4	$2.07 \cdot 10^5$	10^4
Mean	1.00	1.00	1.00	1.00
Variance	0.08	0.08	0.22	0.22
Skewness	0.81	0.81	1.10	1.11
Kurtosis	1.64	1.64	2.43	2.47

(a) $k = 3$

	$\delta = 2$		$\delta = 1$	
# simplices	$2.57 \cdot 10^9$	10^4	$2.57 \cdot 10^5$	10^4
Mean	1.00	1.00	1.00	1.00
Variance	0.06	0.06	0.14	0.14
Skewness	0.62	0.62	0.86	0.86
Kurtosis	1.36	1.34	1.82	1.84

(b) $k = 4$

Table 3.4: Summary statistics of the distances when (a) $k = 3$, (b) $k = 4$ with eigenvalues Λ_C. Table headers indicate the value of δ used. The second row gives the number of simplices sampled.

Overall, it is clear that sampling is an effective way to drastically reduce computation time while maintaining very similar results to the full simplicial distances. This means that using the distances with $\delta \neq 2$ is much more accessible than it otherwise would be. Section 3.3.2 considers the choice of the parameter δ in more depth, including a comparison of computation times.

3.3 Parameter selection for the simplicial distances

The simplicial distance measure is characterised by two parameters. The parameter k dictates the dimension of the simplices used to calculate the simplicial distances (or the degree of the polynomial used, if using the method detailed in Section 3.2.2). Different values of k can affect the performance of the distances drastically, as will be shown in Section 3.3.1, and can provide access to the ℓ_δ distance and Mahalanobis distance (if it exists).

The exponent parameter δ also changes the behaviour of the distance: in Section 3.3.2, the effect of the parameter δ is compared to the commonly used ℓ_δ distance measures. The time taken to compute the distance is affected by the choice of k and δ, which may play an influential part in parameter selection. The time taken to compute the distance will therefore also be explored in Section 3.3.2.

3.3.1 Choosing k in the simplicial distances

The choice of the parameter k can greatly influence the behaviour of the simplicial distances. This section considers how different choices of k affect the distance through experimental results.

For the three datasets detailed in Table 3.1, the k-simplicial distances between all points $x \in X$ to the dataset X itself are found, for both $\delta = 2$ and $\delta = 1$. Values of $k \leq r$ are considered, where r is the rank of the dataset being considered. Figure 3.5 shows the plots of the empirical cumulative distribution function (CDF) for each combination of parameters, for the three datasets. Note that for distances using $\delta = 1$, sampling is used to find the distance, using the method described in Section 3.2.3. Examples using $\delta = 2$ also consider the squared Moore-Penrose (MP) Mahalanobis distance over r, which is equal

to the k-simplicial distance with $k = r$.

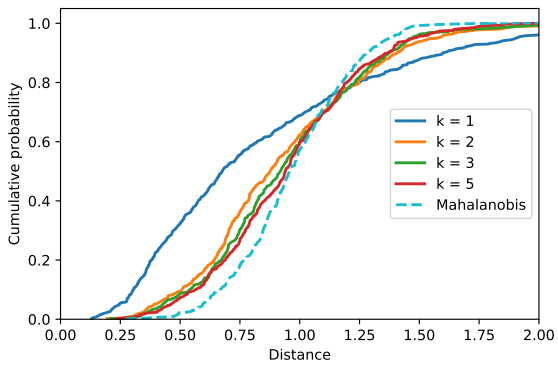
Example 1: $\Lambda = \Lambda_A$**.** The CDFs for the distances with $\delta = 2$ and $\delta = 1$ are given in Figure 3.5a and Figure 3.5b, respectively. These figures indicate that the squared Euclidean distance (proportional to the simplicial distance with $k = 1$, $\delta = 2$) produces a large range of distances with high variance, when compared to the distances produced with other values of k. In the $\delta = 2$ case, low values of k (compared to the rank $r = 9$) begin to converge away from the squared Euclidean distance, and towards the squared MP Mahalanobis distance quickly. Figure 3.5b shows a similar pattern for the distance with $\delta = 1$: the variance of the distances decrease as the value of k increases.

Example 2: $\Lambda = \Lambda_B$**.** The CDFs for the distances with both $\delta = 2$ and $\delta = 1$ are given in Figure 3.5c and Figure 3.5d. For relatively low values of k (compared to the rank $r = 40$), such as $k = 10$, the distances converge towards those produced when $k = r$, i.e. the MP Mahalanobis distance in the case $\delta = 2$. A similar profile is observed for $\delta = 1$.

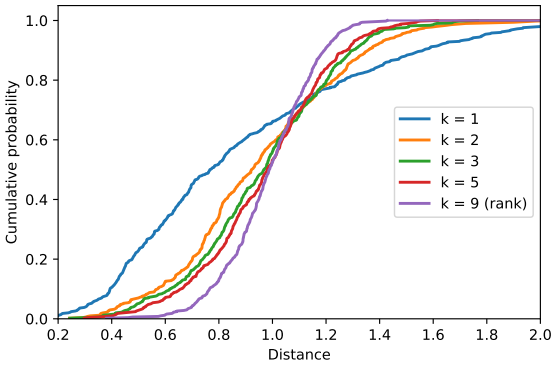
Example 3: $\Lambda = \Lambda_C$**.** The CDFs for the distances with $\delta = 2$ and $\delta = 1$ are given in Figure 3.5e and Figure 3.5f. Again, for relatively low values of k (compared to rank $r = 22$) the CDFs of the simplicial distances converge towards the CDF of the distance where $k = r$. Note that in Figure 3.5f, for $\delta = 1$, $k = 10$, the CDF of the distances lies directly underneath the CDF of the distances using $k = 22$, as the distances produced are so similar.

Figure 3.5 demonstrates that the simplicial distances transition from the squared Euclidean distance to the squared Mahalanobis distance for $\delta = 2$ as k increases, up to some scaling. A similar monotonic behaviour is shown for $\delta = 1$. The eigenvalues of the sample covariance matrix have an effect on what an appropriate choice of k may be. It is important to ensure the most influential dimensions (that is, those with the largest eigenvalues) are all considered, by taking k larger than the number of large eigenvalues.

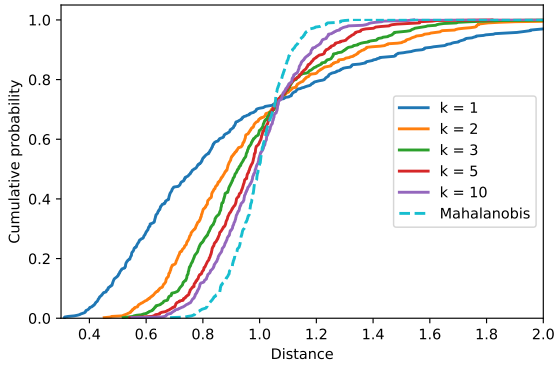
For example, consider Figure 3.5e. The two large eigenvalues in Λ_C $[100, 100]$ result in the distances with $k = 2$ behaving similarly to the distances with $k = 1$, particularly in the $\delta = 2$ case. In Figure 3.5a, which considers Λ_A, the CDF produced using the distance with $k = 2$ is very different to the CDF where $k = 1$, as there is only one large eigenvalue.



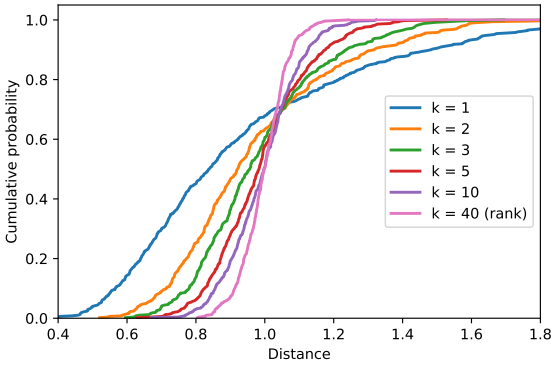
(a) Λ_A, $\delta = 2$



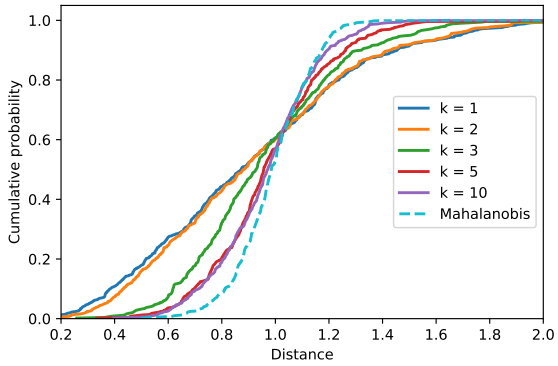
(b) Λ_A, $\delta = 1$



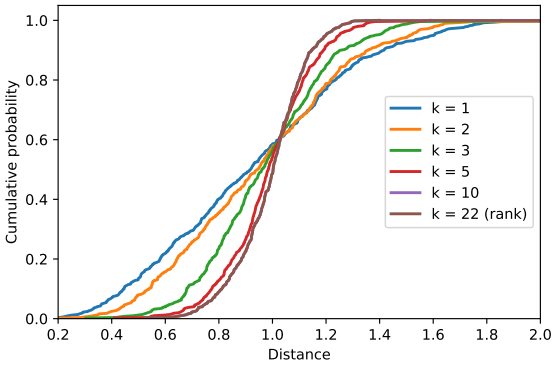
(c) Λ_B, $\delta = 2$



(d) Λ_B, $\delta = 1$



(e) Λ_C, $\delta = 2$



(f) Λ_C, $\delta = 1$

Figure 3.5: CDFs of the simplicial distances using different values of k for the datasets given in Table 3.1 using $\delta = 2$ and $\delta = 1$.

In general, it is recommended to use a value of k that is larger than the number of 'large' eigenvalues the covariance matrix Σ has. This is easier to see when there is a clear elbow or 'drop-off' in the value of the eigenvalues. Otherwise it can be appropriate to find the simplicial distances with several values of k and measure the best value according to some metric appropriate to the task, much like methods used when performing K-means clustering and other parameter-dependent tasks.

Not much performance gain is made by choosing a value of k that also encompasses the smaller eigenvalues. As an example of this, see Figure 3.5c, where there are two 'large' eigenvalues, 5 'medium' eigenvalues, 33 'small' eigenvalues and 10 zero eigenvalues. Taking $k = 10$ does not give a huge improvement in performance compared to $k = 5$, where performance is measured here by the minimizing of variance, but using $k = 10$ is more computationally expensive.

3.3.2 Choosing δ in the simplicial distances

This section considers the choice of the exponent parameter δ. Some intuition behind the different choices of δ is given by drawing comparisons to the well-known ℓ_δ distance measures, defined below. Comparisons between the simplicial distances with different values of δ will be made by measuring the relative contrast of the distances, as considered by Aggarwal et al. in [5]. The subsection will conclude with some timing comparisons between the distance using the methods outlined in Section 3.2.2 for $\delta = 2$, and the sampling methods outlined in Section 3.2.3 for other values of δ.

The choice of the parameter δ has a large influence on the behaviour of the simplicial distances. When $k = 1$, the simplicial distance between a point $x \in \mathbb{R}^d$ and the set $X \in \mathbb{R}^{d\times N}$ is proportional to the ℓ_δ distance between x and μ, the sample mean of the dataset X [192]. The ℓ_δ distances are defined as

$$\ell_\delta(x, \mu) = \left(\sum_{i=1}^{d} |x_i - \mu_i|^\delta \right)^{1/\delta},$$

and were previously introduced in Section 2.2.1 of the literature review.

The ℓ_2 distance is also known as the Euclidean distance, and is perhaps the most commonly used distance measure in data analysis. It is particularly useful for applications

such as outlier detection: a greater value is assigned to outliers due to the square exponent in the ℓ_2 measure, putting more emphasis on outlying points and making them more detectable than distances with $\delta < 2$. However, this does mean that if outliers are not removed from a dataset, the ℓ_2 distance is much more likely to be influenced by them, meaning it is not always a very robust distance. It also reduces the importance of points near each other, as the small value of the distance becomes smaller when squared.

The ℓ_1 distance is also commonly known as the Manhattan distance. It is the sum of absolute differences between each component of the points. On the contrary to the ℓ_2 distance, the ℓ_1 distance is more resistant to outliers in the sense that it does not put more emphasis on large values, and so will not be as easily influenced by outliers. Clearly, the choice of distance measure is very much dependent on the application, and neither of these distance measures will consistently outperform the other, as is true for all distance measures.

Relative constrast of the distances

As discussed in Section 2.4.2 of the literature review, the difference between the minimum and the maximum pairwise distances between any two points in a dataset tends to zero as the dimension d of the data increases [33]. This is shown to be the case for a variety of distance measures and data distributions. Let $D_{\min}^{(\delta,k)}(d)$ and $D_{\max}^{(\delta,k)}(d)$ be the minimum and maximum pairwise distances of all points in a d-dimensional dataset measured by the simplicial distance with parameter δ. Define the relative contrast (RC) as:

$$R^{(\delta,k)}(d) = \frac{D_{\max}^{(\delta,k)}(d) - D_{\min}^{(\delta,k)}(d)}{D_{\min}^{(\delta,k)}(d)}. \tag{3.15}$$

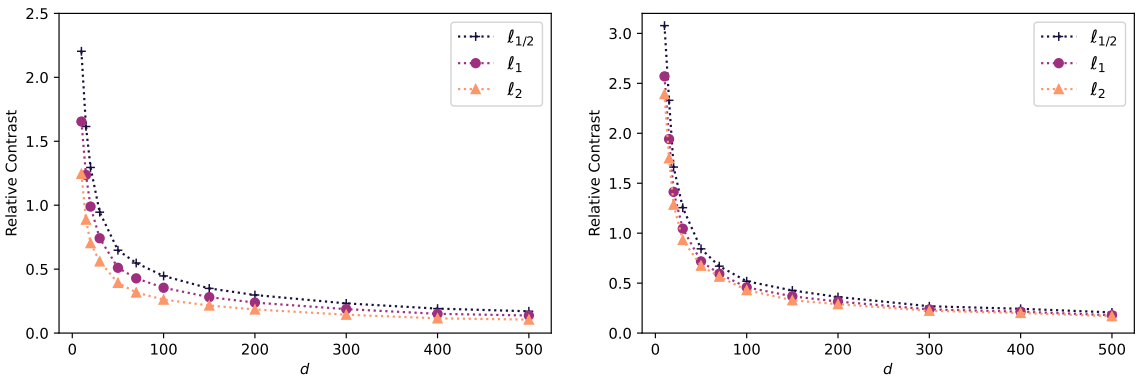
The minimum value of $R^{(\delta,k)}(d)$ is zero, when $D_{\max}^{(\delta,k)}(d) = D_{\min}^{(\delta,k)}(d)$. A small RC value near zero indicates that there is very little difference between the maximum and the minimum distances measured in the dataset, decreasing the meaningfulness of the distance.

It is shown in [5] that $R^{(2,1)}(d)$ (the RC of the Euclidean distance) is typically smaller than $R^{(1,1)}(d)$ (the RC of the Manhattan distance), particularly as d increases, which indicates that the ℓ_1 distance is often a more useful and intuitive distance measure in high dimensions. The authors also show that it can be beneficial to use ℓ_f metrics with $f \in (0,1)$ in high dimensional spaces. The examples given in [5] are extended below to the simplicial distances with different values of k and δ.

First, the RCs of the regular ℓ_δ distances are considered, for $\delta = 1/2$, $\delta = 1$ and $\delta = 2$. Figure 3.6 compares the RCs of these distances for different dimensions, for both uniformly distributed and normally distributed data. The methodology for uniformly distributed data is as follows, for each value of d:

1. Generate 100 d-dimensional datasets from the standard uniform distribution, each with 100 points. Centre each dataset by subtracting its mean.
2. For each dataset, find the distances from each point to the origin using a given ℓ_δ distance.
3. For each dataset, find the RC (3.15) of the distance measure.
4. Calculate the mean RC for each distance measure over the 100 datasets.

The mean RC over the 100 datasets is plotted in Figure 3.6a, for each value of δ. The same method is used in Figure 3.6b, but the datasets are generated from the standard multivariate Gaussian distribution. As expected, given the results in [5], the RC decreases as δ increases, which may indicate that $\delta = 1/2$ is the more suitable distance for high dimensional data analysis. However, the ℓ_δ distance with $\delta < 1$ is known to not satisfy the triangle inequality [135], making it a 'semi-distance' [251], rather than a formal distance measure. Consequently, any methods that rely on the triangle inequality cannot use an ℓ_δ distance with $\delta < 1$ [26, 71, 180, 183]. Therefore, it is recommended to use a well-defined distance rather than a semi-distance to avoid producing nonsensical results [175].

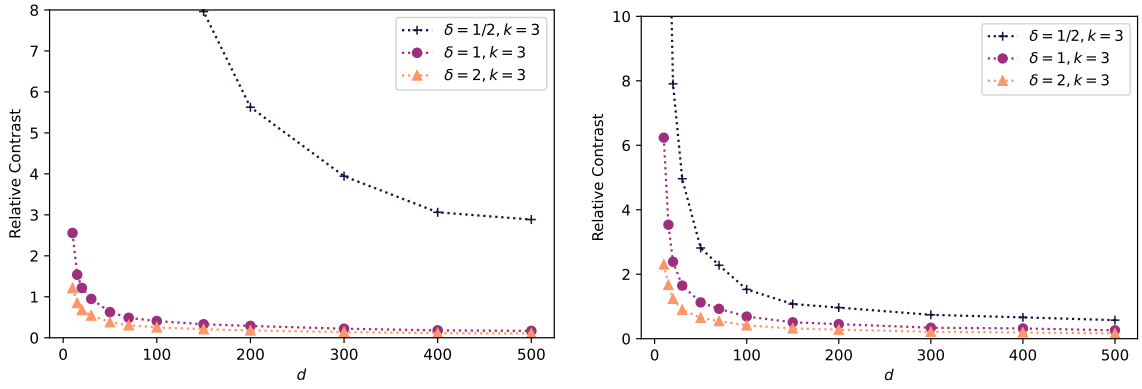


(a) Uniform data

(b) Gaussian data

Figure 3.6: Relative contrast of the $\ell_{1/2}$, ℓ_1 and ℓ_2 distances for (a) uniform data and (b) Gaussian data as d increases, averaged over 100 generated datasets.

This exercise is now repeated using the simplicial distances. Experiments showed that the RC of the simplicial distance was similar for different values of k, so only $k = 3$ is used in Figure 3.7. The simplicial distance with $\delta = 1/2$ clearly has a larger relative contrast than the simplicial distance with $\delta = 1$ and $\delta = 2$ for all values of d considered, for both the uniform and Gaussian examples. Using $\delta = 1$ also consistently has a higher relative contrast than $\delta = 2$. According to the claim that a higher RC means a more informative distance measure [5], the simplicial distance with parameter $\delta = 1/2$ would be the distance of choice, out of the three considered here. However, considering the previous discussion about semi-distances, one should proceed with caution if using the $\delta = 1/2$ parameter, and perhaps should consider sticking to the classical values of $\delta = 1$ and $\delta = 2$, as suggested in [175] for the ℓ_δ distances.



(a) Uniform data, $k = 3$

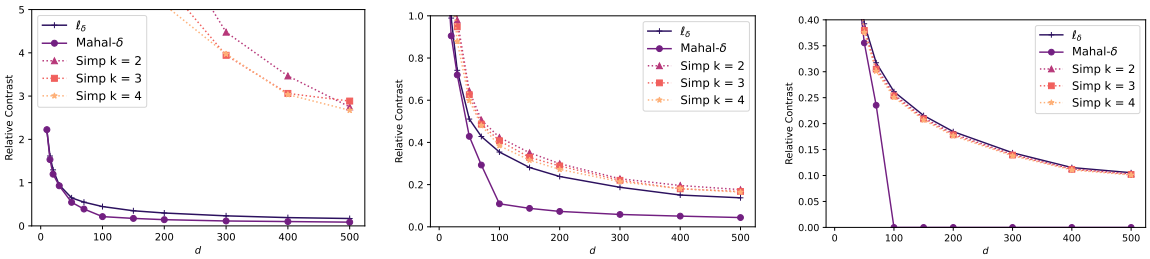
(b) Gaussian data, $k = 3$

Figure 3.7: Relative contrasts of the simplicial distances with $k = 3$ for (a) uniform data and (b) Gaussian data as d increases, averaged over 100 generated datasets.

The RC plots also allow for comparison between different types of distances, e.g. between simplicial distances and ℓ_δ distances. Define the 'Mahalanobis-δ' distance as follows. For a set of points $X \in \mathbb{R}^{d\times N}$, find the mean $\mu \in \mathbb{R}^d$ and covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$. Use the mean and the square root of the inverse covariance matrix to whiten and centre the data: $Y = \Sigma^{-1/2}(X - \mu) \in \mathbb{R}^{d\times N}$. Then find distances from all points in the whitened data Y to the origin with an ℓ_δ distance to give the Mahalanobis-δ distance. In Figures 3.8 and 3.9, the RC values of the ℓ_δ distances are compared to the Mahalanobis-δ distances, as well as the simplicial distance with different values of δ for $k = 2, 3, 4$.

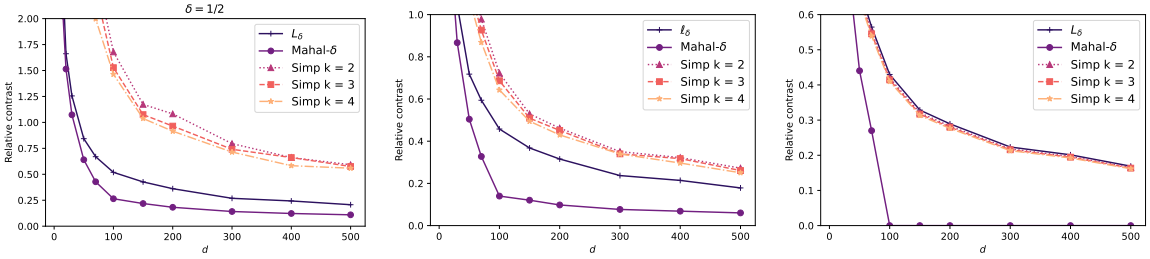
Figures 3.8 and 3.9 show that for all values of δ, the Mahalanobis-δ distance gives the

smallest RC. For $\delta = 2$, there is a negligible difference between the RC values of the ℓ_δ distance and the simplicial distance. For $\delta = 1$ and $\delta = 1/2$, the simplicial distance measures give higher RC than both the ℓ_δ and Mahalanobis-δ distances. According to the claims in [5], this indicates that using the simplicial distance with $\delta = 1$ or $\delta = 1/2$ gives more informative results than using the ℓ_δ or Mahalanobis-δ distances.



(a) $\delta = 1/2$ (b) $\delta = 1$ (c) $\delta = 2$

Figure 3.8: Relative contrasts of the ℓ_δ distances, Mahalanobis-δ distances and simplicial distances with $k = 2, 3, 4$ and different values of δ for uniformly distributed data with increasing d.



(a) $\delta = 1/2$ (b) $\delta = 1$ (c) $\delta = 2$

Figure 3.9: Relative contrasts of the ℓ_δ distances, Mahalanobis-δ distances and simplicial distances with $k = 2, 3, 4$ and different values of δ for Gaussian distributed data with increasing d.

Overall, it is recommended to use the simplicial distance with $\delta = 1$ or $\delta = 2$. The simplicial distance with $\delta = 1/2$ or other values of $\delta < 1$ can produce distances with higher RC values, but the cost of violating the triangle inequality can be great [26].

Choosing between the parameters $\delta = 1$ and $\delta = 2$ is a decision usually based on application. Much like the discussion of the ℓ_2 distance versus the ℓ_1 distance at the start of this section, the simplicial distance with $\delta = 1$ can provide a more robust distance measure that is resistant to outliers. It has been shown [5] that the Manhattan distance is a

more appropriate distance measure to be used in high dimensions than other ℓ_p distances ($p \in \mathbb{Z}^+$), as it has a higher RC. Based on this theory, one could deduce that using $\delta = 1$ in the simplicial distance is a better choice than using $\delta = 2$ for preserving contrast between the distances of the points.

Timing comparisons

The computation time may be another influence in the choice of the parameter δ in the simplicial distance. When using $\delta = 2$, there is a method of computation through polynomials, outlined in Section 3.2.2, which is often faster than computing the distances through simplices. Otherwise, if using $\delta \neq 2$, the volumes of simplices must be calculated directly. Sampling methods (discussed in Section 3.2.3) can help speed up the calculation of simplex volumes to improve computation time when using $\delta \neq 2$.

In this subsection, some simulations are performed to illustrate the differences in timings for different values of the parameter δ. For each value of $d \in \{3, 5, 10, 20, 30, 50, 100, 200, 500, 1000\}$, 100 different datasets are generated from the standard normal distribution, each with $N = 1000$ points. For values of $k \in \{3, 4, 5, 6, 7\}$, where $k \leq d$, the simplicial distance from all 1000 points to the dataset itself is measured. Table 3.5 and Table 3.6 give the mean time (and standard deviation) taken to measure the 1000 distances in one dataset when using the simplicial distance for $\delta = 2$ and $\delta = 1$, respectively.

Table 3.5 gives the mean time and standard deviation to calculate 1000 distances using $\delta = 2$ through polynomial methods. Within the polynomial S_k, the summation in Equation (3.9) includes powers of Σ^i ($i = \{0, 1, \ldots, k-1\}$) to find $q_k(\Sigma)$. As d increases the computation time increases, as it becomes more computationally expensive to find powers of the $d \times d$ matrix Σ. To improve computation time, it is recommended to iteratively multiply by Σ (this method is used to find the timings in Table 3.5) or use Horner's method for polynomial evaluation to find these powers, see Section 4.2 of [101]. Parallel processing could also be used to improve the computation speed, particularly as d gets large.

The timings given for $\delta = 1$ in Table 3.6 are slower than those for $\delta = 2$, for the most part, due to the need to compute the distance via simplex volumes. However, there is a lot of potential improvement to be made in computation time. For example, there may be more efficient ways to find the samples of simplices. Let g be the number of simplices

d \ k	3	4	5	6	7
3	0.015 ± 0.003	—	—	—	—
5	0.024 ± 0.004	0.025 ± 0.004	0.025 ± 0.005	—	—
10	0.042 ± 0.010	0.041 ± 0.008	0.041 ± 0.009	0.041 ± 0.010	0.043 ± 0.011
20	0.093 ± 0.009	0.091 ± 0.008	0.093 ± 0.009	0.094 ± 0.010	0.091 ± 0.008
30	0.171 ± 0.014	0.173 ± 0.013	0.170 ± 0.014	0.170 ± 0.011	0.172 ± 0.014
50	0.379 ± 0.020	0.374 ± 0.015	0.379 ± 0.016	0.383 ± 0.021	0.377 ± 0.016
100	1.558 ± 0.208	1.523 ± 0.163	1.544 ± 0.213	1.513 ± 0.169	1.529 ± 0.206
200	5.500 ± 0.758	5.521 ± 0.811	5.502 ± 0.668	5.484 ± 0.790	5.445 ± 0.649
500	24.597 ± 1.026	24.533 ± 0.989	24.362 ± 0.675	24.340 ± 0.744	24.492 ± 0.683
1000	89.273 ± 3.621	88.955 ± 2.749	89.248 ± 3.413	88.927 ± 2.705	88.619 ± 2.083

Table 3.5: The mean ± standard deviation time (in seconds) taken to compute the simplicial distance with $\delta = 2$ for N=1000 points for changing d and k, using the polynomial method from Section 3.2.2.

to be sampled. Currently, the code used for this example finds g random combinations of the indices $\{1,2,\ldots,N\}$, checks for duplicates and replaces them with a new random combination, and repeats until there are no duplicates. Furthermore, these distances were computed without the use of any parallel processing. Parallel processing could be used in two ways here: either by assigning points to different processors, which could speed the computation up considerably, or by finding the volumes of the g simplices in parallel.

Figure 3.10 gives a visualization of the mean time taken to produce the distances for different values of δ and k. When $\delta = 2$, different values of k have minimal impact on the time taken to produce the distances. However, the time taken to compute the distance is affected by the dimension d of the dataset.

On the other hand, Figure 3.10 shows that the time taken to compute distances when $\delta = 1$ is affected by the parameter k. The dimension of the dataset does have an effect on the time taken to compute distances, but not in a monotonic way, and this effect is not as pronounced as it is on $\delta = 2$.

d \ k	3	4	5	6	7
3	66.956 ± 3.920	—	—	—	—
5	69.163 ± 5.381	70.401 ± 3.927	73.411 ± 4.374	—	—
10	66.360 ± 4.327	67.743 ± 3.278	71.005 ± 4.861	75.329 ± 6.224	76.544 ± 4.746
20	63.099 ± 0.718	65.001 ± 0.647	67.578 ± 0.805	71.033 ± 0.967	73.723 ± 0.907
30	57.888 ± 6.962	60.010 ± 7.406	63.432 ± 8.002	65.949 ± 8.061	67.569 ± 7.656
50	52.631 ± 1.720	56.791 ± 1.874	56.289 ± 1.979	59.299 ± 2.207	61.488 ± 2.227
100	52.635 ± 1.173	56.483 ± 1.348	56.750 ± 1.243	60.303 ± 1.462	62.960 ± 1.348
200	54.216 ± 0.497	55.502 ± 0.378	59.479 ± 0.999	63.319 ± 1.289	67.446 ± 1.572
500	61.209 ± 1.040	62.235 ± 0.798	70.591 ± 0.758	75.887 ± 1.008	84.250 ± 1.592
1000	76.743 ± 6.663	76.971 ± 5.904	89.149 ± 8.086	96.216 ± 6.934	112.356 ± 8.644

Table 3.6: The mean ± standard deviation time (in seconds) taken to compute the simplicial distance with $\delta = 1$ for N=1000 points for changing d and k, using a sample of 1000 simplices.

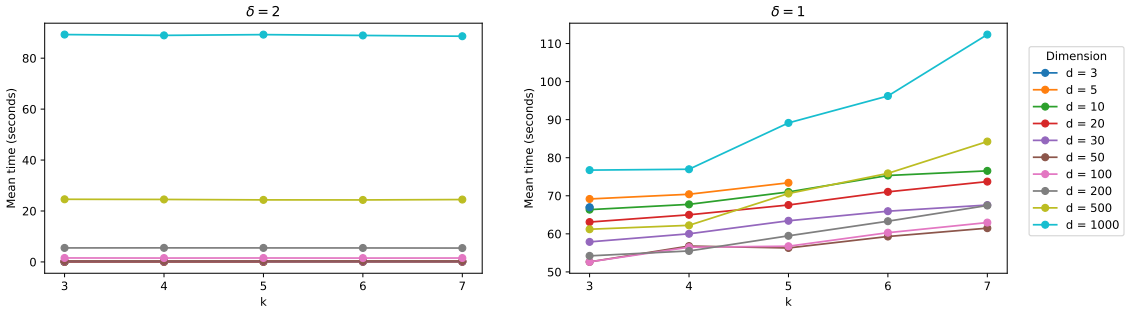


Figure 3.10: Plotting the mean time taken to find $N = 1000$ distances over 100 runs, using the simplicial distances with different values of δ and k. The left plot considers $\delta = 2$, the right plot considers $\delta = 1$ with a sample of 1000 simplices.

Overall, it seems that using $\delta = 2$ through the elementary symmetric function method in Section 3.2.2 is much faster than using $\delta = 1$ for smaller values of d with a sample of simplices. However, as d increases (particularly for $d = 1000$), this time advantage over $\delta = 1$ becomes less significant. This will be affected by the size of the sample of simplices when computing using volumes, but this sampling parameter can be adjusted according to the preference of time vs precision.

3.4 Distribution of the simplicial distances with $\delta = 2$

As in Section 3.2, let $X = \{x_1, \ldots, x_N\}$ be a $d \times N$ matrix. Let the d-dimensional vector μ be the sample mean of the N observations, and let Σ be the $d \times d$ sample covariance matrix of X. From Section 3.2.2, the k-simplicial distance with $\delta = 2$ can be written as the quadratic form

$$\rho_{k,2}^2(x, X) = (x-\mu)^\top \frac{S_k}{k}(x-\mu). \tag{3.16}$$

This makes it possible to use known properties of quadratic forms [171] to find representations of this distance through random variables, as well as the probability density function (PDF) and cumulative distribution function (CDF) of the distances produced using the simplicial distances. The translation of the points x by $-\mu$ gives $E\left[\rho_{k,2}^2(x, X)\right] = 0$, which simplifies the calculations.

3.4.1 Nonsingular case

First, consider the case where Σ is a nonsingular matrix. Define $W = \Sigma^{\frac{1}{2}} \frac{S_k}{k} \Sigma^{\frac{1}{2}}$. Let $P = [P_1, \ldots, P_d]$ be an orthogonal matrix which diagonalizes W; that is,

$$P^\top W P = P^\top \Sigma^{\frac{1}{2}} \frac{S_k}{k} \Sigma^{\frac{1}{2}} P = \operatorname{diag}(\psi_1, \ldots, \psi_d)$$

where $\Psi = \{\psi_1, \ldots, \psi_d\}$ are the eigenvalues of W. Define the matrix $U = P^\top \Sigma^{-\frac{1}{2}} X$. The distance measure can then be represented through random variables as

$$\rho_{k,2}^2(x, X) = \sum_{j=1}^{d} \psi_j U_j^2, \tag{3.17}$$

as given by Equation 3.1a.5 in [171]. It can be shown, thanks to this representation, that $\rho_{k,2}^2(x, X)$ is a linear combination of independent central chi-square variables.

The results of Section 4.2 of [171] give rise to the PDF and CDF of the distances when Σ is nonsingular. The PDF and CDF of $\rho_{k,2}^2(x, X)$ are, respectively,

$$f_d(\Psi, y) = \sum_{\ell=0}^{\infty} (-1)^\ell c_\ell \frac{y^{\frac{d}{2}+\ell-1}}{\Gamma(\frac{d}{2}+\ell)}, \; 0 < y < \infty \tag{3.18}$$

and

$$F_d(\Psi, y) = \sum_{\ell=0}^{\infty} (-1)^\ell c_\ell \frac{y^{\frac{d}{2}+\ell}}{\Gamma(\frac{d}{2}+\ell-1)}, \; 0 < y < \infty,$$

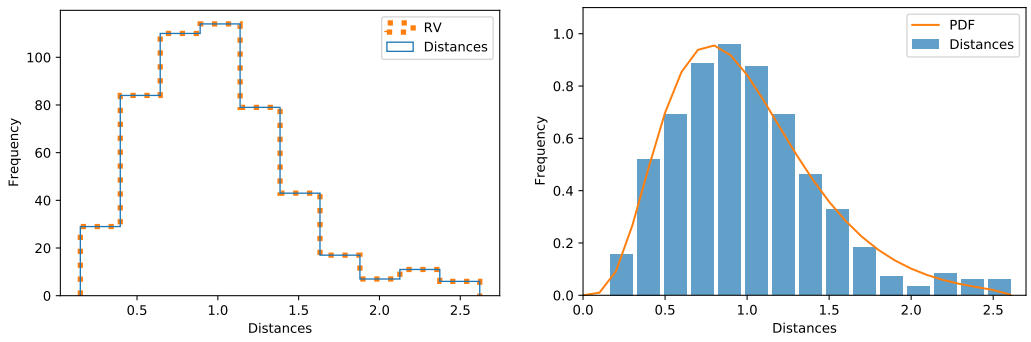
where

$$c_\ell = \begin{cases} \Pi_{j=1}^{d} (2\psi_j)^{-1/2} & \ell = 0, \\ \frac{1}{\ell} \sum_{j=0}^{\ell-1} g_{\ell-j} c_j & \ell \geq 1, \end{cases}$$

and

$$g_\ell = \frac{1}{2} \sum_{j=1}^{d} (2\psi_j)^{-\ell}.$$

The random variable representation and PDF equation are shown in practice in Figure 3.11. 500 observations are generated from a 10-dimensional dataset $X \sim \mathcal{N}(0, \Sigma)$, where Σ has eigenvalues [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]. The distance from every point $x \in X$ to the mean of the dataset X is calculated using the simplicial distance with $k = 6$, and is shown as the blue solid line in Figure 3.11a. The random variable representation as given in (3.17) is shown by the dotted orange line in Figure 3.11a. These two lines in Figure 3.11a coincide, showing the representation exactly matches the distances produced by the simplicial distance with $k = 6$. In Figure 3.11b, the blue histogram again shows the distances calculated using the simplicial distance, and the orange solid line shows the PDF calculated using Equation (3.18). The histogram closely matches the PDF line.



(a) Random variable representation

(b) PDF and distances

Figure 3.11: Using $k = 6$, the simplicial distance is measured from all points in a non-singular 10-dimensional dataset to the centre. Figure (a) compares these distances (blue solid line) with the random variable representation in (3.17) (orange dotted line); Figure (b) compares these distances (blue histogram) with the PDF given in Equation (3.18) (orange solid line).

3.4.2 Singular case

The case where Σ is a singular matrix with rank $r < d$ is now considered. Again following the results given in [171], write $\Sigma = BB^\top$, where B is a matrix of size $d \times r$ with rank r. Consider the linear transformation $X - \mu = BY$, where Y is an $r \times N$ matrix with $E[Y] = 0$ and identity covariance matrix. Then Equation (3.16) can be written as:

$$\rho^2_{k,2}(x, X) = (BY)^\top \frac{S_k}{k} BY = Y^\top B^\top \frac{S_k}{k} BY.$$

Let P be an orthogonal matrix such that

$$P^\top B^\top \frac{S_k}{k} BP = \text{diag}(\psi_1, \ldots, \psi_r),$$

where $\Psi = \{\psi_1, \ldots, \psi_r\}$ are the eigenvalues of $B^\top \frac{S_k}{k} B$. Letting $Z = P^\top Y$, there is then the following representation through random variables:

$$\rho^2_{k,2}(x, X) = Z^\top \text{diag}(\psi_1, \ldots, \psi_r) Z = \sum_{j=1}^{r} \psi_j Z_j^2. \tag{3.19}$$

Assuming $B^\top \frac{S_k}{k} B \neq 0$, the PDF and CDF of the simplicial distance in the singular case are given, respectively, as follows:

$$f_d(\Psi, y) = \sum_{\ell=0}^{\infty} c_\ell \frac{y^{\frac{r}{2}+\ell-1}}{\Gamma(\frac{r}{2}+\ell)}, \; 0 < y < \infty, \tag{3.20}$$

$$F_d(\Psi, y) = \sum_{\ell=0}^{\infty} c_\ell \frac{y^{\frac{r}{2}+\ell}}{\Gamma(\frac{r}{2}+\ell+1)}, \; 0 < y < \infty,$$

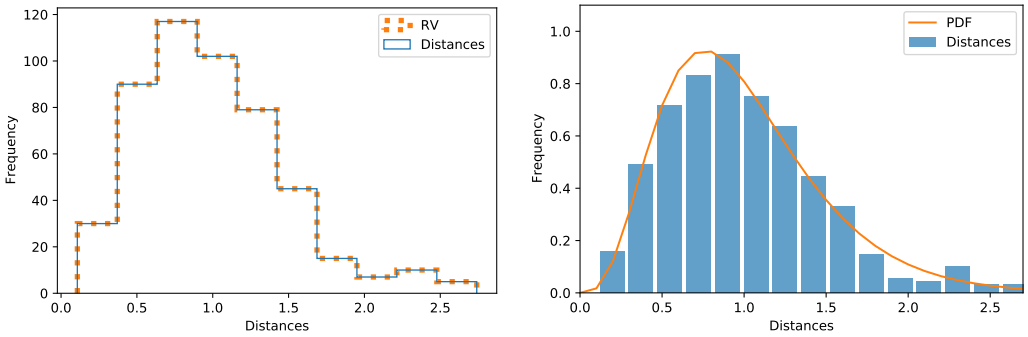
where

$$c_\ell = \begin{cases} \Pi_{j=1}^{r} (2\psi_j)^{-\frac{1}{2}} & \ell = 0, \\ \frac{1}{\ell} \sum_{j=0}^{\ell-1} g_{\ell-j} c_j & \ell \geq 1, \end{cases}$$

and

$$g_\ell = \frac{1}{2} \sum_{j=1}^{\ell} (2\psi_j)^{-\ell} (-1)^\ell.$$

Figure 3.12 repeats the findings of Figure 3.11 for the case where Σ is singular. As before, 500 observations are generated from a 10-dimensional dataset $X \sim \mathcal{N}(0, \Sigma)$, where Σ now has eigenvalues [10, 9, 8, 7, 6, 5, 4, 3, 2, 0]. The histogram of simplicial distance with $k = 5$ between every point $x \in X$ and the sample mean of X again coincides with the histogram produced by the random variable representation in Figure 3.12a. The orange line representing the PDF in Figure 3.12b is also a good fit to the simplicial distances shown by the histogram.



(a) Random variable representation

(b) PDF and distances

Figure 3.12: Using $k = 5$, the simplicial distance is measured from all points in a singular 10-dimensional dataset to the centre. Figure (a) compares these distances (blue solid line) with the random variable representation in Equation (3.19) (orange dotted line); Figure (b) compares these distances (blue histogram) with the PDF given in Equation (3.20) (orange solid line).

3.4.3 Moments of the simplicial distances with $\delta = 2$

The first four central moments of the simplicial distances can be found using Lemma 6.1 of [199]. The general form of the moments of a quadratic form are given in Appendix A.1. For a dataset X with mean μ and covariance matrix Σ, consider the k-simplicial distance with $\delta = 2$ from a point $x \in X$ to the dataset X as the quadratic form:

$$\rho_{k,2}^2(x,X) = (x-\mu)^\top \frac{S_k}{k}(x-\mu).$$

The expectation, variance, skewness and kurtosis of the k-simplicial distance with $\delta = 2$ are:

$$\begin{aligned}
\mathrm{E}\left(\rho_{k,2}^2(x,X)\right) &= \frac{1}{k}\mathrm{trace}\left(S_k\Sigma\right) \\
\mathrm{Var}\left(\rho_{k,2}^2(x,X)\right) &= \frac{2}{k^2}\mathrm{trace}\left([S_k\Sigma]^2\right) \\
\mathrm{Skew}\left(\rho_{k,2}^2(x,X)\right) &= \frac{2\sqrt{2}\mathrm{trace}\left([S_k\Sigma]^3\right)}{\mathrm{trace}\left([S_k\Sigma]^2\right)^{3/2}} \\
\mathrm{Kurt}\left(\rho_{k,2}^2(x,X)\right) &= \frac{12\mathrm{trace}\left([S_k\Sigma]^4\right)}{\mathrm{trace}\left([S_k\Sigma]^2\right)^2}.
\end{aligned} \tag{3.21}$$

For more information on the derivation of the equations in (3.21), see Appendix A.2. The necessary conditions for the moment formulae in (3.21) to hold are:

- The matrix S_k must be symmetric. S_k is a weighted sum of the covariance matrix Σ, which is symmetric by definition.

- $(x-\mu) \sim \mathcal{N}(0,\Sigma)$, with Σ positive definite. Given $x \sim \mathcal{N}(\mu,\Sigma)$, the centered value $(x-\mu)$ satisfies this condition. As Σ is the covariance matrix of X, it is positive definite by definition.

To check the accuracy of these moment formulae, Table 3.7 and Table 3.8 compare them with the empirical moments. To do so, 1000 random Gaussian datasets are generated and the simplicial distances (with various values of k) are calculated. The empirical moments are calculated for each dataset, and the mean (and standard deviation) for each of the moments over the 1000 datasets is reported in the table. These can then be compared to the mean of the theoretical moments given in (3.21). Table 3.7 considers $d = 10$, and Table 3.8 considers $d = 100$.

	Mean	Variance	Skewness	Kurtosis
k = 3				
Empirical	1.000 (6.82e-06)	0.316 (0.007)	1.267 (0.018)	2.619 (0.058)
Theoretical	1.000 (3.39e-16)	0.340 (0.019)	1.320 (0.046)	2.778 (0.203)
k = 5				
Empirical	1.000 (1.55e-06)	0.316 (0.004)	1.260 (0.014)	2.569 (0.061)
Theoretical	1.000 (1.08e-15)	0.289 (0.011)	1.154 (0.022)	2.070 (0.078)
k = 7				
Empirical	1.000 (6.41e-07)	0.300 (0.005)	1.215 (0.013)	2.410 (0.040)
Theoretical	1.000 (2.95e-13)	0.252 (0.008)	1.036 (0.013)	1.634 (0.037)

Table 3.7: Comparison of the empirical and theoretical means (standard deviations) of the moment values of the simplicial distance over 1000 runs for randomly generated 10-dimensional Gaussian datasets.

	Mean	Variance	Skewness	Kurtosis
k = 3				
Empirical	1.000 (8.53e-06)	0.041 (0.005)	0.552 (0.112)	1.088 (1.009)
Theoretical	1.000 (4.88e-16)	0.040 (0.001)	0.502 (0.005)	0.424 (0.011)
k = 5				
Empirical	1.000 (3.92e-07)	0.039 (0.001)	0.494 (0.004)	0.506 (0.026)
Theoretical	1.000 (5.06e-16)	0.039 (0.001)	0.493 (0.005)	0.406 (0.009)
k = 7				
Empirical	1.000 (2.39e-07)	0.039 (0.001)	0.486 (0.002)	0.452 (0.010)
Theoretical	1.000 (4.67e-16)	0.039 (0.008)	0.484 (0.004)	0.390 (0.008)

Table 3.8: Comparison of the empirical and theoretical means (standard deviations) of the moment values of the simplicial distance over 1000 runs for randomly generated 100-dimensional Gaussian datasets.

3.5 Applications of the simplicial distances

The possible applications of the simplicial distances are abundant, as many multivariate data analysis methods are reliant on measures of distance between points in space. As the dimension increases, finding a reliable distance measure becomes challenging, thanks to poor relative contrast (see Section 2.4.2) and counterintuitive geometrical properties (see Section 2.4.1). It is increasingly likely that data will have correlations as more variables are added, making the use of the Mahalanobis distance more desirable as dimensions increase. However, these correlations, combined with the often low-rank nature of high dimensional data [235], mean that the Mahalanobis distance is commonly unavailable due to degeneracy, and an alternative distance measure must be used.

The applications considered here include outlier detection, K-means clustering and the whitening of datasets. Of course, the need for distances that are usable in degenerate and correlated datasets stretch far beyond these examples, and can be considered in applications as varied as approximate Bayesian computation [10], image processing [269, 270] and support vector machines [242]. The Mahalanobis distance is used across many fields, including chemometrics [42, 62], finance [137, 224] and genomics [226, 256].

3.5.1 Outlier labelling

This section describes one potential application of the simplicial distance measure. The simplicial distance is a useful tool in identifying outlying points in high dimensional degenerate datasets, where the Euclidean distance struggles to measure distance meaningfully, and the Mahalanobis relies on the inversion of a matrix possessing many small (and possibly zero) eigenvalues. This example considers how the parameter k and the scalar power δ affect the performance of outlier detection using simplicial distances.

Dataset	Eigenvalues	μ_1	μ_2
D_I	$\Lambda_I = [100, 10, 1, 1] + [0.00001] * 5 + [0]$	$[0] * 10$	$[1] * 10$
D_{II}	$\Lambda_{II} = [100, 10, 1, 1] + [0.00001] * 5 + [0]$	$[0] * 10$	$[0] * 5 + [1] * 5$
D_{III}	$\Lambda_{III} = [100, 4, 3, 2, 1] + [0.00001] * 4 + [0]$	$[0] * 10$	$[1] * 10$

Table 3.9: Details of the datasets to be used in outlier labelling. All have $d = 20$ and are made of two clusters of different sizes and different means μ_1 and μ_2, but the same covariance matrix with the eigenvalues given in the table. The table gives details in Python notation.

Three examples will be given, each with different sets of data. Each dataset D_i is made up of two clusters: $D_{i,1}$ and $D_{i,2}$ for $i = I, II, III$. The first cluster $D_{i,1}$ has 450 points, mean μ_1 as specified in Table 3.9 and covariance matrix produced by a matrix with eigenvalues as specified in the table, rotated by a rotation matrix. See Appendix C.1 for more information on the rotation method. The second cluster $D_{i,2}$ has 50 points, a different mean μ_2 but the same covariance matrix as $D_{i,1}$. By doing this, the robustness of the distances against rotations and correlations in the data is tested, as well as the ability to tell two similar but separate clusters apart.

The distances of all points in the dataset D_i to the larger cluster $D_{i,1}$ are measured. The furthest 50 points from this larger cluster $D_{i,1}$ are labelled as outliers for each dataset i. Table 3.10 shows how many of the points the simplicial distances correctly labelled as inliers and outliers, for different values of k and δ. If the method incorrectly labels all the outlying points as inliers, the minimum value of 400 is achieved. A score of 500 indicates all points were labelled correctly as outliers and inliers.

	Dataset D_I		Dataset D_{II}		Dataset D_{III}	
k	$\delta = 2$	$\delta = 1$	$\delta = 2$	$\delta = 1$	$\delta = 2$	$\delta = 1$
1	412	412	408	408	406	406
2	412	412	418	418	420	416
3	440	444	432	442	436	436
4	492	500	482	498	444	448
5	500	500	500	500	482	496
6	500	500	500	500	500	500
7	414	500	416	500	498	500
8	410	500	408	500	406	500
9	412	500	410	500	406	500

Table 3.10: Number of points correctly identified as outliers and inliers by the simplicial distances with differing values of k and δ, for datasets detailed in Table 3.9. Minimum value is 400, maximum value is 500.

Table 3.11 provides the area under the receiver operating characteristic curve (AUC) score for the labels produced by the distances, for different values of k and δ. The AUC score measures the overall performance of a binary classifier, where a score of 1 indicates a perfect labelling and 0 is the minimum score, with a score of 0.5 indicating an uninformative classifier.

Considering the simplicial distance with $\delta = 2$, the values of k that perform best are those that roughly correspond to the number of 'large' eigenvalues, as explained in Section 3.3.1. For dataset D_I, there are 4 'large' eigenvalues and the values $k = \{4, 5, 6\}$ perform best when using $\delta = 2$. Similar results are shown in datasets D_{II} and D_{III}. Larger values of k begin to break down when $\delta = 2$ as they require the use of the smaller eigenvalues in all distance calculations, indicating that lower values of k outperform the squared Moore-Penrose Mahalanobis distance over r.

Distances using $\delta = 1$ are more robust to the effect of degeneracy. The performance improves as k increases, but unlike the $\delta = 2$ case, there is no breakdown in success once k encompasses the smaller eigenvalues too, making it less sensitive to the choice of k than the distance with $\delta = 2$. This is likely due to the instability in the polynomials used to com-

	Dataset D_I		Dataset D_{II}		Dataset D_{III}	
k	$\delta = 2$	$\delta = 1$	$\delta = 2$	$\delta = 1$	$\delta = 2$	$\delta = 1$
1	0.51	0.51	0.49	0.49	0.48	0.48
2	0.51	0.51	0.54	0.54	0.56	0.53
3	0.67	0.69	0.62	0.68	0.64	0.64
4	0.96	1.00	0.90	0.99	0.69	0.71
5	1.00	1.00	1.00	1.00	0.90	0.98
6	1.00	1.00	1.00	1.00	1.00	1.00
7	0.52	1.00	0.53	1.00	0.99	1.00
8	0.50	1.00	0.49	1.00	0.48	1.00
9	0.51	1.00	0.50	1.00	0.48	1.00

Table 3.11: AUC scores for outlier detection when using the simplicial distance with different values of k and δ, for datasets detailed in Table 3.9.

pute the distances with $\delta = 2$. The distances using $\delta = 1$ were computed through simplex volumes using very low sampling amounts, and so there is not considerable computational time disadvantage in using $\delta = 1$ over $\delta = 2$, as seen in Section 3.3.2.

Overall, this example shows that using $\delta = 1$ can produce a more stable and robust distance measure than using $\delta = 2$ as k increases, particularly for outlier detection applications.

3.5.2 K-means clustering

K-means clustering is an unsupervised machine learning algorithm, used to group data into K groups, known as 'clusters'. The K-means algorithm aims to find K groups within the data, each with a centre point (known as a 'centroid') such that the total sum of distances between the points and their respective cluster centroid is minimized. An overview of the method is given in Algorithm 1. The K-Means algorithm is classically applied using the Euclidean distance, but research has shown success in applying the algorithm with the Mahalanobis distance to exploit the covariance structure of a dataset [87, 174]. The performance of K-means clustering can be highly dependent on the initial method of picking the first centroids, also known as the 'initialisation method' [53]. For further details on the algorithm and its modifications, see [7, 120, 160, 221].

Algorithm 1: K-means algorithm

Input: K: the number of clusters desired; $X = \{x_1, x_2, \ldots, x_N\}$: data matrix; initialisation method; $D(x, y)$: distance measure

Output: Produce K clusters of the points

1 Find the initial set of cluster centroids $\{c_1, \ldots, c_k\}$ for the clusters $C_1, \ldots, C_k$, using the initialisation method chosen.

2 **for** $x_i \in X$ **do**

3 Compute the distance from x_i to each cluster centroid c_j using the chosen distance measure $D(x_i, c_j)$.

4 Let c^* be the cluster centroid which minimizes this distance:

$$c^* = \arg\min_{c_j} D(x_i, c_j)$$

5 Assign x_i to the cluster associated with centroid c^*.

6 **end**

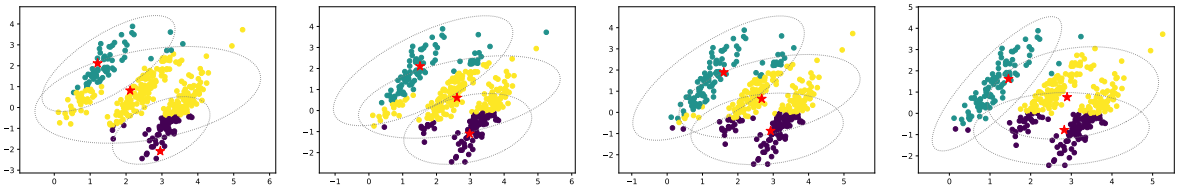
7 Update the cluster centroids with the mean of all the points in each cluster. For $j = 1, \ldots, K$:

$$c_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$$

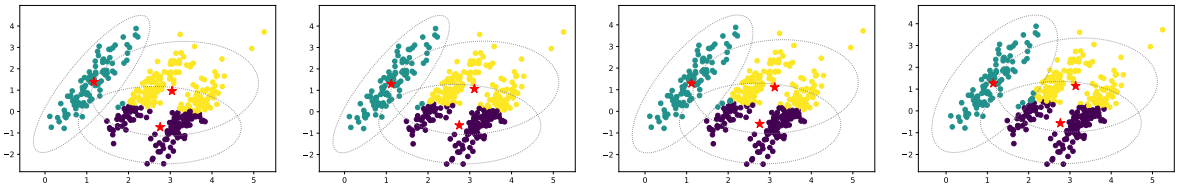
8 Repeat steps 2-7 until there is no reassignment of clusters to any points and no movement of centroids, or some other stopping criterion.

Figure 3.13 shows iterations of the K-means algorithm on a 2-dimensional dataset generated to have 3 clusters. K-means is applied here using the state of the art Python package Scikit-Learn [185]. The three clusters have the same covariance matrix, but different means. In this example, the Euclidean distance does not correctly identify the three clusters due to the elliptical nature of the clusters. When using the Euclidean distance, it is assumed that the clusters are spherical.

To make use of the simplicial distance (or Mahalanobis distance) when using K-means, an initial estimate of the clusters is needed to provide a starting covariance matrix for each of the clusters [56]. Figure 3.14 shows the iterations of K-means clustering on the same dataset as the one in Figure 3.13, but using the simplicial distance with $k = 2$ and $\delta = 2$ as the distance function. Iteration 8 of the Euclidean K-means (Figure 3.13h) is used as the



(a) Iteration 1 (b) Iteration 2 (c) Iteration 3 (d) Iteration 4



(e) Iteration 5 (f) Iteration 6 (g) Iteration 7 (h) Iteration 8

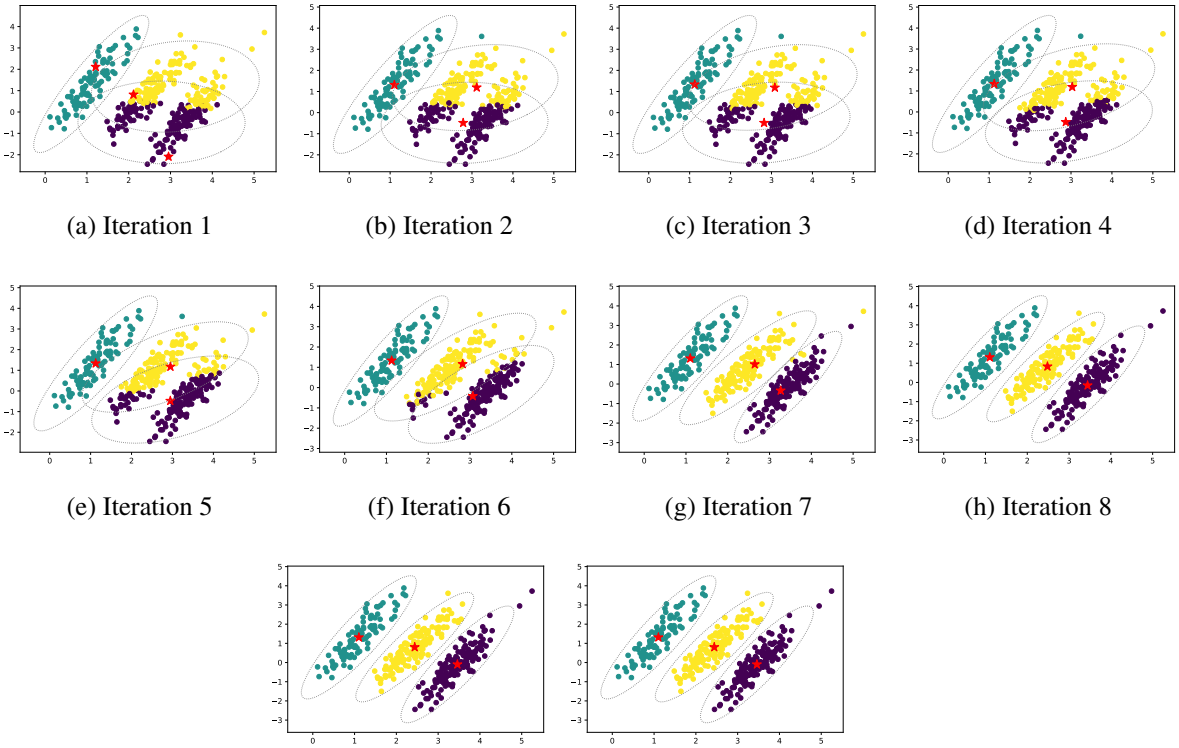
Figure 3.13: Iterations of K-means with number of clusters $K = 3$, using the Euclidean distance. Different colours indicate clusters, red stars indicate centroids and grey dotted ellipses show the confidence ellipse for each cluster with 3 standard deviations.

starting point. The distance from every point $x_i \in X$ to each cluster centroid c_j is found using the respective sample covariance matrix Σ_j of that cluster in that iteration. Given that $k = d$ and $\delta = 2$, this is equivalent to using the Mahalanobis distance (upto a scaling factor):

$$\rho^2_{2,2}(x_i, C_j) = \frac{1}{2}(x_i - c_j)^\top \Sigma_j^{-1}(x_i - c_j)$$

As the simplicial and Mahalanobis distances can account for non-spherical covariance matrices, K-means using these distances correctly identifies the clusters in Figure 3.14.

The benefits of using the simplicial distance for applications such as clustering become more evident as the dimensionality of the dataset increases. Consider a 20-dimensional dataset made of three clusters, again all generated with the same covariance matrix Σ. Σ is generated using eigenvalues `[10, 0.5] + [0.3 ** i for i in range(18)]` (using Python notation), and then rotated using the method detailed in Appendix C.1. This creates a degenerate, correlated covariance matrix with a rank that is hard to detect. K-means clustering is performed 500 times using three distance measures: the Euclidean distance, the Mahalanobis distance with the Moore-Penrose pseudoinverse (denoted here by Mahalanobis-pinv) and the simplicial distance with $k = 3$, all of which are performed using the steps given in Algorithm 1. The adjusted rand (AR) score is used to compare



(a) Iteration 1 (b) Iteration 2 (c) Iteration 3 (d) Iteration 4

(e) Iteration 5 (f) Iteration 6 (g) Iteration 7 (h) Iteration 8

(i) Iteration 9 (j) Iteration 10

Figure 3.14: Iterations of K-means with number of clusters $K = 3$, using the simplicial distance with $k = 2$ and $\delta = 2$. Different colours indicate clusters, red stars indicate centroids and grey dotted ellipses show the confidence ellipse for each cluster with 3 standard deviations.

the 'true' labels to the labels returned by the K-means algorithm. This metric is a measure of similarity between two cluster labellings and is adjusted for chance (for more information, see Appendix D.1 or [113, 185]). Figure 3.15 shows violin plots of the adjusted rand scores.

Due to the non-spherical distribution of the clusters, the Euclidean distance does not perform as well in the clustering algorithm as those distances that account for elliptical distributions. The Mahalanobis-pinv distance performs better than the Euclidean distance thanks to this, but the lack of clarity in the rank of the clusters causes issue in using the Moore-Penrose pseudoinverse (see Section 2.5.4 for more information on the downfalls of the Moore-Penrose pseudoinverse). The true inverse is not available for use within the Mahalanobis distance here due to singularity. The simplicial distance with $k = 3$ performs much more successfully, with median AR score equal to 1. There is a wider spread of

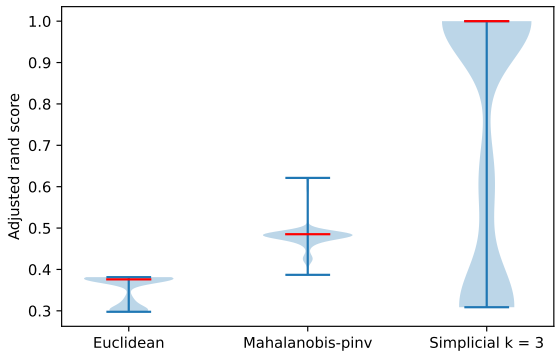


Figure 3.15: Violin plot showing the distribution of adjusted rand scores of 500 runs of K-means clustering with the Euclidean, Mahalanobis-pinv and simplicial $k = 3$ distances. The red line shows the median adjusted rand score of the 500 runs for each distance.

AR scores for the simplicial distance, but the K-means algorithm is usually run several times to find the best performing labelling, due to possible influences from initializations and the possibility of falling into local minima [185]. This example shows that there are circumstances in which the Euclidean distance and Mahalanobis distance do not perform as well as the simplicial distance in clustering a dataset; particularly when the dataset is correlated and degenerate.

3.5.3 Data whitening with simplices

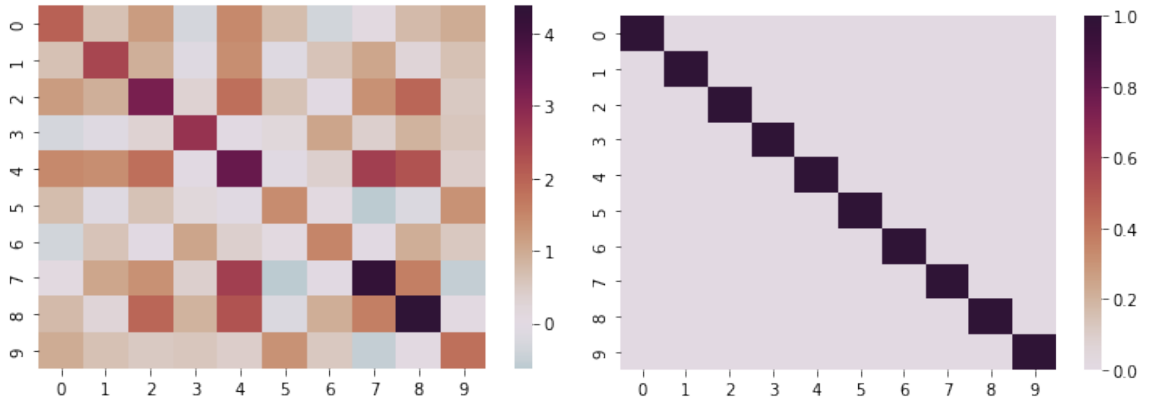
Let $X \sim \mathcal{N}_d(\mu, \Sigma)$. When using $\delta = 2$ in the simplicial distances, the matrix S_k is used as an alternative to Σ^{-1} when finding distances of the form $\rho^2_{\Sigma^{-1}}(x, X) = (x-\mu)^\top \Sigma^{-1}(x-\mu)$. There are many applications where it is beneficial to whiten a dataset, see Section 2.3 in the literature review for some examples. Whitening a full-rank dataset transforms the dataset to have mean zero and covariance matrix equal to the $d \times d$ identity matrix. A whitening transformation usually takes the form

$$X_W = W(X - \mu)$$

where W is referred to as the whitening matrix. In Mahalanobis whitening, $W = \Sigma^{-1/2}$. It is proposed here that $W = S_k^{1/2}$ is used in place of $\Sigma^{-1/2}$; this method will be referred to as 'simplicial whitening'.

As an example, consider a dataset $X^{(A)}$ randomly generated from a multivariate normal distribution in $d = 10$ dimensions, with mean zero and covariance matrix $\Sigma^{(A)}$, where $\Sigma^{(A)}$

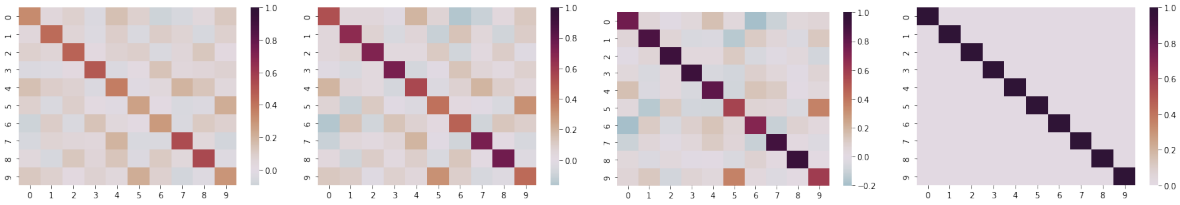
is generated to have eigenvalues [10, 5, 4, 3, 2, 1.5, 1, 0.5, 0.1, 0.01] and rotated using the method detailed in Appendix C.1. The whitening of $X^{(A)}$ is illustrated using heatmaps of the covariance matrix: Figure 3.16a shows the heatmap of the covariance matrix of $X^{(A)}$, and Figure 3.16b shows the heatmap after Mahalanobis whitening, which is exactly equal to the identity matrix.



(a) Covariance matrix of dataset $X^{(A)}$

(b) Covariance matrix of $\Sigma^{-1/2}X^{(A)}$

Figure 3.16: Heatmaps of the covariance matrix of the dataset $X^{(A)}$ (a) before whitening and (b) after being whitened by the inverse square root of the covariance matrix.

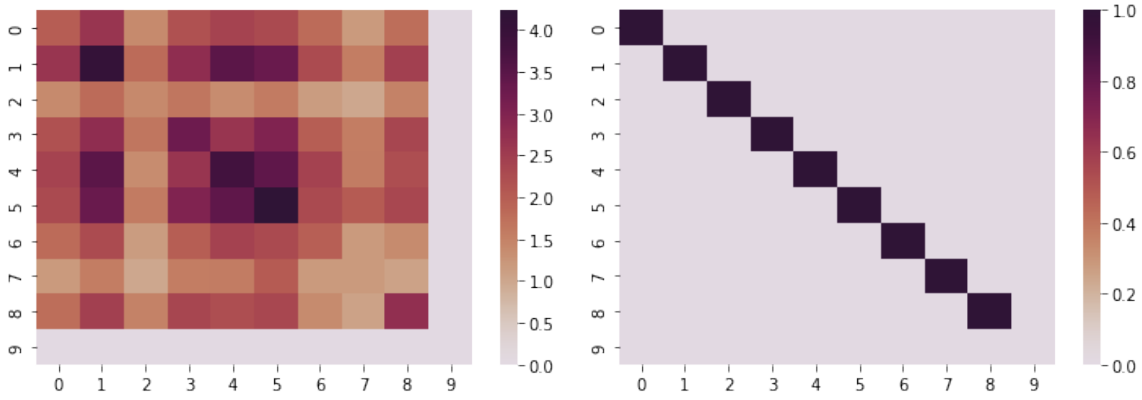


(a) $S_4^{1/2}X^{(A)}$ (b) $S_6^{1/2}X^{(A)}$ (c) $S_8^{1/2}X^{(A)}$ (d) $S_{10}^{1/2}X^{(A)}$

Figure 3.17: Heatmaps of the covariance matrices of the dataset $X^{(A)}$ after being whitened by the square root of the simplicial matrix S_k with (a) $k = 4$, (b) $k = 6$, (c) $k = 8$, (d) $k = 10$.

As with the simplicial distances, there is a gradual movement towards results similar to Mahalanobis whitening as k increases in simplicial whitening, with $k = d$ equal exactly to the Mahalanobis whitening result.

The above exercise is repeated with a degenerate dataset. $X^{(B)}$ is again generated from a 10-dimensional multivariate normal distribution with zero mean. Let L be a uniform random matrix generated by the `numpy.random.rand` function in Python. Set the last

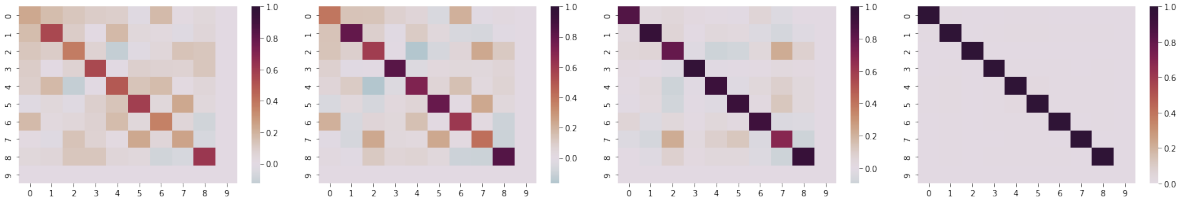


(a) Covariance matrix of dataset $X^{(B)}$

(b) Covariance matrix of $\Sigma^{-1/2}X^{(B)}$

Figure 3.18: Heatmaps of the covariance matrix of the dataset $X^{(B)}$ (a) before whitening and (b) after being whitened by the Moore-Penrose pseudoinverse square root of the covariance matrix.

row and column of L to be all zeros, and then let $\Sigma^{(B)} = L^\top L$ be the covariance matrix used to generate $X^{(B)}$. The eigenvalues of the empirical covariance matrix of $X^{(B)}$ are [20.09, 1.53, 1.16, 0.80, 0.57, 0.35, 0.13, 0.05, 0.02, 0]. Since the covariance matrix is singular, the square root of the Moore-Penrose pseudoinverse is used in Figure 3.18b. The fully whitened covariance matrix is equal to the identity matrix with the last diagonal entry equal to zero.



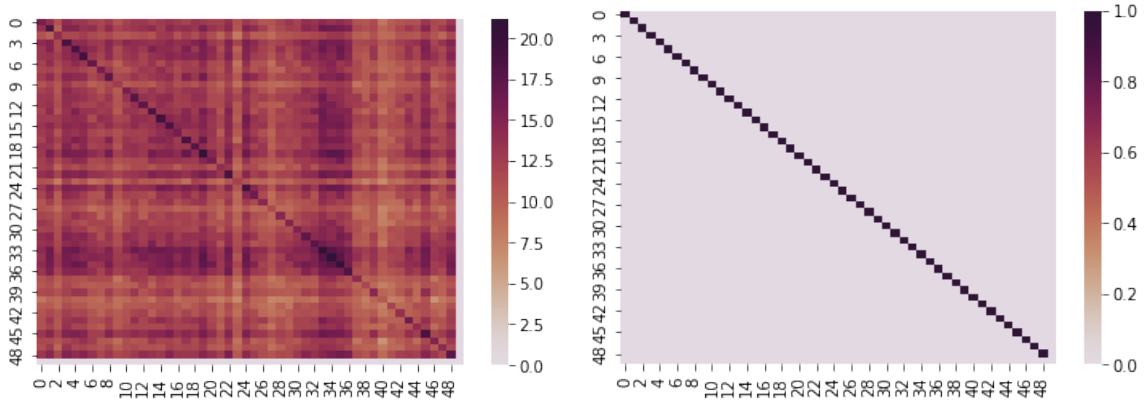
(a) $S_4^{1/2}X^{(B)}$ (b) $S_6^{1/2}X^{(B)}$ (c) $S_8^{1/2}X^{(B)}$ (d) $S_9^{1/2}X^{(B)}$

Figure 3.19: Heatmaps of the covariance matrices of the dataset $X^{(B)}$ after being whitened by the square root of the simplicial matrix S_k with (a) $k = 4$, (b) $k = 6$, (c) $k = 8$, (d) $k = 9$.

As with the previous example, the whitening becomes gradually more successful as k increases. When $k = r$, where r is the rank of $X^{(B)}$, simplicial whitening produces the same results as whitening with the square root of the Moore-Penrose pseudoinverse.

Consider a third dataset $X^{(C)}$ in 50-dimensions. The dataset is generated in the same way as dataset $X^{(B)}$, and has eigenvalues [610.83, 15.84, 14.94, 12.82, 12.17, 11.37, 10.51,

9.60, 8.81, 8.15, 7.95, 7.15, 6.50, 5.99, 5.47, 5.26, 4.64, 4.19, 3.94, 3.92, 3.52, 3.33, 3.06, 2.82, 2.64, 2.40, 2.14, 1.89, 1.77, 1.56, 1.39, 1.30, 1.04, 0.95, 0.87, 0.75, 0.66, 0.44, 0.36, 0.34, 0.28, 0.26, 0.17, 0.06, 0.05, 0.03, 0.03, 0, 0, 0]. Figure 3.20 shows the heatmaps of the covariance matrix of $X^{(C)}$ and the covariance matrix after whitening by the square root of the Moore-Penrose pseudoinverse matrix.



(a) Covariance matrix of dataset $X^{(C)}$

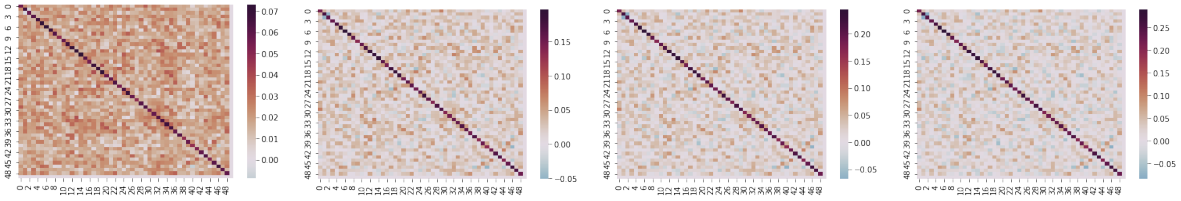
(b) Covariance matrix of $\Sigma^{-1/2}X^{(C)}$

Figure 3.20: Heatmaps of the covariance matrix of the dataset $X^{(C)}$ (a) before whitening and (b) after being whitened by the Moore-Penrose pseudoinverse square root of the covariance matrix.

Figure 3.21 shows heatmaps of the covariance matrices after simplicial whitening with different values of k. As with the 10-dimensional examples, a gradual improvement is shown as k increases. However, perfect whitening (like the Moore-Penrose pseudoinverse in Figure 3.20b) is not reached, as after $k = 12$ there is instability in the whitening.

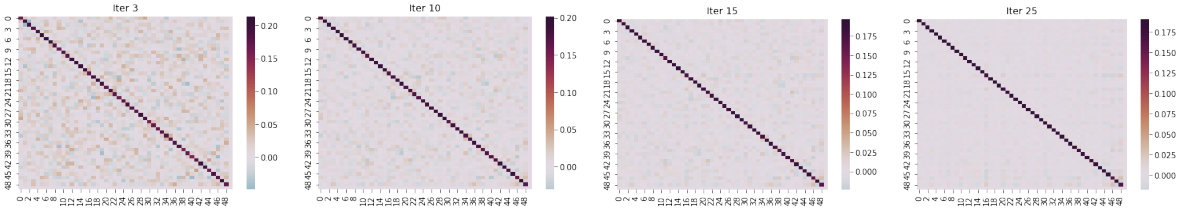
To improve the simplicial whitening results, a method called 'iterative whitening' will briefly be introduced here. This method will be more formally introduced in Section 5.4. For simplicity and without loss of generality, assume $\mu = 0$. Let Σ_0 be the covariance matrix of the dataset $X = X^{(C)}$.

To use iterative whitening, apply the whitening transformation $X_{W_1} = W_1 X$, where $W_1 = S_k^{1/2}$ using covariance matrix Σ_0. Find the covariance matrix of X_{W_1}, denoted Σ_1, and find a new whitening matrix $W_2 = S_k^{1/2}$ using covariance matrix Σ_1. X_{W_1} is then whitened by calculating $X_{W_2} = W_2 X_{W_1}$. This is repeated as many times as desired. As shown by Figure 3.22, this results in much improved whitening capabilities using simplicial whitening.



(a) $S_3^{1/2}X^{(C)}$ (b) $S_8^{1/2}X^{(C)}$ (c) $S_{10}^{1/2}X^{(C)}$ (d) $S_{12}^{1/2}X^{(C)}$

Figure 3.21: Heatmaps of the covariance matrices of the dataset $X^{(C)}$ after being whitened by the square root of the simplicial matrix S_k with (a) $k = 3$, (b) $k = 8$, (c) $k = 10$, (d) $k = 12$.



(a) Iteration 3 (b) Iteration 10 (c) Iteration 15 (d) Iteration 25

Figure 3.22: Heatmaps of the covariance matrix of the dataset $X^{(C)}$ after being iteratively whitened by the square root of the simplicial matrix S_k with $k = 9$ after (a) 3 iterations, (b) 10 iterations (c) 15 iterations, (d) 25 iterations.

Of course, applying iterative whitening is more computationally intensive than applying the whitening transformation once. Figure 3.23 illustrates the time taken to compute the whitened dataset with iterative whitening using different values of k. The black dotted line also shows how long it takes to compute the whitened dataset using $k = 12$ and no iteration, as given in Figure 3.22(d). This is clearly faster, but the results given by iterative whitening are superior, and so it is likely worthwhile to use the slightly more expensive iterative method.

The last dataset considered, $X^{(D)}$, is a 64-dimensional real dataset known as 'Digits'. This dataset will be used throughout examples in this thesis, see Section 5.3.1 for more information (specifically Table 5.2). The dataset has rank $r = 61$, number of observations $N = 1791$, and the covariance matrix of the dataset is given in Figure 3.24a. As $r < d$, the covariance matrix of the dataset is singular. The square root of the Moore-Penrose pseudoinverse is therefore used to perform Mahalanobis whitening, the results of which are given in Figure 3.24b.

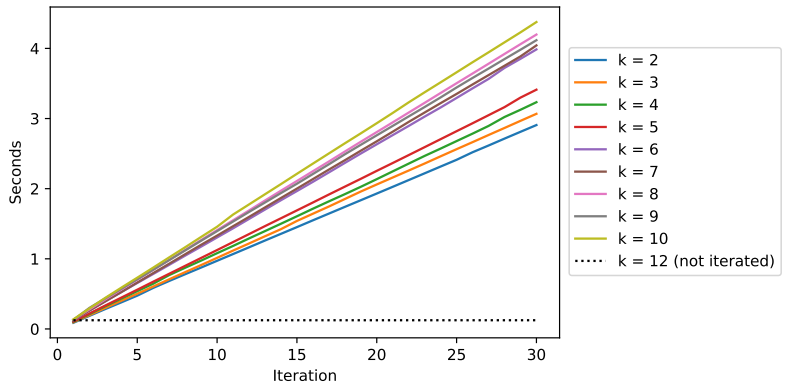
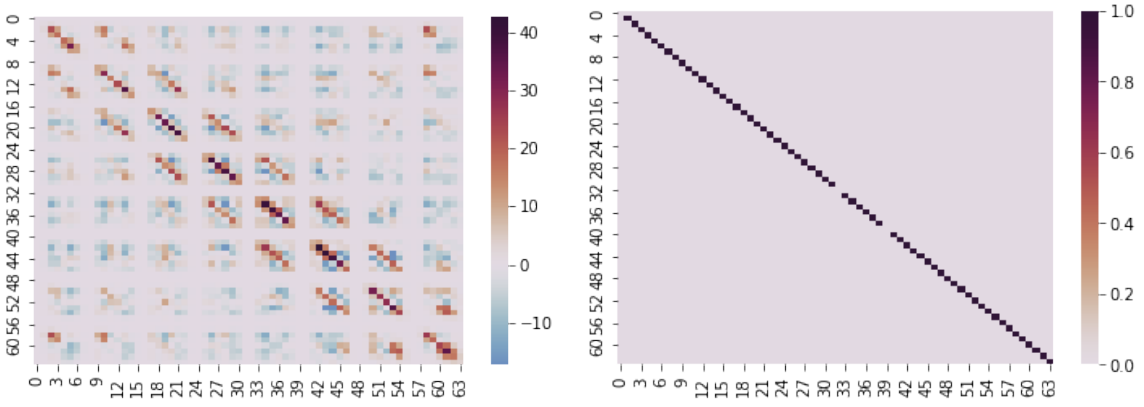


Figure 3.23: Time taken to compute simplicial whitened copies of X_C, using iterative whitening with different values of k. The black dotted line shows the time to compute a non-iteratively whitened dataset using $k = 12$.

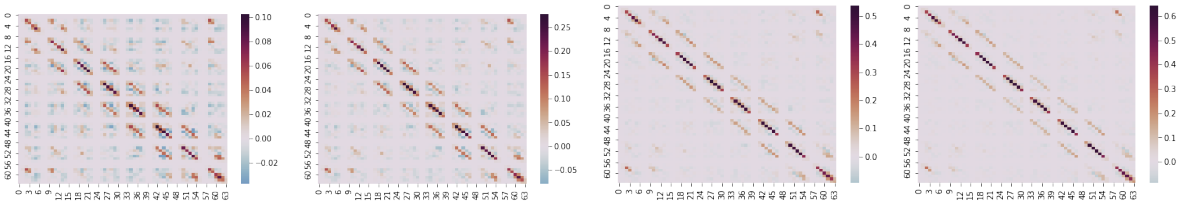
Figure 3.25 shows that simplicial whitening does remove some correlations, but doesn't whiten the dataset as well as the Moore-Penrose pseudoinverse does in Figure 3.24b. As with the previous example, the whitening transformation can be improved by applying several iterations of simplicial whitening, as in Figure 3.27, where the dataset is near-whitened (although requires some scaling). Such an improvement is more time consuming, as demonstrated in Figure 3.26. Lower values of k can be used to improve such time cost, at the potential expense of results, making this a performance versus time tradeoff, as experiments show higher values of k perform better for this example.



(a) Covariance matrix of dataset Digits

(b) Covariance matrix of whitened dataset Digits

Figure 3.24: Heatmaps of the covariance matrix of the dataset Digits (a) before whitening and (b) after being whitened by the Moore-Penrose pseudoinverse square root of the covariance matrix.



(a) $S_3^{1/2}X^{(D)}$ (b) $S_9^{1/2}X^{(D)}$ (c) $S_{19}^{1/2}X^{(D)}$ (d) $S_{24}^{1/2}X^{(D)}$

Figure 3.25: Heatmaps of the covariance matrices of the dataset Digits after being whitened by the square root of the simplicial matrix S_k with (a) $k = 3$, (b) $k = 9$, (c) $k = 19$, (d) $k = 24$.

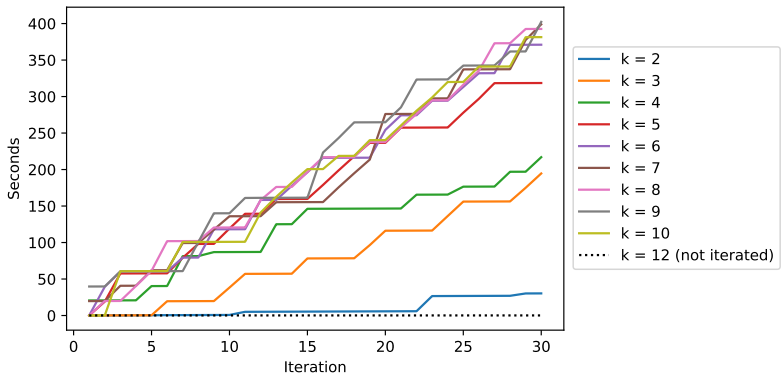


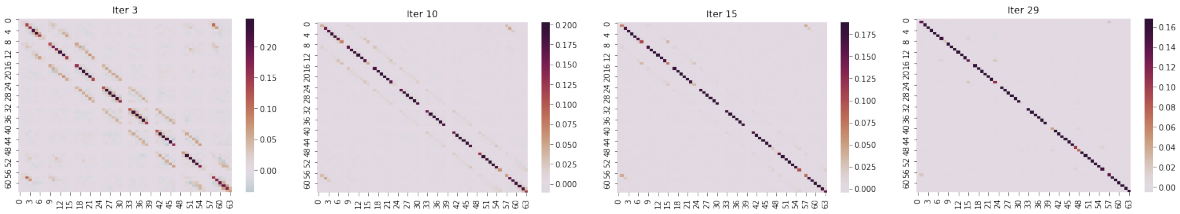
Figure 3.26: Time taken to compute simplicial whitened copies of the Digits dataset, using iterative whitening with different values of k. The black dotted line shows the time to compute a non-iteratively whitened dataset using $k = 12$.

Overall, the simplicial distance matrix S_k can perform as a good alternative to the inverse covariance matrix when it is not available, as has been demonstrated in a data whitening setting amongst other applications.

3.6 Chapter summary

This chapter has presented the simplicial distances, a spectrum of distance measures found using the volumes of k-dimensional simplices formed by data observations. The distance was first introduced in [192], and has been adapted and built upon in [85] and this chapter. The benefits of this distance measure include:

- The ability to account for correlations and rotations in the data when measuring distances, which many common multivariate distance measures fail to do;



(a) Iteration 3 (b) Iteration 10 (c) Iteration 15 (d) Iteration 29

Figure 3.27: Heatmaps of the covariance matrix of the dataset $X^{(D)}$ (so-called 'Digits') after being iteratively whitened by the square root of the k-simplicial matrix S_k with $k = 9$ after (a) 3 iterations, (b) 10 iterations (c) 15 iterations, (d) 29 iterations.

- The ability to be used in degenerate (and near-degenerate) data, unlike the Mahalanobis distance;
- The amenability of the distance measure using parameters, making it more suitable for varying applications;
- The lack of assumptions imposed on the data when using the distance, unlike many alternatives to Σ and Σ^{-1} (see Section 2.5);
- Quick methods of computation via elementary symmetric functions and sampling methods.

Some of the limitations of this method include:

- Instability if k is chosen too high in the elementary symmetric function representation. This is in line with other polynomial-based methods, where it is not recommended to use a high degree [101];
- Not always achieving perfect whitening in Section 3.5.3. The suggestion of iterative whitening methods can help to alleviate this issue;
- Calculating the full distance through simplex volumes is computationally expensive. Using elementary symmetric functions or subsampling methods can reduce the cost of computing the distance.

Two different methods for constructing the simplicial distances were provided, depending on parameter choices. For any choice of the parameter δ, the distance can be constructed using volumes of simplices (Section 3.2.1). When $\delta = 2$, a method using elementary sym-

metric functions and polynomials can be used to calculate the distances (Section 3.2.2), making the distance more accessible and time-efficient. This method can be used to represent the distance through quadratic forms, allowing moments and distribution properties to be found (Section 3.4). If the parameter δ is chosen not to be 2, a method of sampling the simplices is given to improve the computational expense and time spent finding the distance (Section 3.2.3). This is shown to have very minimal effect on the distance found, making $\delta \neq 2$ a viable parameter choice for the simplicial distances.

A discussion around the selection of parameters is given in Section 3.3. The parameter k controls the dimension of the simplices (or degree of the polynomial, in the $\delta = 2$ case). When $k = 1$, the distance is proportional to the Euclidean distance. As k increases towards r (the rank of the dataset), the behaviour of the distances become more and more similar to the behaviour of the Mahalanobis distances. However, in later examples, numerical instability starts to cause issues for high values of k, particularly when used in conjunction with the elementary symmetric function method of calculation. It is therefore recommended to use a low value of k, which accounts for correlations in a similar way to the Mahalanobis distance (if it were to exist), but will not be hindered by instability issues.

The parameter δ is an exponent in the simplicial distances, and can be likened to the choice of δ in the ℓ_δ distances. While $\delta = 2$ provides a distance similar to the very popular Euclidean distance, it is known that ℓ_1 distances can be more robust in high dimensions. This is corroborated by the investigation into relative contrasts in Section 3.3.2. Therefore, $\delta = 1$ may be a good parameter choice in examples where outliers are present, or the dimension is very high. A further consideration in the choice of parameter δ is the time expense, which is also investigated in Section 3.3.2. It is shown that for low dimensions, using the functional approach for $\delta = 2$ is quicker than calculating distances directly through simplex volumes. However, as the dimensions increase, this time advantage decreases, particularly if the sample size of simplices is well controlled.

Various numerical examples that highlight the potential applications of the simplicial distances are given in Section 3.5. The robustness of the distances using $\delta = 1$ is shown in an outlier detection setting. The simplicial distances outperform the Mahalanobis and Euclidean distance in certain clustering examples with degeneracy and rotation in high dimensions. There are examples illustrating how the polynomial matrix found when using

the $\delta = 2$ parameter can be used to whiten data, and the concept of iterative data whitening is briefly introduced, which will be further considered in Section 5.4.

Overall, this chapter has demonstrated that the simplicial distance measure is a robust yet flexible method of measuring proximity in multivariate data. It performs well in cases where existing methods like the Mahalanobis distance does not, and it does not impose structural assumptions in its construction, unlike many other alternative methods to the Mahalanobis distance.

The next chapter of this thesis will consider a novel distance measure with similar benefits, inspired by some of the results produced by the simplicial distances.

Chapter 4

Minimal-Variance Distances

The research presented in this chapter forms part of a publication I have co-authored [85], entitled **Simplicial and Minimal-Variance Distances in Multivariate Data Analysis**, published in *Journal of Statistical Theory and Practice*, available at https://doi.org/10.1007/s42519-021-00227-7.

The differences between the published manuscript and the contents of this chapter are:

- This chapter only includes research on minimal-variance distances. Research regarding simplicial distances is given in Chapter 3;
- This chapter details construction of the minimal-variance distances through polynomials and weighted linear regression, whereas the publication only considers the polynomial method;
- A more general approach to the constraint imposed on the distance measure is proposed in this chapter. Investigations into the effects caused by changing the parameter in the constraint are given;
- Further applications of the minimal-variance distances are considered here. This includes an outlier labelling example and comparisons to other distances in HDLSS settings;
- Two alternative constraints for use within the minimal-variance distances are suggested in this chapter.

The aims of the research presented in this chapter are:

- To produces a family of distance measures which perform similarly to the Mahalanobis distance in multivariate data, but are not subject to the issues of degeneracy faced by the Mahalanobis distance (see Section 2.2.2);
- To compute distance measures which minimize the overall variance of the distances produced, motivated by trends shown by the simplicial distances in Chapter 3;
- To illustrate the flexibility of the minimal-variance distances, through choices of parameters and different forms of constraints;
- To compare the minimal-variance distances to the simplicial distances, as well as the Euclidean and Mahalanobis distances.

4.1 Introduction

Let $X = \{x_1, x_2, \ldots, x_N\} \in \mathbb{R}^{d \times N}$ be a d-dimensional set of N points, with mean μ and covariance matrix Σ. This chapter presents a new family of distance measures, the so-called 'minimal-variance' distances. The motivation behind these distance measures is driven by trends shown in images such as Figure 4.1, which was first presented in Section 3.3.1. This figure shows the cumulative distribution functions (CDFs) of the simplicial distances, discussed in Chapter 3, for various parameters k. The distance measures that perform 'best' (i.e. similarly to the Mahalanobis distance) are those that produce distances with the smallest variance. The aim is therefore to find a matrix A that minimizes the variance of the quadratic form $(x_i - \mu)^\top A(x_i - \mu)$ for all $x_i \in X$.

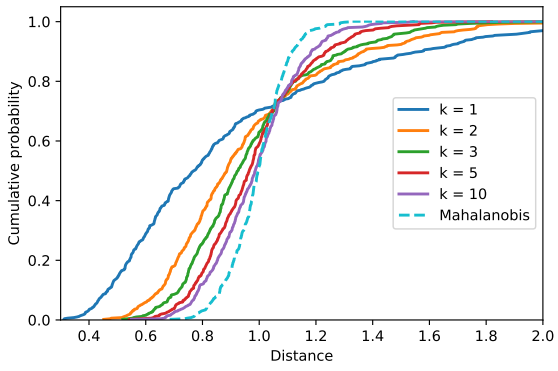


Figure 4.1: CDFs of simplicial distances measured in a 50-dimensional dataset with different values of k. This figure is first seen in Section 3.3.1.

4.2 Constructing the minimal-variance distances

In Section 4 of [85], the minimal-variance distances are constructed using polynomials in the covariance matrix. A Lagrange multiplier method is used to find the polynomial coefficients that will minimize the variance of the distances, while abiding by a constraint which ensures the minimal-variance matrix A behaves similarly to an inverse. In Section 4.2.1, this method is generalized to use alternative constraints. Section 4.2.2 offers another method of construction, using weighted linear regression to find the distances.

4.2.1 Construction through polynomials

Let X be defined as in Section 4.1. For a $d \times d$ symmetric matrix A, define the quadratic form

$$\rho_A^2(x, X) = (x - \mu)^\top A(x - \mu).$$

If A is positive definite, then $\rho_A(x, X)$ can be considered as a generalized distance from a point $x \in \mathbb{R}^d$ to the set of points X. Assuming for now that Σ is full-rank, the minimal-variance distance constructs a matrix polynomial A in Σ of degree $k \leq d$, where k is a user-defined parameter. Like the simplicial distances, this forms a spectrum of distances which includes the Mahalanobis distance when the degree parameter k is equal to d. Under the assumptions $x \sim \mathcal{N}_d(\mu, \Sigma)$ and $X \sim \mathcal{N}_d(\mu, \Sigma)$ (x does not necessarily belong to X), the first two moments of $\rho_A^2(x, X)$ are given by:

$$\begin{aligned} \mathrm{E}\left(\rho_A^2(x, X)\right) &= \mathrm{trace}(A\Sigma), \\ \mathrm{Var}\left(\rho_A^2(x, X)\right) &= 2\mathrm{trace}\left([A\Sigma]^2\right), \end{aligned} \tag{4.1}$$

see Appendix A.5 for details on the derivation of these moments.

Construction using a generalized constraint

For a chosen degree parameter $k \in \mathbb{Z}$ with $k \leq d$, let the matrix A be the matrix that minimizes the variance $\mathrm{Var}\left(\rho_A^2(x, X)\right)$ subject to the constraint

$$\mathrm{trace}\left(A\Sigma^\alpha\right) = \mathrm{trace}\left(\Sigma^{\alpha-1}\right), \tag{4.2}$$

where $\alpha \in \mathbb{R}$. This constraint forces the matrix A to behave similarly to Σ^{-1} (if Σ^{-1} exists). The effect that different values of α have on the approximation to Σ^{-1} will be considered

in Section 4.3.2. The constraint (4.2) differs to the constraint given in [85], where (4.2) only takes the value $\alpha = 1$. The method with a general value of α will be outlined here, and a summary of the results with the case $\alpha = 1$ is given at the end of this subsection. Furthermore, Equation (4.2) is not the only constraint that can force A to have similar properties to Σ^{-1}; different forms of constraints are discussed in Section 4.6 that were not considered in [85].

Define A_k to be a polynomial in Σ of degree $k-1$:

$$A_k = \sum_{i=0}^{k-1} \theta_i \Sigma^i = \theta_\alpha^\top \Sigma_{(k)}\,, \tag{4.3}$$

where $\theta_\alpha = (\theta_0, \theta_1, \ldots, \theta_{k-1})^\top$ is defined to be the vector of k coefficients to be found, and $\Sigma_{(k)} = \left(\Sigma^0, \Sigma^1, \ldots, \Sigma^{k-1}\right)^\top$. Define the Vandermonde matrix

$$V = \left(\lambda_j^{i+1}\right)_{\substack{j=1,\ldots,d,\\ i=0,\ldots,k-1}} = \begin{pmatrix} \lambda_1 & \lambda_1^2 & \ldots & \lambda_1^k \\ \lambda_2 & \lambda_2^2 & \ldots & \lambda_2^k \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_d & \lambda_d^2 & \ldots & \lambda_d^k \end{pmatrix}, \tag{4.4}$$

where $\lambda_1, \lambda_2, \ldots, \lambda_d$ are the eigenvalues of Σ. From Equations (4.1) and (4.3), the variance of the quadratic form can be written as:

$$\mathrm{Var}\left(\rho^2_{A_k}(x, X)\right) = 2\mathrm{trace}\left([A_k\Sigma]^2\right) = 2\mathrm{trace}\left(\sum_{i=0}^{k-1}\theta_i\Sigma^{i+1}\sum_{j=0}^{k-1}\theta_j\Sigma^{j+1}\right) = 2\theta_\alpha^\top V^\top V\theta_\alpha\,. \tag{4.5}$$

Let $S_\alpha = \mathrm{trace}\,(\Sigma^\alpha)$, and define

$$S_{(\alpha,k)} = (S_\alpha, S_{\alpha+1}, \ldots, S_{\alpha+k-1})^\top = \left(\mathrm{trace}\,(\Sigma^\alpha), \mathrm{trace}\left(\Sigma^{\alpha+1}\right), \ldots, \mathrm{trace}\left(\Sigma^{\alpha+k-1}\right)\right)^\top.$$

Using Equation (4.3),

$$\mathrm{trace}(A_k\Sigma^\alpha) = \mathrm{trace}\left(\sum_{i=0}^{k-1}\theta_i\Sigma^{i+\alpha}\right) = \theta_\alpha^\top \mathrm{trace}\left(\Sigma_{(k)}\Sigma^\alpha\right) = \theta_\alpha^\top S_{(\alpha,k)}$$

and so the constraint (4.2) can now be written in the form

$$\theta_\alpha^\top S_{(\alpha,k)} = S_{\alpha-1}\,. \tag{4.6}$$

The following theorem produces the optimal vector of coefficients θ_α that gives the minimal-variance distances subject to the constraint (4.6) with parameter k.

Theorem 2. *Define υ to be the number of nonzero unique eigenvalues of the positive definite covariance matrix Σ. For $k \leq \upsilon$, the matrix polynomial*

$$A_k = \sum_{i=0}^{k-1} \theta_i \Sigma^i = \theta_\alpha^\top \Sigma_{(k)}$$

that minimizes $Var\left(\rho^2_{A_k}(x,X)\right)$ subject to the constraint $\theta_\alpha^\top S_{(\alpha,k)} = S_{\alpha-1}$ has coefficients

$$\theta^*_\alpha = \frac{S_{\alpha-1}}{S^\top_{(\alpha,k)}(V^\top V)^{-1} S_{(\alpha,k)}} (V^\top V)^{-1} S_{(\alpha,k)}. \tag{4.7}$$

Proof. Form a Lagrange function to minimize $\frac{1}{4}\text{Var}\left(\rho^2_{A_k}(x,X)\right) = \frac{1}{2}\theta_\alpha^\top V^\top V \theta_\alpha$, subject to the constraint (4.6). The Lagrange function with multiplier ω is given by

$$\mathcal{L}(\theta_\alpha, \omega) = \frac{1}{2}\theta_\alpha^\top V^\top V \theta_\alpha - \omega\left(\theta_\alpha^\top S_{(\alpha,k)} - S_{\alpha-1}\right). \tag{4.8}$$

Minimize (4.8) by differentiating with respect to θ_α and setting the result to 0, which gives:

$$V^\top V \theta_\alpha = \omega S_{(\alpha,k)},$$

and therefore θ_α is found to be

$$\theta_\alpha = \omega\left(V^\top V\right)^{-1} S_{(\alpha,k)}. \tag{4.9}$$

Re-writing the constraint $\theta_\alpha^\top S_{(\alpha,k)} = S_{\alpha-1}$ using Equation (4.9) gives

$$\omega S^\top_{(\alpha,k)}(V^\top V)^{-1} S_{(\alpha,k)} = S_{\alpha-1}. \tag{4.10}$$

Let the scalar ω be written as $\omega_{\alpha,k}$ to show the dependency on parameters α and k. Then, rearrange (4.10) to find $\omega_{\alpha,k}$:

$$\omega = \omega_{\alpha,k} = \frac{S_{\alpha-1}}{S^\top_{(\alpha,k)}(V^\top V)^{-1} S_{(\alpha,k)}} \tag{4.11}$$

and therefore the solution to (4.8) is given by Equation (4.7). □

By substituting (4.7) into Equation (4.5), an expression for the variance of the minimal-variance distance using A_k can be found:

$$\text{Var}\left(\rho^2_{A_k}(x,X)\right) = \frac{2S^2_{\alpha-1}}{S^\top_{(\alpha,k)}(V^\top V)^{-1} S_{(\alpha,k)}}.$$

Construction using a constraint with $\alpha = 1$

In [85], the constraint is given with $\alpha = 1$, rather than the generalized version discussed above. Different choices of the parameter α will be studied in Section 4.3.2, where it will be shown that using $\alpha = 1$ gives an unbiased estimator for Σ^{-1}, and is therefore a natural choice. Using $\alpha = 1$, the constraint (4.6) becomes

$$\operatorname{trace}(A_k \Sigma) = d, \tag{4.12}$$

which can be written in the form

$$\theta_1^\top S_{(1,k)} = d.$$

Substituting $\alpha = 1$ into Equation (4.7), the vector of coefficients that provides the minimal-variance distance for this parameter is given as:

$$\theta_1^* = \frac{d}{S_{(1,k)}^\top (V^\top V)^{-1} S_{(1,k)}} (V^\top V)^{-1} S_{(1,k)}. \tag{4.13}$$

The variance of the minimal-variance distance with $\alpha = 1$ is therefore

$$\operatorname{Var}\left(\rho_{A_k}^2(x, X)\right) = \frac{2d^2}{S_{(1,k)}^\top (V^\top V)^{-1} S_{(1,k)}}. \tag{4.14}$$

4.2.2 Construction through weighted linear regression

In this section, a different approach to constructing the minimal-variance polynomials is used. The aim is to find a polynomial of degree k that approximates a function of the inverse eigenvalues, and this is done using weighted linear regression. As before, the method is first considered using a general value of the parameter α, and then the specific case with $\alpha = 1$ is detailed.

General values of the parameter α

Consider the problem from the perspective of weighted linear regression: approximate the inverse covariance matrix Σ^{-1} by finding the spectral polynomial $P_k(\lambda)$ such that $P_k(\lambda_j)$ approximates $\frac{1}{\lambda_j^\beta}$ for $j = 1, \ldots, d$, where β is a function of α that will be determined shortly. Define the regression model

$$\lambda_j^{-\beta} = \sum_{i=0}^{k-1} \theta_i \lambda_j^i + \varepsilon(\lambda_j) \quad (j = 1, \ldots, d), \tag{4.15}$$

where the approximation errors $\varepsilon(\lambda_j)$ are assumed to be uncorrelated random values with zero mean and variances $\sigma^2(\lambda_j) = 1/w(\lambda_j)$, where $w : (0,\infty) \to (0,\infty)$ is a weight function to be chosen. The regression model (4.15) can be written in the matrix notation as $Y = L\theta_\alpha + \varepsilon$, where

$$Y = \begin{pmatrix} \lambda_1^{-\beta} \\ \lambda_2^{-\beta} \\ \vdots \\ \lambda_d^{-\beta} \end{pmatrix}, \quad L = \left(\lambda_j^i\right)_{\substack{j=1,\ldots,d,\\ i=0,\ldots,k-1}} = \begin{pmatrix} 1 & \lambda_1 & \ldots & \lambda_1^{k-1} \\ 1 & \lambda_2 & \ldots & \lambda_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_d & \ldots & \lambda_d^{k-1} \end{pmatrix}, \tag{4.16}$$

$\theta_\alpha = (\theta_0, \theta_1, \ldots, \theta_{k-1})^\top$ is the vector of coefficients sought and $\varepsilon = (\varepsilon(\lambda_1), \ldots, \varepsilon(\lambda_d))^\top$ is the vector of errors. By assumption, $E(\varepsilon) = 0$ and the covariance matrix of ε is

$$W = D(\varepsilon) = \operatorname{diag}\left(\frac{1}{w(\lambda_1)}, \ldots, \frac{1}{w(\lambda_d)}\right) \tag{4.17}$$

for some given weight function w. Then $W^{-1} = \operatorname{diag}(w(\lambda_1), \ldots, w(\lambda_d))$. The weighted least squares estimate (WLSE) $\hat{\theta}_\alpha$ of θ_α is given by

$$\hat{\theta}_\alpha = \left(L^\top W^{-1} L\right)^{-1} L^\top W^{-1} Y = \left(L^\top W^{-\frac{1}{2}} W^{-\frac{1}{2}} L\right)^{-1} L^\top W^{-\frac{1}{2}} W^{-\frac{1}{2}} Y. \tag{4.18}$$

To make $\hat{\theta}_\alpha$ correspond to θ^*_α from Equation (4.7), define the weight function w to be $w(\lambda) = \lambda^2$. This gives

$$W^{-\frac{1}{2}} L = \begin{pmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \lambda_d \end{pmatrix} \begin{pmatrix} 1 & \lambda_1 & \ldots & \lambda_1^{k-1} \\ 1 & \lambda_2 & \ldots & \lambda_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_d & \ldots & \lambda_d^{k-1} \end{pmatrix} = \begin{pmatrix} \lambda_1 & \lambda_1^2 & \ldots & \lambda_1^k \\ \lambda_2 & \lambda_2^2 & \ldots & \lambda_2^k \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_d & \lambda_d^2 & \ldots & \lambda_d^k \end{pmatrix} = V.$$

It is therefore possible to replace $W^{-\frac{1}{2}} L$ in Equation (4.18) with the matrix V, which gives $\hat{\theta}_\alpha = (V^\top V)^{-1} V^\top W^{-\frac{1}{2}} Y$. To make $\hat{\theta}_\alpha$ equal to θ^*_α from Equation (4.7), the vector $Y = (y_1, \ldots, y_d)^\top$ needs to be defined such that $V^\top W^{-\frac{1}{2}} Y = S_{(\alpha,k)}$. This can be written in summation form as

$$\sum_{i=1}^{d} \lambda_i^{j+1} y_i = \sum_{i=1}^{d} \lambda_i^{\alpha+j-1} \tag{4.19}$$

for $j = 1, \ldots, k$. For Equation (4.19) to hold, let $y_i = \lambda_i^{\alpha-2}$. Therefore

$$Y = \left(\lambda_1^{\alpha-2}, \lambda_2^{\alpha-2}, \ldots, \lambda_d^{\alpha-2}\right)^\top, \tag{4.20}$$

which gives $\beta = 2 - \alpha$. It is therefore possible to write $\hat{\theta}_\alpha$ as:

$$\hat{\theta}_\alpha = \left(V^\top V\right)^{-1} V^\top W^{-\frac{1}{2}} Y = \left(V^\top V\right)^{-1} S_{(\alpha,k)}.$$

Redefine $\hat{\theta}_\alpha$ to include a constant that ensures the minimal-variance constraint (4.2) is satisfied. Let

$$\hat{\theta}_\alpha = \omega_{\alpha,k}\left(V^\top V\right)^{-1} S_{(\alpha,k)}. \tag{4.21}$$

To find the optimal value of the constant $\omega_{\alpha,k}$, substitute $\hat{\theta}_\alpha$ in place of θ_α in the constraint (4.6): $\hat{\theta}_\alpha S_{(\alpha,k)} = S_{\alpha-1}$. Replacing $\hat{\theta}_\alpha$ with the right-hand side of Equation (4.21) and rearranging for $\omega_{\alpha,k}$ gives

$$\omega_{\alpha,k} = \frac{S_{\alpha-1}}{S_{(\alpha,k)}^\top (V^\top V)^{-1} S_{(\alpha,k)}} = \frac{S_{\alpha-1}}{Y^\top W^{-1} L (L^\top W^{-1} L)^{-1} L^\top W^{-1} Y}.$$

Thus, the vector $\hat{\theta}_\alpha$ is given by

$$\begin{aligned}\hat{\theta}_\alpha &= \frac{S_{\alpha-1}}{Y^\top W^{-1} L (L^\top W^{-1} L)^{-1} L^\top W^{-1} Y}\left(L^\top W^{-1} L\right)^{-1} L^\top W^{-1} Y \\ &= \frac{S_{\alpha-1}}{S_{(\alpha,k)}^\top (V^\top V)^{-1} S_{(\alpha,k)}} (V^\top V)^{-1} S_{(\alpha,k)},\end{aligned}$$

which is exactly the vector given in Equation (4.7). The WLSE for general α gives the matrix polynomial

$$A_k = \sum_{i=0}^{k-1} \theta_i \Sigma^i$$

which estimates $f(\lambda_j) = \frac{1}{\lambda_j^{2-\alpha}}$ for all eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_d$ of Σ.

Using the parameter $\alpha = 1$

In Section 4.3.2, the effects of different values of the parameter α on the minimal-variance distances will be considered. However, using $\alpha = 1$ is a sensible choice for many reasons. As indicated above, the minimal-variance polynomial estimates $f(\lambda_j) = \frac{1}{\lambda_j^{2-\alpha}}$. Using $\alpha = 1$ is therefore the only value to provide an unbiased estimator for Σ^{-1} (although corrections for other values of α are given in Section 4.3.2).

Using $\alpha = 1$ produces the constraint (4.12), and the formula for the vector of coefficients is given by

$$\begin{aligned}\hat{\theta}_1 &= \frac{d}{Y^\top W^{-1} L (L^\top W^{-1} L)^{-1} L^\top W^{-1} Y}\left(L^\top W^{-1} L\right)^{-1} L^\top W^{-1} Y \\ &= \frac{d}{S_{(1,k)}^\top (V^\top V)^{-1} S_{(1,k)}} (V^\top V)^{-1} S_{(1,k)}.\end{aligned}$$

4.2.3 Equivalence to the Mahalanobis distance

The following theorem shows that if Σ is invertible and all eigenvalues are different, then $A_d = \Sigma^{-1}$ when $\alpha = 1$. This indicates that the minimal-variance distances with $\alpha = 1$ and $k = d$ are equal to the Mahalanobis distance.

Theorem 3. *Assume that all eigenvalues $\lambda_1, \ldots, \lambda_d$ of the $d \times d$ matrix Σ are positive and pairwise different. Then for $\alpha = 1$, $A_d = \Sigma^{-1}$, where A_d is defined by Equations (4.3) and (4.13) with $k = d$.*

Proof. Let the matrices L and W be defined as in (4.16) and (4.17), respectively, with the weight function in W defined as $w(\lambda) = \lambda^2$. Let the vector Y be defined as in Equation (4.20). Recall from Equations (4.18) and (4.21) that, for $\alpha = 1$, the coefficient vector can be written as $\theta_1 = \omega_{1,d}\left(L^\top W^{-1} L\right)^{-1} L^\top W^{-1} Y$, and so $\left(L^\top W^{-1} L\right)^{-1} L^\top W^{-1} Y = \frac{1}{\omega_{1,d}}\theta_1$. Since $k = d$, and $\frac{1}{\omega_{1,d}}\theta_1$ is the WLSE,

$$\frac{1}{\omega_{1,d}}\theta_1^\top \Sigma_{(d)} = \frac{1}{\omega_{1,d}} A_d = \Sigma^{-1}$$

exactly. Therefore, if $\omega_{1,d} = 1$, this implies that $A_d = \Sigma^{-1}$. Recall that

$$\omega_{1,d} = \frac{d}{S_{(1,d)}^\top (V^\top V)^{-1} S_{(1,d)}}.$$

As $k = d$ and all eigenvalues $\lambda_i, i = 1, \ldots, d$, are positive and pairwise different, the $d \times d$ Vandermonde matrix V is non-degenerate. Following the methodology of [100], denote $(V^{-1})^\top = B = \left(b_{ij}\right)_{i,j=1}^{d}$. From the definition of an inverse, $BV = I$, which can be written as:

$$BV = \sum_{t=1}^{d} b_{it}\lambda_j^t = \begin{cases} 1 & i = j, \\ 0 & i \neq j. \end{cases}$$

The tth entry of the vector $S_{(1,k)}$ is equal to $\sum_{1=1}^{d} \lambda_j^t$. Thus, $(V^{-1})^\top S_{(1,k)} = \mathbb{1}$, where $\mathbb{1}$ is a d-dimensional vector of ones. This gives

$$\omega_d = \frac{d}{S_{(1,d)}^\top V^{-1}(V^{-1})^\top S_{(1,d)}} = \frac{d}{\mathbb{1}^\top \mathbb{1}} = 1.$$

□

The theorem can easily be generalized to the case when the multiplicities of the eigenvalues are arbitrary, or when there are zero eigenvalues. Recall that υ is the number of

unique nonzero eigenvalues. In the case where all the eigenvalues are nonzero, $A_\upsilon = \Sigma^{-1}$. If there are eigenvalues equal to zero, the true inverse of Σ does not exist, but A_υ is a generalized inverse according to the definition in [29]. In these cases, use the Moore-Penrose pseudoinverse of V in place of V^{-1}.

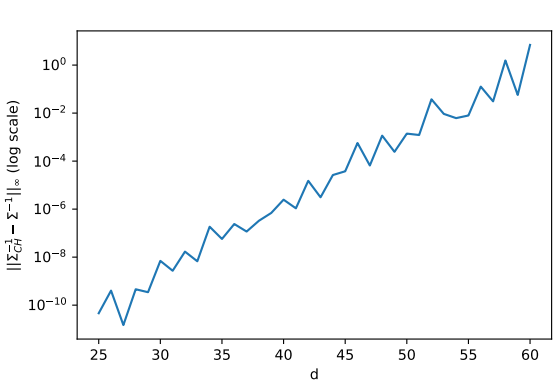
4.2.4 Comparison to characteristic polynomial inversion

It is known that methods that take advantage of matrix multiplications, such as polynomial functions of matrices, are usually much faster than directly inverting a matrix when performed on modern computers [101]. For a full-rank matrix Σ with characteristic polynomial $\det(\Sigma - xI) = x^d + c_1 x^{d-1} + \cdots + c_d$, the Cayley-Hamilton theorem states that the inverse of Σ can be calculated by:

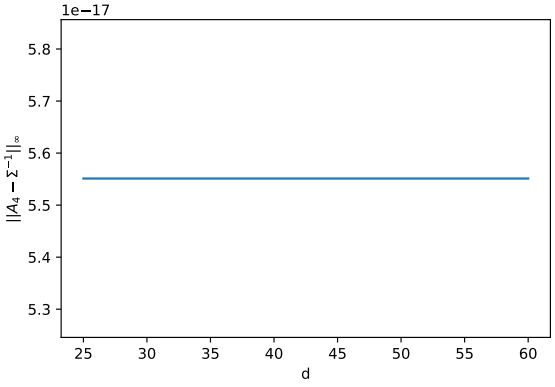
$$\Sigma_{CH}^{-1} = -\frac{1}{c_d}\left(\Sigma^{d-1} + \sum_{i=1}^{d-1} c_i \Sigma^{d-i-1}\right).$$

However, in Section 4.8 of [101] the authors write that using the characteristic polynomial to compute a function (such as the inverse) of a matrix is not recommended, as the characteristic polynomial of a matrix cannot be reliably computed in floating point arithmetic. The authors illustrate this with a numerical example. Let $\Sigma = 3I$, where I is the d-dimensional identity matrix. The authors then compute the inverse Σ_{CH}^{-1} using the Cayley-Hamilton theorem for $d \in [25, 60]$. Figure 4.2a recreates Figure 4.2 of [101], where the infinity norm between Σ_{CH}^{-1} and the true inverse $\Sigma^{-1} = \frac{1}{3}I$ is plotted. As d increases, the error becomes more and more pronounced.

Figure 4.2b performs the same exercise using the minimal-variance polynomial with $k = 4$. The degree of this polynomial is clearly much lower than the degree of the characteristic polynomial, and therefore does not suffer from the same numerical instability and error as Σ_{CH}^{-1}. Therefore, the minimal-variance polynomial is not only cheaper than using the characteristic polynomial, but is also much more stable. It will be shown in the numerical examples given in Section 4.5 that using low values of k is recommended for performance, and as such using a polynomial method to compute an approximation of the inverse is justified.



(a) Characteristic polynomial method



(b) Minimal-variance method, $k = 4$

Figure 4.2: For increasing d, the infinity norm is plotted between the true inverse of $\Sigma = 3I$ and (a) the inverse calculated through the characteristic polynomial; (b) the inverse calculated through the minimal-variance polynomial with $k = 4$. Note the scale is $1e-17$ in (b).

4.3 Effects of the parameter α on the minimal-variance distances

As previously alluded to, the parameter choice $\alpha = 1$ is the only value of α that provides an unbiased estimator to Σ^{-1} in the minimal-variance distances. In this section, the effects of different values of the parameter α are considered. First, comparisons to the best linear unbiased estimator are given by comparing the variances of the distances produced with different values of α. The effect of α on the scalar value $\omega_{\alpha,k}$ is then explored in the context of recovering the desired eigenvalues. A correction for the biasedness of the estimator with $\alpha \neq 1$ is also discussed.

4.3.1 Comparison to the best linear unbiased estimator

An unbiased estimator $\hat{\tau}$ of a parameter τ is called the best linear unbiased estimator (BLUE) if, for all unbiased estimators $\tilde{\tau}$,

$$\text{Var}(\hat{\tau}) \leq \text{Var}(\tilde{\tau}).$$

That is, the BLUE of τ has the minimum variance of all unbiased estimators of τ. Typically, the BLUE is found using the weighted least squares method, as in Equation (4.18),

using weights that ensure equal variance of the errors (where the errors are as defined in Equation (4.15)).

The estimators proposed for θ_α are weighted least squares estimators scaled by some constant $\omega_{\alpha,k}$. This scaling ensures that the estimators satisfy the constraint (4.2) for varying values of α. Using this scaling constant has an effect on the bias and the variance of the estimators. Therefore, values of α can be chosen such that a slightly biased estimator is obtained, with smaller variance than the BLUE.

For the various estimators of θ_α considered so far, a correspondence can be made to the BLUE of θ_α, denoted $\hat{\theta}_{\text{BLUE}}$, if $\omega_{\alpha,k} = 1$. For instance, in Section 4.2.1, the coefficients of the polynomial are given by

$$\hat{\theta}_\alpha = \omega_{\alpha,k}\left(L^\top W^{-1} L\right)^{-1} L^\top W^{-1} Y,$$

which, if $\omega_{\alpha,k} = 1$, is exactly the BLUE of the vector of coefficients θ_α. Theorem 3 shows that for full rank Σ, when $\alpha = 1$ and $k = d$, $\omega_{1,d} = 1$.

Consider the Mean Squared Error (MSE) of the estimators as a function of the constant $\omega_{\alpha,k}$, and compare it to the MSE of the BLUE of θ_α. Let $\tilde{\theta}_\alpha(\lambda)$ be the estimator, and let $\hat{\theta}_{\text{BLUE}}(\lambda)$ be the BLUE of θ_α. Then it is always the case that

$$\tilde{\theta}_\alpha(\lambda) = \omega_{\alpha,k}\hat{\theta}_{\text{BLUE}}(\lambda).$$

The MSE of an estimator can be written as the sum of the variance and the squared bias of the estimator. Define $\text{Var}(\omega_{\alpha,k})$ and $\text{Bias}(\omega_{\alpha,k})$ to be the variance and squared bias, respectively, of the estimator $\tilde{\theta}_\alpha(\lambda) = \omega_{\alpha,k}\hat{\theta}_{\text{BLUE}}(\lambda)$ as follows:

$$\text{Var}(\omega_{\alpha,k}) = E\left[\int \left(\tilde{\theta}_\alpha(\lambda) - E\left[\tilde{\theta}_\alpha(\lambda)\right]\right)^2\right] d\lambda = \omega_{\alpha,k}^2 E\left[\int \left(\hat{\theta}_{\text{BLUE}}(\lambda) - E\left[\hat{\theta}_{\text{BLUE}}(\lambda)\right]\right)^2\right] d\lambda,$$

$$\text{Bias}(\omega_{\alpha,k}) = \int \left(\frac{1}{\lambda} - \omega_{\alpha,k}\frac{1}{\lambda}\right)^2 d\lambda = (1-\omega_{\alpha,k})^2 \int \frac{1}{\lambda} d\lambda.$$

Let $\text{Var}_{\text{BLUE}} = \text{Var}(1)$ denote the variance of the BLUE of the coefficient vector θ_α. It is therefore clear that $\text{Var}(\omega_{\alpha,k}) = \omega_{\alpha,k}^2 \text{Var}_{\text{BLUE}}$. The MSE of the estimator with the constant $\omega_{\alpha,k}$ can then be written as

$$\text{MSE}(\omega_{\alpha,k}) = \text{Var}(\omega_{\alpha,k}) + \text{Bias}(\omega_{\alpha,k}) = \omega_{\alpha,k}^2 \text{Var}_{\text{BLUE}} + (1-\omega_{\alpha,k})^2 \Lambda,$$

where $\Lambda = \frac{1}{d}\sum_{i=1}^{d} \lambda_i^2$.

As an example of a constant $\omega_{\alpha,k}$ that produces a biased but low-variance estimator, define

$$\breve{\omega}_{\alpha,k} = \frac{\Lambda}{\text{Var}_{\text{BLUE}} + \Lambda},$$

such that

$$\text{MSE}(\breve{\omega}_{\alpha,k}) = \frac{\Lambda \text{Var}_{\text{BLUE}}}{\Lambda + \text{Var}_{\text{BLUE}}} < \text{Var}_{\text{BLUE}} = \text{MSE}(1).$$

Therefore, choosing $\omega_{\alpha,k} = \breve{\omega}_{\alpha,k}$ gives an estimator $\tilde{\theta}_\alpha(\lambda)$ with smaller MSE than the BLUE $\hat{\theta}_{\text{BLUE}}(\lambda)$. This is desirable, but does make the estimator $\tilde{\theta}_\alpha(\lambda)$ a biased estimator. At the end of Section 4.2.1, it is shown that the polynomials with parameter α approximate $\Sigma^{2-\alpha}$. Several numerical examples are considered in Section 4.3.2, where this biasedness is discussed, along with an approach to counteract it.

4.3.2 Numerical examples on the effects of α

This section explores the impact of different values of α on the minimal-variance distances. As seen in Section 4.2.3 and Section 4.3.1, using $\alpha = 1$ and $k = d$ produces the BLUE of θ_α, which means using these parameters recovers the Mahalanobis distance exactly, if Σ is not singular (see Theorem 3). Using $\alpha > 1$ gives a biased estimator for θ_α, but with the potential to find distances $\rho_{A_k}(x, X)$ with smaller variance. The following numerical examples will explore this concept, and discuss methods to correct for the biasedness when using $\alpha \neq 1$.

Example 1

The effect of different values of α will first be illustrated with a small toy example, with $d = 6$. Let Σ be a diagonal matrix with diagonal entries $[6, 5, 4, 3, 2, 1]$. Table 4.1a shows the values of $\omega_{\alpha,k}$ (given by Equation (4.11)) for different values of α and k. When $\alpha = 1$ and $k = 6$, A_6 is a diagonal matrix with diagonal entries $[0.167, 0.2, 0.25, 0.33, 0.5, 1]$ which is exactly equal to Σ^{-1} (noting that 0.167 and 0.33 have been rounded).

Recall from Section 4.3.1 that $\text{Var}(\omega_{\alpha,k}) = \omega_{\alpha,k}^2 \text{Var}_{\text{BLUE}}$. Table 4.1a shows that when $\alpha > 1$, $\omega_{\alpha,k} < \omega_{1,k}$ for each k considered. That is, when $\alpha > 1$, an estimator with lower variance than the BLUE is produced, which is corroborated by Table 4.1b.

However, the estimators of θ_α approximate $\Sigma^{\alpha-2}$. When $\alpha = 1$, this means the estimators approximate Σ^{-1}. If $\alpha = 1.05$ is used, for example, the estimators are actually approxi-

α \ k	2	3	4	5	6
1.00	1.055	1.013	1.002	1.000	1.000
1.01	1.041	1.001	0.991	0.989	0.989
1.02	1.028	0.990	0.980	0.978	0.978
1.05	0.988	0.955	0.947	0.946	0.945

(a) $\omega_{\alpha,k}$

α \ k	2	3	4	5	6
1.00	12.659	12.156	12.026	12.002	12.000
1.01	12.635	12.149	12.025	12.002	12.000
1.02	12.610	12.141	12.022	12.000	11.998
1.05	12.535	12.115	12.010	11.991	11.989

(b) $\mathrm{Var}\left(\rho_{A_k}(x, X)\right)$ with α

Table 4.1: Example 1: The values (3.d.p) of (a) $\omega_{\alpha,k}$ and (b) the variance of the minimal-variance distances with parameters α and k.

α	A_6	$\Sigma^{\alpha-2}$
1.00	[0.167, 0.2, 0.25, 0.333, 0.5, 1.0]	[0.167, 0.2, 0.25, 0.333, 0.5, 1.0]
1.01	[0.168, 0.201, 0.251, 0.333, 0.498, 0.989]	[0.17, 0.203, 0.253, 0.337, 0.503, 1.0]
1.02	[0.169, 0.202, 0.251, 0.333, 0.496, 0.978]	[0.173, 0.207, 0.257, 0.341, 0.507, 1.0]
1.05	[0.172, 0.205, 0.253, 0.333, 0.489, 0.945]	[0.182, 0.217, 0.268, 0.352, 0.518, 1.0]

Table 4.2: Example 1: Diagonal entries of A_6 and $\Sigma^{\alpha-2}$ for different values of α.

mating $\Sigma^{1.05-2} = \Sigma^{-0.95}$. Table 4.2 shows the diagonal entries of A_6 for different values of α, as well as the diagonal entries for $\Sigma^{\alpha-2}$.

The biasedness of the approximation to Σ^{-1} can be corrected for as follows. Given that the estimators approximate $\Sigma^{\alpha-2}$, this approximation should be expontentiated by $1/(2-\alpha)$ by the following logic:

$$(A_k)^{\frac{1}{2-\alpha}} \approx \left(\Sigma^{\alpha-2}\right)^{\frac{1}{2-\alpha}} = \Sigma^{-1}.$$

To investigate how well this approximation works, consider the value of $(A_k)^{\frac{1}{2-\alpha}}\Sigma$. If $(A_k)^{\frac{1}{2-\alpha}}$ is a good approximation to Σ^{-1}, $(A_k)^{\frac{1}{2-\alpha}}\Sigma$ should equal the identity matrix. Table 4.3 shows the diagonal entries of $A_6\Sigma$ and $(A_6)^{\frac{1}{2-\alpha}}\Sigma$ for this example. The diagonal entries of $A_6\Sigma$ are roughly centered around 1, but have different values. On the other hand, $(A_6)^{\frac{1}{2-\alpha}}\Sigma$ produces a scaled identity matrix, roughly equal to $(2-\alpha)I$, where I is the identity matrix. Of course, from here it is easy to recover a matrix that multiplies by Σ to produce the exact identity matrix, if required. Table 4.4 shows that the variance does not increase in this case, meaning the approximaion is still of minimal variance.

α	$A_6\Sigma$	$(A_6)^{\frac{1}{2-\alpha}}\Sigma$
1.00	[1.0, 1.0, 1.0, 1.0, 1.0, 1.0]	[1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
1.01	[1.007, 1.005, 1.003, 1.0, 0.996, 0.989]	[0.989, 0.989, 0.989, 0.989, 0.989, 0.989]
1.02	[1.014, 1.01, 1.006, 1.0, 0.992, 0.978]	[0.978, 0.978, 0.978, 0.978, 0.978, 0.978]
1.05	[1.034, 1.025, 1.013, 0.999, 0.979, 0.945]	[0.943, 0.943, 0.943, 0.943, 0.943, 0.943]

Table 4.3: Example 1: Diagonal values of $A_6\Sigma$ and $(A_6)^{\frac{1}{2-\alpha}}\Sigma$.

α	$A_6\Sigma$	$(A_6)^{\frac{1}{2-\alpha}}\Sigma$
1.00	12.000	12.000
1.01	12.000	11.736
1.02	11.998	11.470
1.05	11.989	10.661

Table 4.4: Example 1: Variance of the quadratic form $\rho_B^2(x,X)$ for $B = A_6\Sigma$ and $B = (A_6)^{\frac{1}{2-\alpha}}\Sigma$, using (4.1).

Note, however, that even without the correction described above, the minimal-variance polynomial is still an extremely close approximation to Σ^{-1} for values of α close to 1. The polynomials found using $k = 6$ and different values of α are given in Figure 4.3, where it is clear the polynomials are almost equal, particularly in the range of eigenvalues (i.e. between 1 and 6 for this example).

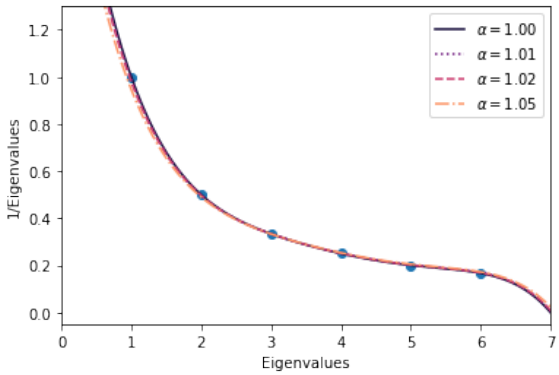


Figure 4.3: Example 1: Polynomial fit of A_6 with different values of α. All polynomials considered are roughly equal.

Example 2

The above exercise is now repeated, but with a degenerate covariance matrix. Let Σ be a diagonal matrix with entries $[5,4,3,2,1,0]$. The covariance matrix has $d = 6$ dimensions, but has rank $r = 5$, so only values of k up to and including 5 are considered.

Table 4.5a shows the values of $\omega_{\alpha,k}$ for this dataset. Unlike the previous example, when $k = 5$ and $\alpha = 1$, $\omega_{\alpha,k}$ does not equal 1. Recall that the constraint (4.2) of the minimal-variance polynomial is

$$\operatorname{trace}\left(A\Sigma^{\alpha}\right) = \operatorname{trace}\left(\Sigma^{\alpha-1}\right).$$

When $\alpha = 1$, this constraint is therefore

$$\operatorname{trace}\left(A\Sigma\right) = \operatorname{trace}\left(\Sigma^{0}\right),$$

the right-hand side of which is always equal to d. However, if A is to behave like a pseudoinverse, the matrix multiplication $A\Sigma$ should equal the identity matrix with the last $d-r$ diagonal entries equal to 0, meaning the desired constraint is actually $\operatorname{trace}(A\Sigma) = r$. Therefore, in degenerate cases, $\omega_{1,r} = d/r$. To counteract this, simply multiply A_r by r/d, which is shown in Table 4.5a (and other tables relating to this example) denoted by $\alpha = 1.00^{*}$. This is not an issue with other values of α, as $\Sigma^{\alpha-1}$ on the right side of the constraint has the same rank as Σ for $\alpha \neq 1$. Table 4.5b shows the variance of the minimal-variance distances with different values of k and α for example 2. As expected, as $\omega_{\alpha,k}$ decreases, so does the variance, for each value of k.

For $\alpha \neq 1$, Table 4.6 shows that A_5 with $\alpha = 1$ does not approximate the Moore-Penrose pseudoinverse Σ^{-} perfectly. For $\alpha = 1$, the diagonal values corresponding to nonzero values are equal to the diagonal values of $\frac{6}{5}\Sigma^{-}$, as discussed previously. The row with $\alpha = 1.00^{*}$ indicates the matrix has been multiplied by $5/6$, and the diagonal entries corresponding to the nonzero entries of Σ match the Moore-Penrose inverse exactly.

Table 4.7 shows the result of exponentiating A_5 by $\frac{1}{2-\alpha}$ when multiplied by Σ. As expected, when $\alpha = 1$, the diagonal entries of $A_5\Sigma$ are d/r. When $\alpha \neq 1$, the exponential is needed to ensure the minimal-variance approximation multiplied by Σ produces a multiple of the identity matrix I, as in example 1. Table 4.8 shows the variance of each approximation, which is not negatively affected by the exponentiation.

α \ k	2	3	4	5
1.00	1.255	1.210	1.201	1.200
1.00*	1.045	1.008	1.001	1.000
1.01	1.034	0.998	0.991	0.990
1.02	1.022	0.988	0.982	0.981
1.05	0.988	0.959	0.953	0.952

(a) $\omega_{\alpha,k}$

α \ k	2	3	4	5
1.00	15.055	14.520	14.411	14.400
1.00*	12.545	12.100	12.010	12.000
1.01	10.438	10.080	10.007	10.000
1.02	10.422	10.075	10.006	9.999
1.05	10.371	10.060	9.998	9.992

(b) $\mathrm{Var}\left(\rho_{A_k}(x,X)\right)$ with α

Table 4.5: Example 2: The values (3.d.p) of (a) $\omega_{\alpha,k}$ and (b) the variance of the minimal-variance distances with parameters α and k. Parameter $\alpha = 1.00^*$ indicates that $\omega_{1,k}$ has been multiplied by r/d.

α	A_5	$\Sigma^{\alpha-2}$
1.00	[0.24, 0.3, 0.4, 0.6, 1.2, 2.74]	[0.2, 0.25, 0.333, 0.5, 1.0, 0.0]
1.00*	[0.2, 0.25, 0.333, 0.5, 1.0, 2.238]	[0.2, 0.25, 0.333, 0.5, 1.0, 0.0]
1.01	[0.201, 0.251, 0.334, 0.499, 0.99, 2.249]	[0.203, 0.253, 0.337, 0.503, 1.0, 0.0]
1.02	[0.203, 0.252, 0.334, 0.497, 0.981, 2.216]	[0.207, 0.257, 0.341, 0.507, 1.0, 0.0]
1.05	[0.206, 0.255, 0.335, 0.493, 0.952, 2.116]	[0.217, 0.268, 0.352, 0.518, 1.0, 0.0]

Table 4.6: Example 2: Diagonal entries of A_5 and $\Sigma^{\alpha-2}$ for different values of α. $\alpha = 1.00^*$ indicates that $\omega_{1,k}$ has been multiplied by r/d.

α	$A_5\Sigma$	$(A_5)^{\frac{1}{2-\alpha}}\,\Sigma$
1.00	[1.2, 1.2, 1.2, 1.2, 1.2, 0.0]	[1.2, 1.2, 1.2, 1.2, 1.2, 0.0]
1.00*	[1.0, 1.0, 1.0, 1.0, 1.0, 0.0]	[1.2, 1.2, 1.2, 1.2, 1.2, 0.0]
1.01	[1.006, 1.004, 1.001, 0.997, 0.99, 0.0]	[0.99, 0.99, 0.99, 0.99, 0.99, 0.0]
1.02	[1.013, 1.008, 1.003, 0.995, 0.981, 0.0]	[0.98, 0.98, 0.98, 0.98, 0.98, 0.0]
1.05	[1.032, 1.02, 1.006, 0.986, 0.952, 0.0]	[0.95, 0.95, 0.95, 0.95, 0.95, 0.0]

Table 4.7: Example 2: Diagonal values of $A_5\Sigma$ and $(A_5)^{\frac{1}{2-\alpha}}\,\Sigma$. $\alpha = 1.00^*$ indicates that $\omega_{1,k}$ has been multiplied by r/d.

α	$A_5\Sigma$	$(A_5)^{\frac{1}{2-\alpha}}\Sigma$
1.00	14.400	14.400
1.00*	10.000	10.000
1.01	10.000	9.807
1.02	9.999	9.613
1.05	9.992	9.019

Table 4.8: Example 2: Variance of the quadratic form $\rho_B^2(x,X)$ for $B = A_5\Sigma$ and $B = (A_5)^{\frac{1}{2-\alpha}}\Sigma$, using (4.1). $\alpha = 1.00^*$ indicates that $\omega_{1,k}$ has been multiplied by r/d.

Similarly to example 1, Figure 4.4 shows that the minimal-variance polynomials are very similar for various values of α, except for $\alpha = 1$. This is due to the rank of trace$\left(\Sigma^0\right)$, as previously described. As before, simplify multiply this polynomial by $r/d = 5/6$ to find the polynomial with correct rank, denoted by $\alpha = 1.00^*$ in Figure 4.4.

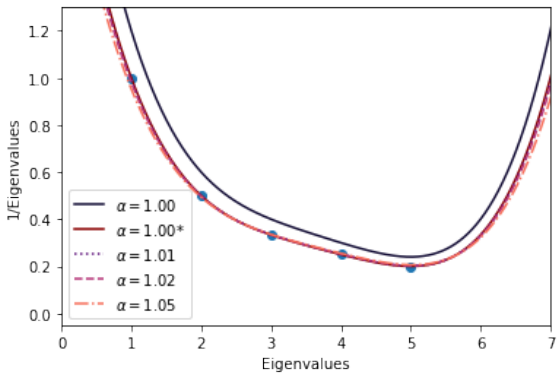


Figure 4.4: Example 2: Polynomial fit of A_5 with different values of α. The parameter $\alpha = 1.00^*$ denotes that the polynomial has been scaled by r/d. None of the polynomials have been exponentiated by $1/(2-\alpha)$.

4.4 Efficiency of the minimal-variance and simplicial distances

In this section, the efficiency of the minimal-variance distances with $\alpha = 1$ and the simplicial distances with $\delta = 2$ (introduced in Chapter 3) are compared in relation to the Mahalanobis distance. In Section 3.2.2, it is shown that the simplicial distances with

$k = r$, where $r = \text{rank}(\Sigma)$, are equal to the Mahalanobis distances over r (where the Moore-Penrose pseudoinverse is used if $r \neq d$). To align with the simplicial distances, the Mahalanobis distances are multiplied by $1/r$ in this section, and the minimal-variance distances are multiplied by $1/d$ for comparability. The previous section explains why $1/d$ is needed rather than $1/r$ in the minimal-variance case with $\alpha = 1$.

Define the efficiency of the minimal-variance distances as the variance of the minimal-variance distances over the variance of the Mahalanobis distances:

$$\text{eff}_{MV}^{(k)} = \frac{\text{Var}\left(\rho^2_{A_k/d}(x,X)\right)}{\text{Var}\left(\rho_{\Sigma^{-1}/r}(x,X)\right)} = \frac{2/\left(S^{\top}_{(1,k)}(V^{\top}V)^{-1}S_{(1,k)}\right)}{2/r}, \tag{4.22}$$

with $\text{Var}\left(\rho^2_{A_k}(x,X)\right)$ derived in Equation (4.14) and $\text{Var}\left(\rho_{\Sigma^{-1}}(x,X)\right)$ defined in Appendix A.4.

The efficiency of the simplicial distances with $\delta = 2$ is defined analogously as

$$\text{eff}_{simp}^{(k)} = \frac{\text{Var}\left(\rho^2_{k,2}(x,X)\right)}{\text{Var}\left(\rho^2_{\Sigma^{-1}/r}(x,X)\right)} = \frac{(2/k^2)\text{trace}\left([S_k\Sigma]^2\right)}{2/r}, \tag{4.23}$$

with $\text{Var}\left(\rho^2_{k,2}(x,X)\right)$ stated in (A.1) in Appendix A.2.

In the following three examples, $N = 500$ points are generated from d-dimensional multivariate normal distributions with zero mean and diagonal covariance matrices Σ_i with eigenvalues $\Lambda_i = \{\lambda_1, \ldots, \lambda_d\}$, for $i = 1, 2, 3$. The eigenvalues of each covariance matrix Σ_i are, respectively:

$$\begin{aligned}
\Lambda_1 &= [10,7,6,5,4,3,2,1,1,1], \\
\Lambda_2 &= [10,4,3,2,1,1,1,1,1,1], \\
\Lambda_3 &= [10,5,3,2,1,1,1,1,1,0].
\end{aligned}$$

Table 4.9 demonstrates the good efficiency of the minimal-variance distances even for small k. Note that $k-1$ is the order of the minimal-variance polynomial; in these examples, linear and quadratic polynomials perform well. The efficiency of the simplicial distances improves as k gets larger but also has variance tolerably close to that of the Mahalanobis distance even for k significantly smaller than r.

For larger dimensions, with covariance matrices possessing a number of zero eigenvalues, the examples are more striking. Table 4.10 gives the results of performing the same

exercise on the datasets described in Section 3.2.3, with eigenvalues given in Table 3.1. Table 4.10 shows that the minimal-variance distances start to have similar variance to the squared Mahalanobis distances (with the Moore-Penrose pseudoinverse) when using much lower values of k than the simplicial distances. For values as low as $k = 2$, the variance of the minimal-variance distances is much closer than the variance of the simplicial distances to the variance of the Mahalanobis distances.

	Λ_1		Λ_2		Λ_3	
k	$\text{eff}_{simp}^{(k)}$	$\text{eff}_{MV}^{(k)}$	$\text{eff}_{simp}^{(k)}$	$\text{eff}_{MV}^{(k)}$	$\text{eff}_{simp}^{(k)}$	$\text{eff}_{MV}^{(k)}$
1	1.51	1.51	2.16	2.16	2.06	2.06
2	1.40	1.18	1.64	1.20	1.62	1.22
3	1.32	1.05	1.38	1.02	1.37	1.03
4	1.24	1.01	1.23	1.00	1.21	1.00
5	1.17	1.00	1.13	1.00	1.12	1.00
6	1.12	1.00	1.07	1.00	1.06	1.00
7	1.07	1.00	1.04	1.00	1.02	0.99
8	1.03	1.00	1.01	1.00	1.00	1.00
9	1.01	1.00	1.00	1.00	1.00	1.00
10	1.00	0.00	1.00	0.00	N/A	N/A

Table 4.9: Efficiences (4.22) and (4.23) for different k, with three different sets of eigenvalues of the covariance matrix Σ given by Λ_i, $i = 1, 2, 3$.

	Λ_A		Λ_B		Λ_C	
k	$\text{eff}_{simp}^{(k)}$	$\text{eff}_{MV}^{(k)}$	$\text{eff}_{simp}^{(k)}$	$\text{eff}_{MV}^{(k)}$	$\text{eff}_{simp}^{(k)}$	$\text{eff}_{MV}^{(k)}$
1	7.58	7.58	24.25	24.25	10.00	10.00
2	2.83	2.07	9.79	4.72	9.17	5.55
3	2.09	1.84	5.87	1.95	4.77	1.87
4	1.87	1.80	4.21	1.90	3.20	1.83

Table 4.10: Efficiences (4.22) and (4.23) for different k, with three different sets of eigenvalues of the covariance matrix Σ as given in Table 3.1 in Section 3.2.3.

Little performance gain is seen when choosing $k > 3$ in these examples, indicating that A_k even with small k is often a good enough approximation to the inverse of the covariance matrix from the viewpoint of the distances generated by this matrix. From an efficiency perspective, the minimal-variance distances produce better results at a lower computational cost than the simplicial distances.

4.5 Applications of the minimal-variance distances

As with the simplicial distances discussed in Chapter 3, the minimal-variance distances have a multitude of applications due to the dependancy of multivariate data analysis on distance measures. Distances in correlated high dimensional spaces are notoriously difficult to measure reliably due to variable subspaces [182] and singularity [235], which hinder the performance of the Mahalanobis distance. Classical low-dimensional distance measures such as the Euclidean and Manhattan distances also struggle, thanks to the correlated nature of high dimensional data and the issues of diminishing relative contrast [5]. For further information on distance measures in high dimensions, see Section 2.4.2 of the literature review.

This section considers the use of minimal-variance distances in applications such as K-means clustering and outlier detection. It is also shown that the minimal-variance distances perform well in cases where the dimensionality d is greater than the number of observations N in a dataset, which has been a prominent and much-studied problem in multivariate data analysis in recent years [9, 95, 204].

4.5.1 K-means clustering

The K-means clustering method was introduced in Section 3.5.2, with Algorithm 1 detailing the exact procedure. As discussed previously, the K-means algorithm can benefit from replacing the classic Euclidean distance measure with a distance that can account for elliptical distributions in data. Figure 3.15 in Section 3.5.2 illustrates the potential benefit available when using the simplicial distance, rather than the Euclidean or so-called Mahalanobis-pinv distance, which uses the Moore-Penrose pseudoinverse in the Mahalanobis distance in case of degenerate data. Using the same datasets and methodology as those that produced Figure 3.15, Figure 4.5 shows the adjusted rand score of K-means

clustering with the different distance measures, now including the minimal-variance distance. Figure 4.5 shows that the minimal-variance distances have very similar performance to the simplicial distances with $k = 3$. Both of the distances suggested in this thesis outperform the Euclidean and Mahalanobis-pinv distances on the degenerate, correlated datasets generated in this example.

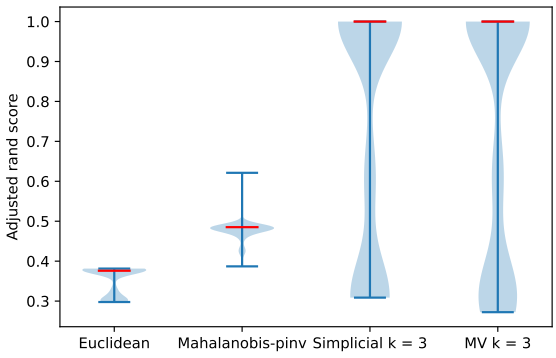


Figure 4.5: Violin plot showing the distribution of adjusted rand scores of 500 runs of K-means clustering with the Euclidean, Mahalanobis-pinv, simplicial $k = 3$ and minimal-variance $k = 3$ distances. The red line shows the median adjusted rand score of the 500 runs for each distance.

To further illustrate the performance of the K-means clustering algorithm with the minimal-variance and simplicial distances, 5 real datasets are considered. The datasets are obtained from the UCI Machine Learning Repository [68], with the exception of 'Digits' which was obtained through the Python package Scikit-Learn [185]. The details of the datasets are given in Table 4.11.

Each dataset was appropriately pre-processed: rows with missing values were removed, and the data was normalized such that each variable has values in range [0, 1]. It is important to note that the parameter K used in the K-Means clustering algorithm is used to indicate how many clusters are sought, and is different to the parameter k used in the distance measures. For each dataset, the choice of K in the K-Means algorithm is used as the 'true' number of clusters, given in Table 4.11, as these datasets are all fully labelled.

A more thorough discussion of how to use non-Euclidean distances in K-means clustering is given in Section 3.5.2. As with the examples given in that section, the minimal-variance, simplicial and Mahalanobis distances all require an initial estimate of the co-

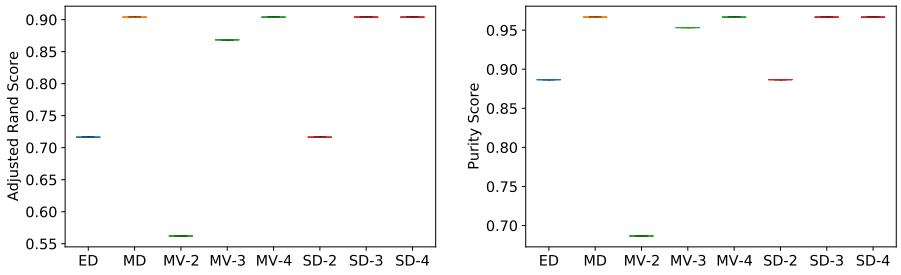
Dataset	d	N	No. of Clusters
Iris	4	150	3
Wine	13	178	3
Image Seg.	19	210	7
Digits	64	1797	10
Protein	77	1080	8

Table 4.11: Real datasets used to evaluate performance of distances when used with K-Means clustering. d is the number of variables in the dataset, and N is the number of observations.

variance matrices of the clusters, which is obtained by performing a few iterations of K-means clustering using the Euclidean distance first, as in [56]. Clearly, this initial estimate can have a large influence on the resulting clusters found by the other distances, so the K-Means algorithm is run 1000 times for each distance.

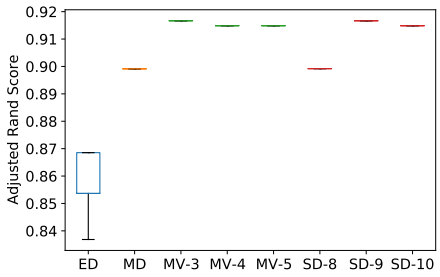
As these datasets are fully labelled, the labels given by K-Means can be compared to the 'true' labels to assess the performance of the clustering algorithm. Two external evaluation methods are used in this section, namely the adjusted rand (AR) score [113, 185] and the purity score [167]. These two different evaluation methods are used alongside one another to corroborate the results. AR scores are in the range [-1, 1] and purity scores are in the range [0, 1], with larger scores indicating a better labelling for both metrics. Further information about these clustering metrics is given in Appendix D.

Figure 4.6 gives the AR and purity scores for the K-Means clustering of each dataset using the distance measures being considered. For the minimal-variance and simplicial distances, the distances with values of k which produced the highest scores are shown. Note that the Mahalanobis distance uses the Moore-Penrose pseudoinverse when the data is degenerate. The eigenvalues for each of the datasets in Table 4.11 can be found in Appendix C.2. The influence of these eigenvalues on the performance of the distance measures is important, particularly when choosing values of k. When discussing eigenvalues being 'close to zero', this is in relation to the largest eigenvalue. There is no specific threshold for being 'close to zero', but the examples that follow should give some intuition about choosing the parameter k.

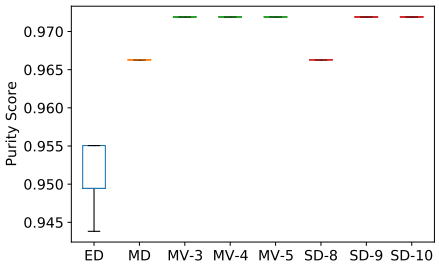


(a) Iris, AR

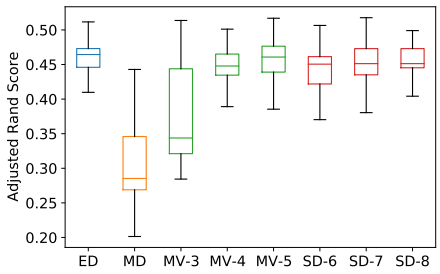
(b) Iris, Purity



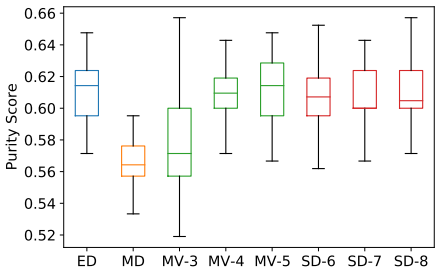
(c) Wine, AR



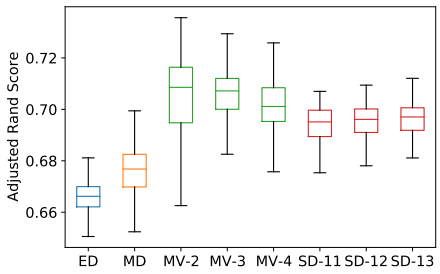
(d) Wine, Purity



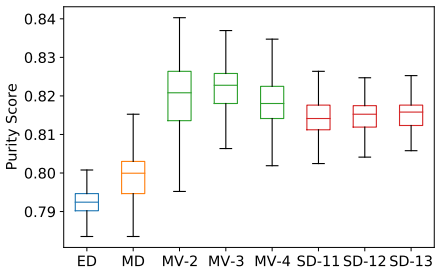
(e) Image, AR



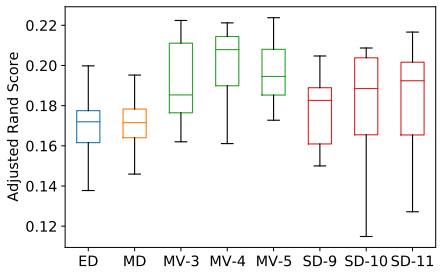
(f) Image, Purity



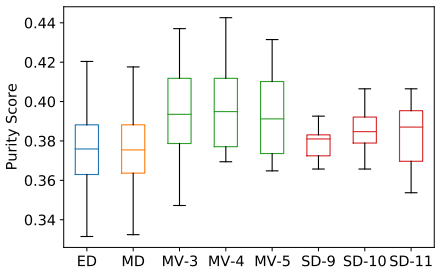
(g) Digits, AR



(h) Digits, Purity



(i) Protein, AR



(j) Protein, Purity

Figure 4.6: Adjusted rand scores and purity scores of the clusterings produced by K-Means when using different distance measures. ED: Euclidean distance, MD: Mahalanobis distance, MV-k: Minimal-variance distance with parameter k, SD-k: Simplicial distance with parameter k.

Iris is a 4-dimensional dataset, with no extreme small eigenvalues in comparison to its largest eigenvalue. Figure 4.6a, Figure 4.6b and Table 4.12 show that the Mahalanobis distance performs best, joint with the simplicial and minimal-variance distances when $k = 4$ (recall that these distances are equivalent to the Mahalanobis distance when $k = d$). The Iris dataset is low-dimensional and full-rank, and hence the Mahalanobis distance can use the true inverse of the covariance matrix. This example illustrates the performance gains that can be made in cluster analysis by taking the correlations in the data into consideration.

k	Simplicial	Min-Var
2	0.716 (0)	0.562 (0)
3	**0.904 (0)**	0.869 (0)
4	**0.904 (0)**	**0.904 (0)**
Euc.	0.716 (0)	
Mah.	**0.904 (0)**	

Table 4.12: Iris dataset: Median AR scores (and standard deviations) for each given distance. Bold figures denote the highest score(s) out of all methods used.

Figure 4.6c, Figure 4.6d and Table 4.13a consider the Wine dataset, and show that the Mahalanobis, minimal-variance and simplicial distances outperform the Euclidean distance, again highlighting the importance of accounting for correlation. The Wine dataset has some very small eigenvalues compared to its largest eigenvalue, and as such the Moore-Penrose pseudoinverse is likely to have been adversely impacted [109]. Choosing the minimal-variance or simplicial distance avoids this impact, and as such produces better clustering results.

The Wine example highlights that the minimal-variance distance performs well with lower values of k, whereas the simplicial distance requires a higher value of k to achieve its best results, as seen before in the efficiency evaluations in Section 4.4. This indicates that both distances can achieve equally good results, but the minimal-variance distance does so with lower computational time. However, the minimal-variance distance is more sensitive to a too-high choice of k, as seen by the decrease in AR scores in Table 4.13a.

k	Simplicial	Min-Var
2	0.714 (0.000)	0.817 (0.017)
3	0.759 (0.000)	**0.917 (0.007)**
4	0.759 (0.000)	0.915 (0.007)
5	0.818 (0.005)	0.915 (0.007)
6	0.833 (0.004)	0.915 (0.007)
7	0.899 (0.004)	0.915 (0.007)
8	0.899 (0.006)	0.915 (0.007)
9	**0.917 (0.004)**	0.915 (0.006)
10	0.915 (0.006)	0.913 (0.005)
11	0.915 (0.006)	0.869 (0.017)
12	0.915 (0.006)	0.899 (0.006)
13	0.899 (0.012)	0.854 (0.000)
Euc.	0.854 (0.01)	
Mah.	0.899 (0.01)	

(a) Wine dataset

k	Simplicial	Min-Var
2	0.360 (0.063)	0.238 (0.084)
3	0.247 (0.074)	0.344 (0.063)
4	0.339 (0.070)	0.448 (0.027)
5	0.392 (0.047)	**0.465 (0.027)**
6	0.451 (0.051)	0.460 (0.026)
7	0.451 (0.022)	0.456 (0.023)
8	0.451 (0.044)	0.454 (0.021)
9	0.449 (0.047)	0.454 (0.028)
Euc.	0.464 (0.020)	
Mah.	0.285 (0.067)	

(b) Image Segmentation dataset

Table 4.13: Wine and Image Segmentation datasets: Median AR scores (and standard deviations) for each given dataset and distance. Bold figures denote the highest score(s) out of all methods used.

The Image Segmentation dataset has a number of very large eigenvalues, some eigenvalues very close to zero, and five zero eigenvalues, see Appendix C.2 for more details on these eigenvalues. Figure 4.6e and Figure 4.6f show that the Mahalanobis distance performs worse than the Euclidean distance, perhaps due to the effect of very small eigenvalues on the Moore-Penrose pseudoinverse, as noted in [109].

The minimal-variance and simplicial distances outperform the Mahalanobis distance when clustering the Image Segmentation dataset, as they are less likely to be adversely affected by these small eigenvalues. Table 4.13b shows that the minimal-variance distance attains the highest AR score out of all the distances, but does not improve greatly on the Euclidean distance.

k	Simplicial	Min-Var
2	0.596 (0.019)	**0.709 (0.018)**
3	0.620 (0.017)	0.707 (0.017)
4	0.642 (0.019)	0.701 (0.016)
5	0.657 (0.019)	0.696 (0.017)
6	0.663 (0.017)	0.693 (0.016)
7	0.677 (0.019)	0.695 (0.016)
8	0.686 (0.019)	0.691 (0.015)
9	0.690 (0.019)	0.689 (0.142)
10	0.693 (0.019)	0.686 (0.144)
11	0.695 (0.018)	0.689 (0.111)
12	0.696 (0.018)	0.679 (0.230)
13	0.697 (0.018)	0.664 (0.174)
14	0.696 (0.018)	0.605 (0.226)
15	0.696 (0.017)	0.673 (0.166)
16	0.695 (0.017)	0.679 (0.173)
17	0.694 (0.017)	0.658 (0.180)
18	0.692 (0.017)	0.664 (0.222)
Euc.	0.666 (0.012)	
Mah.	0.677 (0.014)	

(a) Digits dataset

k	Simplicial	Min-Var
2	0.140 (0.057)	0.130 (0.057)
3	0.140 (0.055)	0.185 (0.020)
4	0.141 (0.050)	**0.208 (0.021)**
5	0.143 (0.044)	0.195 (0.015)
6	0.156 (0.024)	0.194 (0.019)
7	0.164 (0.024)	0.189 (0.014)
8	0.176 (0.026)	0.183 (0.017)
9	0.183 (0.026)	0.181 (0.027)
10	0.189 (0.028)	0.179 (0.032)
11	0.192 (0.026)	0.178 (0.041)
12	0.194 (0.021)	0.186 (0.058)
13	0.197 (0.020)	0.184 (0.022)
14	0.196 (0.019)	0.185 (0.008)
15	0.191 (0.021)	0.184 (0.026)
Euc.	0.172 (0.012)	
Mah.	0.172 (0.012)	

(b) Protein dataset

Table 4.14: Digits and Protein datasets: Median AR scores (and standard deviations) for each given dataset and distance. Bold figures denote the highest score(s) out of all methods used.

The Digits and Protein datasets (Tables 4.14a and 4.14b, respectively) both have a substantial number of small and zero eigenvalues (see Appendix C.2), indicating why the distances proposed perform better than the Mahalanobis distance (using the pseudoinverse). The Mahalanobis distance does not add much performance gain compared to the Euclidean distance in these examples, but the correct choice of k in the minimal-variance or simplicial distances provides improvement.

These examples show that K-Means clustering with the minimal-variance distance reaches its best AR score with relatively low values of k, whereas the simplicial distance needs higher values of k to reach this. However, the simplicial distance is less likely to breakdown for too-high a choice of k, as seen in Table 4.13a and Table 4.14a with the minimal-variance distance. For the simplicial distance, the values of k that produce the best AR score roughly match with the number of 'larger' eigenvalues in the datasets, as seen in Section 3.3.1.

4.5.2 Outlier labelling

Outlier labelling is often reliant on distance measures to measure the proximity of a point to a set of points [132, 214]. This section will apply the Euclidean, Mahalanobis, simplicial and minimal-variance distances in an outlier detection setting. 23 real datasets are used in this section, all obtained from the Outlier Detection DataSets (ODDS) source [198], which collates labelled outlier detection datasets for use by the research community. Information on the dimensionality, number of observations and number of outliers in each dataset is given in Table C.2 in Appendix C.2.

The method used will be a simple one (the same as in Section 3.5.1) to ensure the comparisons are made based on the distances used, rather than their interaction with more complex outlier detection methods. As the datasets used in this example are fully labelled, it is known how many outliers there are. Let m be the number of outliers for each dataset. For each distance being considered, calculate the distance from each point to the dataset itself, and label the furthest m points as the outliers. Table 4.15 gives the adjusted rand (AR) scores of the labels assigned by each distance (see Appendix D.1 for information on the AR score metric). For the simplicial and minimal-variance distances, only the highest AR scores are given, along with the corresponding value of k used to find them. Note that the Moore-Penrose pseudoinverse has been used in the Mahalanobis distance, as several of the datasets considered have singular covariance matrices.

The Euclidean distance only performs best on 2 of the 23 datasets considered (BreastW and Glass, the latter of which has the same AR score as the Mahalanobis distance). The Mahalanobis distance has a strictly higher AR score than the other distances for 4 of the 23 datasets. Of those datasets where the maximum AR score is given by both the Ma-

halanobis distance and either the simplicial and/or minimal-variance distance (i.e. Glass, Speech, Musk, Ionosphere), the value of k used in the latter distance is much lower than the dimension d (and the rank r) of the datasets. This means that using the simplicial or minimal-variance distance is less computationally intensive than the Mahalanobis distance, which corresponds to these distances with $k = r$.

Dataset	Euclidean	Mahalanobis	Simplicial	Min-Var
Lympho	0.287	0.633	**0.814** (k = 5)	**0.814** (k = 2)
WBC	0.568	**0.620**	0.568 (k = 1)	0.568 (k = 1)
Glass	**0.066**	**0.066**	**0.066** (k = 1)	**0.066** (k = 1)
Vowels	0.142	0.569	0.569 (k = 11)	**0.611** (k = 6)
Cardio	0.554	0.547	**0.627** (k = 10)	**0.627** (k = 4)
Thyroid	0.123	0.580	0.491 (k = 5)	**0.558** (k = 5)
Musk	0.201	**1.000**	**1.000** (k = 2)	**1.000** (k = 2)
Satimage-2	0.825	0.652	**0.942** (k = 7)	**0.942** (k = 3)
Letter	-0.012	**0.268**	0.159 (k = 10)	0.258 (k = 10)
Speech	-0.000	**0.129**	0.016 (k = 4)	**0.129** (k = 5)
Pima	0.140	0.132	0.145 (k = 2)	**0.149** (k = 2)
Satellite	0.200	0.349	0.364 (k = 8)	**0.395** (k = 8)
Shuttle	0.864	0.951	**0.953** (k = 6)	0.947 (k = 6)
BreastW	**0.863**	0.830	**0.863** (k = 1)	**0.863** (k = 1)
Arrhythmia	0.333	**0.953**	0.402 (k = 9)	0.420 (k = 9)
Ionosphere	0.178	**0.743**	0.723 (k = 7)	**0.743** (k = 4)
MNIST	0.333	0.512	0.418 (k = 10)	**0.547** (k = 8)
Optdigits	-0.021	-0.028	0.135 (k = 21)	**0.207** (k = 3)
Cover	-0.010	0.077	0.384 (k = 5)	**0.507** (k = 4)
Mammography	0.247	0.355	0.347 (k = 5)	**0.367** (k = 5)
Annthyroid	0.035	**0.318**	0.297 (k = 5)	0.305 (k = 4)
Pendigits	0.173	0.053	0.372 (k = 3)	**0.398** (k = 2)
Wine	0.875	0.755	0.875 (k = 1)	**1.000** (k = 4)

Table 4.15: Adjusted rand (AR) scores of the outlier labellings given by different distance measures. Bold values indicate the highest AR score(s) across the distances for each dataset.

There is one dataset where the simplicial distance performs strictly best (Shuttle), but there are six more where it performs joint best with the minimal-variance distance. However, the minimal-variance distance performs strictly best on 10 of the 23 datasets, and joint best on a further 8 datasets. This means of the 23 datsets considered, the minimal-variance distance performs best in detecting outliers out of all the distances considered for 18 of them.

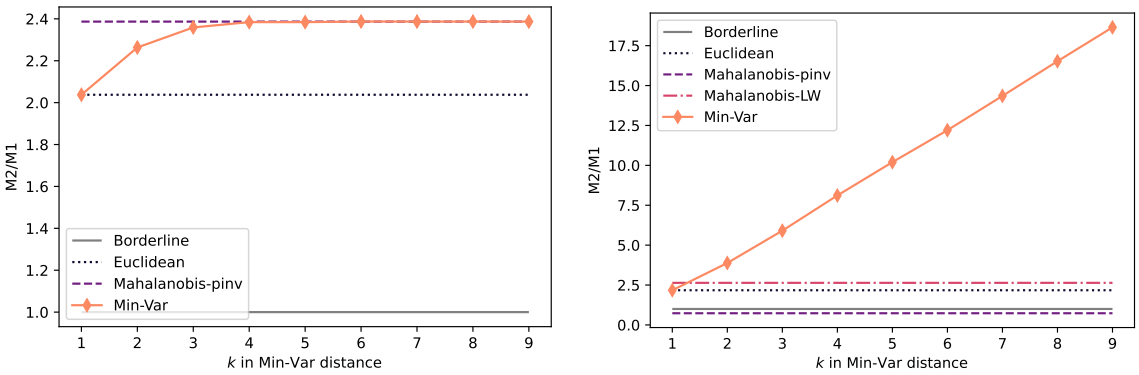
Consider the datasets (Lympho, Cardio, Satimage-2) where the simplicial and minimal-variance distances produce equal AR scores, not including those where $k = 1$ as this is just the Euclidean distance. The value of k used to achieve these AR scores is consistently lower in the minimal-variance distance than the simplicial distance, making it a more efficient distance measure to use, with less risk of instability.

Of those datasets where the minimal-variance distance does not perform the best out of all distance measures, it is often very close to the higher AR score (e.g. datasets Letter, Shuttle, Annthyroid). The datasets considered here all vary greatly in dimensionality, number of observations, number of outliers and distribution, and yet the minimal-variance seems to perform well on the vast majority of them.

4.5.3 Using minimal-variance distances when $d > N$

It is common for multivariate datasets to have dimension d less than the number of observations N, particularly in fields such as genomics [59], medical imaging and chemometrics [95]. More information on the issues of so-called 'high dimension low sample size' (HDLSS) datasets is given in Section 2.4 of the literature review.

Two synthetic datasets will be used to compare the performance of distances on high dimensional datasets; one dataset has $d < N$, the other has $d > N$. In the $d < N$ example, X is a 100-dimensional dataset made of two clusters, X_1 and X_2. Both clusters have 500 points each and identity covariance matrix. X_1 has mean $\mu_1 = 0$, and X_2 has mean $\mu_2 = 1$. The distance to μ_1, i.e. to the cluster X_1, is found using the Euclidean distance, the Mahalanobis distance, and the minimal-variance distance with $k \in [1, 9]$, for every point $x \in X$. For each distance measure, let $M1$ be the mean distance from the points in X_1 to μ_1, and let $M2$ be the mean distance from the points in X_2 to μ_1. The ratio $M1/M2$ therefore indicates how well separated the distances are between the clusters.



(a) $d < N$ (b) $d > N$

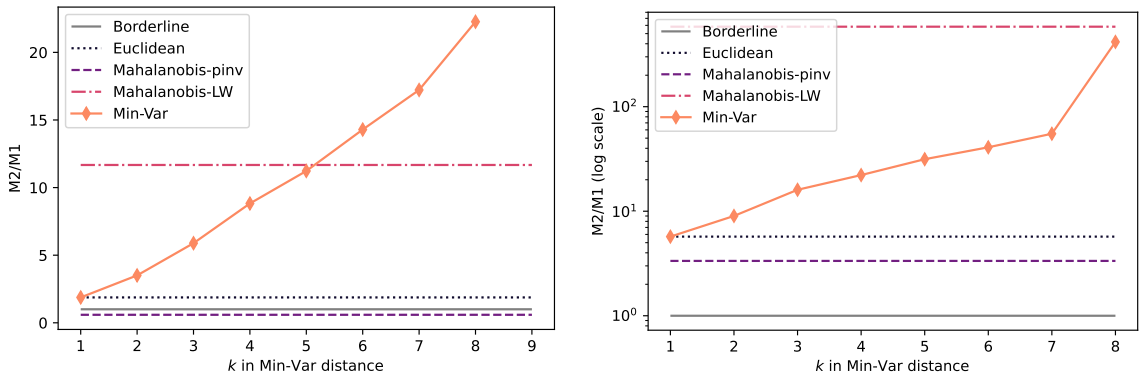
Figure 4.7: The ratios of the mean of the distances from the first cluster over the mean of the distances from the second cluster, using the Euclidean, Mahalanobis and minimal-variance distances. The x-axis indicates the value of k used in the minimal-variance distance.

Figure 4.7a shows a plot of the ratios $M1/M2$ for each of the distances considered, and the exact values are given in Table 4.16. The grey solid line shows the 'borderline' value of 1, which would imply no separability in the clusters. The Mahalanobis distance clearly demonstrates the best separability, as expected. The Euclidean distance also shows good separability, and the minimal-variance distances transition between the two classical distance measures as the parameter k increases.

For the $d > N$ case, the only difference in the construction of the dataset is that the dimensionality is 200, and both clusters only have 50 points each. The same experiment is performed, where the Mahalanobis distance now uses the Moore-Penrose pseudoinverse as the covariance matrix of this dataset is singular, and is referred to as the 'Mahalanobis-pinv' distance. The 'Mahalanobis-LW' distance uses the Ledoit-Wolf shrinkage covariance estimator, which is discussed in Section 2.5.2 of the literature review. Here, the Ledoit-Wolf shrinkage covariance estimator is found using the `covariance.LedoitWolf` class from the Scikit-Learn package [185] in Python.

Figure 4.7b shows the ratio $M2/M1$ in the $d > N$ case. In this example, the Mahalanobis-pinv distance produces a ratio less than 1 (0.73 to be exact, see Table 4.16 for all values), indicating that the distance between points in X_2 were measured closer to μ_1 than the points in X_1 were. The Moore-Penrose pseudoinverse is known to have issues when

$d > N$ [109], so this finding is perhaps unsurprising. The Mahalanobis distance is clearly improved by using the Ledoit-Wolf estimator in place of the sample covariance matrix; the ratio using this distance is greater than 1, and greater than the ratio obtained when using the Euclidean distance. However, the minimal-variance distances (using the classical sample covariance matrix) clearly outperform the other distance measures in this example, with a higher ratio value for all values of $k > 1$.



(a) Lung Cancer Dataset

(b) Breast Cancer Dataset (log scale)

Figure 4.8: The ratios of the mean of the distances from the first cluster over the mean of the distances from the second cluster in (a) the Lung Cancer dataset and (b) the Breast Cancer dataset, using the Euclidean, Mahalanobis and minimal variance distances. The x-axis indicates the value of k used in the minimal-variance distances. Note that the y-axis in (b) is log-scale.

Figure 4.8 shows the results of the same exercise having been performed on real datasets. The 'Lung Cancer' dataset [105] (obtained from the UCI Machine Learning Repository [68]) has 56 dimensions and only 36 observations, making it a $d > N$ dataset. The dataset has 3 groups, but the second and third group have been merged in this example to make it a binary classification problem. The distance is again measured from all points in the first cluster to its mean μ_1, and from all points in the second cluster to μ_1. Figure 4.8a shows that the Mahalanobis distance with the Moore-Penrose pseudoinverse does not appropriately separate the data, with a ratio value of 0.59 (again given in Table 4.16). The Euclidean distance provides an improvement on this, as does the Mahalanobis distance with the Ledoit-Wolf shrinkage estimator. However, the minimal-variance distances with $k > 5$ clearly provide the best separation of the clusters.

The 'Breast Cancer' dataset [170] is an array of mRNA expression data, with $d = 1925$ and $N = 133$, made up of two clusters. The Mahalanobis distance (with the Moore-Penrose pseudoinverse) and the Euclidean distance both surpass the borderline value of 1, but the minimal-variance distances provide a much larger separation. The Mahalanobis distance with the Ledoit-Wolf shrinkage estimator has the highest ratio $M2/M1$ in this example, see Figure 4.8b and Table 4.16, but the minimal-variance distance is very close.

The minimal-variance distances consistently provide reliable results (and often the best results) when considering the separation of clusters in datasets with both $d < N$ and $d > N$. Other distance measures may sometimes provide better results in given circumstances (e.g. the Mahalanobis distance when $d < N$, and the Mahalanobis distance with the Ledoit-Wolf estimator in given $d > N$ examples), but the minimal-variance distances perform consistently well over all examples. Thus, it can be recommended to use the minimal-variance distance for multivariate data analysis tasks that rely on good separation, such as clustering and classification problems.

Distance	$d < N$	$d > N$	Lung Cancer	Breast Cancer
Euclidean	2.04	2.17	1.88	5.71
Mahalanobis	**2.39**	0.73	0.59	3.35
Mahalanobis-LW	1.05	2.64	11.67	**582.52**
Minimal-Variance				
$k = 1$	2.04	2.17	1.88	5.71
$k = 2$	2.26	3.88	3.51	9.01
$k = 3$	2.36	5.91	5.89	16.04
$k = 4$	*2.38*	8.11	8.82	22.17
$k = 5$	*2.38*	10.2	11.23	31.45
$k = 6$	**2.39**	12.2	14.29	40.88
$k = 7$	**2.39**	14.35	17.22	55.10
$k = 8$	**2.39**	*16.53*	*22.27*	*416.72*
$k = 9$	**2.39**	**18.65**	**24.22**	63.46

Table 4.16: Values of the ratio M2/M1 for each of the distances considered, for each dataset in Section 4.5.3. Bold and italic values indicate the highest and second highest values of the ratio for each dataset.

4.6 Alternative constraints for minimal-variance distances

In Section 4.2, the minimal-variance distances are constructed by minimizing the variance of distances produced, subject to a constraint. The constraint given in (4.2) is $\text{trace}(A\Sigma^{\alpha}) = \text{trace}\left(\Sigma^{\alpha-1}\right)$, where the parameter α has default value 1 in [85] and in this thesis. The motivation for this choice of constraint is to force the matrix A to behave similarly to the inverse covariance matrix Σ^{-1}, if it exists. However, this is clearly not the only possible constraint that can force A to behave similarly to the inverse of Σ. In this section, the performance of the minimal-variance distances with two different constraints will be considered. Let the above constraint with $\alpha = 1$ be denoted 'constraint 0' in this section.

Constraint 1 The first new method seeks to minimize $\text{Var}\left(\rho^2_{A_k}(x, X)\right)$ subject to the new constraint

$$||A_k\Sigma - I||^2 = 0, \tag{4.24}$$

where $||B||^2 = \text{trace}(B^\top B)$, for any matrix B. Following the same method as that in Section 4.2.1, using γ as the Lagrange multiplier, the Lagrangian is found to be

$$\begin{aligned}
\Phi(A_k) &= \text{trace}\left([A_k\Sigma]^2\right) + \gamma||A_k\Sigma - I||^2 \\
&= \text{trace}\left(A_k\Sigma^2A_k\right) + \gamma\left(\text{trace}\left(A_k\Sigma^2A_k\right) - 2\text{trace}(A_k\Sigma) + d\right) \\
&= (1+\gamma)\text{trace}\left(\theta^\top\Sigma_{(k)}\Sigma^2\Sigma_{(k)}\theta\right) - 2\gamma\text{trace}\left(\theta^\top\Sigma_{(k)}\Sigma\right) + \gamma d \\
&= (1+\gamma)\theta^\top V^\top V\theta - 2\gamma\theta^\top S_{(1,k)} + \gamma d \\
&= \phi_k(\theta).
\end{aligned}$$

Therefore, minimizing the variance subject to the constraint (4.24) is equivalent to minimizing $\phi_k(\theta)$ with respect to θ. Differentiating $\phi_k(\theta)$, setting the result to zero and rearranging for θ gives the solution

$$\hat{\theta}_\gamma = \frac{\gamma}{1+\gamma}(V^\top V)^{-1}S_{(1,k)}.$$

That is, the previous constant $\omega_{\alpha,k}$ is replaced by $\gamma/(1+\gamma)$. If $\gamma \to \infty$, the BLUE is obtained.

Constraint 2 The second new method aims to minimize $\text{Var}\left(\rho^2_{A_k}(x, X)\right)$ subject to the constraint

$$||\Sigma A_k\Sigma - \Sigma||^2 = 0,$$

where, again, $||B||^2 = \text{trace}(B^\top B)$. The Lagrange function in this case is

$$\begin{aligned}\Psi(A_k) &= \text{trace}\left([A_k\Sigma]^2\right) + \gamma||\Sigma A_k\Sigma - \Sigma||^2 \\ &= \text{trace}\left(A_k^\top\Sigma^2 A_k\right) + \gamma(\text{trace}\left(A_k\Sigma^4 A_k\right) - 2\text{trace}(A_k\Sigma^3) + \text{trace}(\Sigma^2)) \\ &= \theta^\top V^\top V\theta + \gamma(\theta^\top\bar{V}^\top\bar{V}\theta - 2\theta^\top S_{(3,k)} + S_2) \\ &= \psi_k(\theta),\end{aligned}$$

where

$$\bar{V} = \left(\lambda_j^{i+2}\right)_{\substack{j=1,\ldots,d,\\ i=0,\ldots,k-1}} = \begin{pmatrix}\lambda_1^2 & \lambda_1^3 & \ldots & \lambda_1^{k+1}\\ \lambda_2^2 & \lambda_2^3 & \ldots & \lambda_2^{k+1}\\ \vdots & \vdots & \vdots & \vdots\\ \lambda_d^2 & \lambda_d^3 & \ldots & \lambda_d^{k+1}\end{pmatrix}.$$

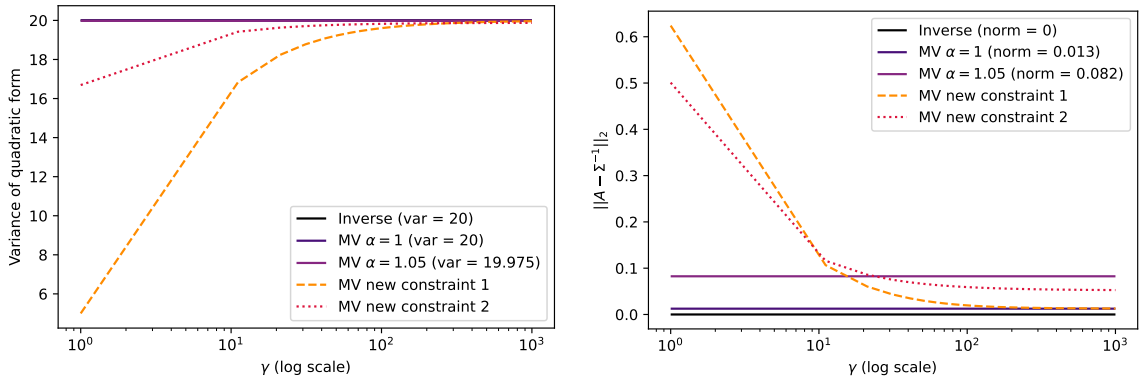
The solution to the minimization problem is given by

$$\bar{\theta}_\gamma = \gamma\left(V^\top V + \gamma\bar{V}^\top\bar{V}\right)^{-1} S_{(3,k)}.$$

Full rank example with new constraints Consider the 10-dimensional covariance matrix $\Sigma = \text{diag}(10, 9, 8, 7, 6, 5, 4, 3, 2, 1)$. Five different methods will be used to invert Σ:

- The true inverse Σ^{-1};
- The minimal-variance method with constraint 0 and $\alpha = 1$;
- The minimal-variance method with constraint 0 and $\alpha = 1.05$;
- The minimal-variance method with constraint 1 with various values of γ;
- The minimal-variance method with constraint 2 with various values of γ.

The first set of examples will use $k = 7$ in all minimal-variance methods. Figure 4.9a shows the variances of the quadratic form using each of the methods above, using Equation (4.1) to find this variance. The two new constraints give a much lower variance for low values of γ, and converge towards the variance of the other methods as γ increases. As previously seen, low variances in distances can provide more helpful distance measures in multivariate settings. Figure 4.9b shows the 2-norm $||A_7 - \Sigma^{-1}||$ between the matrix A_7 produced by each technique and the true inverse Σ^{-1}. Clearly, the value for the inverse is 0. The new constraints have larger differences to the inverse for low values of γ, but this converges to a smaller value as γ increases, particularly for the first new constraint.



(a) Variance of quadratic forms

(b) Norms $\left\|A_7 - \Sigma^{-1}\right\|_2$

Figure 4.9: Comparisons of the minimal-variance methods with different constraints, with $k = 7$, for a full-rank matrix. The x-axis shows the parameter γ, which relates to the two new constraints.

Table 4.17 shows the variances of distances produced using the new minimal-variance methods, as well as the eigenvalues of the product $A_7\Sigma$. For contrast, when using the exact inverse, the variance is exactly 20, and the eigenvalues are all exactly 1. When using the original minimal-variance method with $k = 7$, the variance is 20.002, and the eigenvalues of $A_7\Sigma$ are [1.017, 1.012, 1.011, 1.003, 1.001, 0.995, 0.994, 0.994, 0.992, 0.982]. The eigenvalues for all minimal-variance methods remain close to one. The first new constraint produces lower variance for lower values of γ, whereas the second new constraint has lower variance as γ gets higher.

If the above example is considered with $k = 9$, the second new constraint has instability issues. However, the first new constraint has good results, given in Table 4.18. For reference, the original minimal-variance method with $k = 9$ has variance 20 and eigenvalues [1.001, 1.001, 1. , 1. , 1. , 1. , 1. , 0.999, 0.999, 0.999].

Both new constraints suggested give distances with lower variance than both the original minimal-variance method and using the true inverse. The eigenvalues of the product $A\Sigma$ are close to being all ones, particularly as the parameter γ increases. Much like the choice of α in the previous minimal-variance definition, the choice of the parameter γ informs the tradeoff between accuracy and low variance. The first new constraint is preferable in this example, as the variances are lower for most values of k, the eigenvalues of $A_7\Sigma$ are closer to all ones and it is more stable for higher values of k.

γ	Variance (constraint 1)	Eigenvalues of $A_7\Sigma$ (constraint 1)	Variance (constraint 2)	Eigenvalues of $A_7\Sigma$ (constraint 2)
50	19.221395	[0.997, 0.992, 0.991, 0.983, 0.981, 0.976, 0.975, 0.974, 0.972, 0.963]	19.77133	[1.02, 1.004, 1.002, 1.0, 1.0, 0.999, 0.996, 0.994, 0.99, 0.935]
100	19.603901	[1.006, 1.002, 1.001, 0.993, 0.991, 0.985, 0.984, 0.984, 0.982, 0.972]	19.824753	[1.023, 1.005, 1.003, 1.0, 1.0, 0.999, 0.996, 0.995, 0.991, 0.942]
250	19.838911	[1.012, 1.007, 1.007, 0.999, 0.997, 0.991, 0.99, 0.989, 0.988, 0.978]	19.857155	[1.026, 1.005, 1.003, 1.001, 1.0, 0.999, 0.996, 0.995, 0.992, 0.947]
500	19.918187	[1.014, 1.01, 1.009, 1.001, 0.999, 0.993, 0.992, 0.991, 0.99, 0.98]	19.86803	[1.026, 1.005, 1.003, 1.001, 1.0, 0.999, 0.996, 0.995, 0.992, 0.948]
1000	19.958004	[1.015, 1.011, 1.01, 1.002, 1.0, 0.994, 0.993, 0.992, 0.991, 0.981]	19.873449	[1.027, 1.005, 1.003, 1.001, 1.0, 0.999, 0.996, 0.995, 0.992, 0.949]

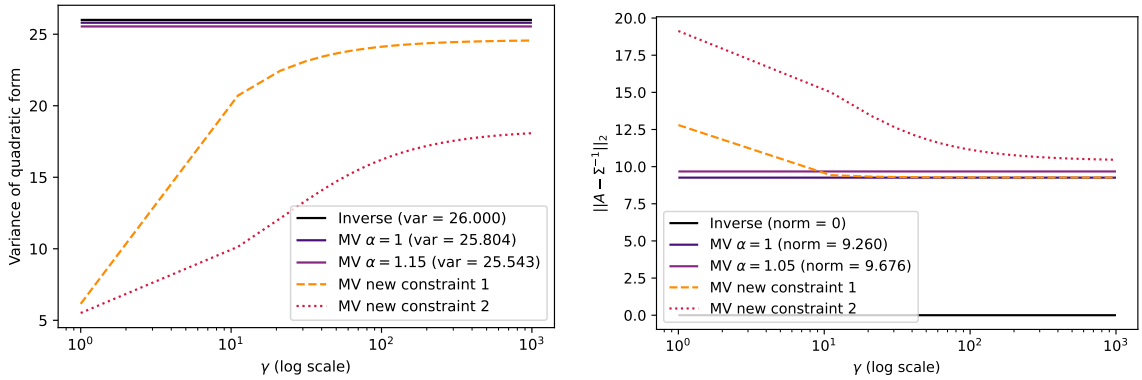
Table 4.17: The variance of the quadratic forms for the two new minimal-variance methods with $k = 7$, for different parameters γ, for a full-rank matrix. The eigenvalues of $A\Sigma$ are also given for both methods.

γ	Variance (constraint 1)	Eigenvalues of $A_9\Sigma$ (constraint 1)
50	19.22186	[0.981, 0.981, 0.981, 0.981, 0.980, 0.980, 0.980, 0.980, 0.980, 0.979]
100	19.60438	[0.991, 0.991, 0.990, 0.990, 0.990, 0.990, 0.990, 0.989, 0.989, 0.989]
250	19.83939	[0.997, 0.997, 0.996, 0.996, 0.996, 0.996, 0.996, 0.995, 0.995, 0.995]
500	19.91867	[0.999, 0.999, 0.998, 0.998, 0.998, 0.998, 0.998, 0.997, 0.997, 0.997]
1000	19.95849	[1.0, 1.0, 0.999, 0.999, 0.999, 0.999, 0.999, 0.998, 0.998, 0.998]

Table 4.18: The variance of the quadratic forms for the minimal-variance method using constraint 1 with $k = 9$, for different parameters γ, for a full-rank matrix. The eigenvalues of $A\Sigma$ are also given.

Degenerate example with new constraints The same exercise is repeated for a matrix that does not have full rank: $\Sigma = \text{diag}(5,4,2,1,1/2,1/3,1/4,1/5,1/6,1/7,1/8,1/9,1/10,0)$. Figure 4.10a shows the variance of the quadratic forms using the different methods as listed in the previous example, replacing the true inverse with the Moore-Penrose pseudoinverse Σ^-. The parameter $k = 5$ is used for all minimal-variance methods. The variance of the two new methods is much lower than the other methods, for all values of γ considered.

Figure 4.10b shows the 2-norm between the matrix produced and the true inverse of the covariance matrix. The two new methods have higher norm than the other methods, but it decreases as γ increases. Like the previous method, the parameter γ can be used to adjust the tradeoff between accuracy and variance.



(a) Variance of quadratic forms

(b) Norms $\left\|A_5 - \Sigma^-\right\|_2$

Figure 4.10: Comparisons of the minimal-variance methods with different constraints, with $k = 5$, for a degenerate matrix. The x-axis shows the parameter γ, which relates to the two new constraints.

The distances produced using the pseudoinverse have variance $2r = 26$, and the eigenvalues of $\Sigma^-\Sigma$ are all exactly one or zero. The original minimal-variance method has variance 25.804 when $k = 5$, and the eigenvalues of $A_5\Sigma$ are [1.399, 1.327, 1.173, 1.062, 1.032, 1.025, 1.02, 0.915, 0.819, 0.793, 0.741, 0.675, 0.62, 0]. Table 4.19 shows the variances of the quadratic forms with the two new constraints, and the eigenvalues of $A\Sigma$ for A produced by the two new methods. The eigenvalues using constraint 1 are slightly closer to 1 than those produced using constraint 2. However, the variance using constraint 2 is considerably smaller for all values of γ.

Overall, the optimal constraint to be used in the minimal-variance method is dependent on the dataset, the application and the desire for accuracy versus low variance. The minimal-variance method is extremely adaptable thanks to the interchangeable constraints shown here, and further adjustments to both the constraint and the function to be minimized could be considered.

γ	Variance (constraint 1)	Eigenvalues $A\Sigma$ (constraint 1)	Variance (constraint 2)	Eigenvalues $A\Sigma$ (constraint 2)
50	23.654503	[1.34, 1.27, 1.123, 1.017, 0.988, 0.981, 0.976, 0.876, 0.784, 0.76, 0.709, 0.646, 0.593, 0.0]	14.698196	[0.999, 0.999, 0.995, 0.982, 0.975, 0.829, 0.698, 0.598, 0.522, 0.462, 0.414, 0.375, 0.342, 0.0]
100	24.125228	[1.353, 1.283, 1.134, 1.027, 0.998, 0.991, 0.986, 0.885, 0.792, 0.767, 0.716, 0.653, 0.599, 0.0]	16.227192	[1.062, 1.0, 0.999, 0.998, 0.98, 0.913, 0.775, 0.667, 0.584, 0.518, 0.465, 0.421, 0.385, 0.0]
250	24.414439	[1.361, 1.29, 1.141, 1.033, 1.004, 0.997, 0.992, 0.89, 0.797, 0.772, 0.72, 0.656, 0.603, 0.0]	17.400193	[1.118, 1.0, 1.0, 1.0, 0.983, 0.973, 0.83, 0.717, 0.628, 0.558, 0.501, 0.455, 0.416, 0.0]
500	24.511999	[1.364, 1.293, 1.143, 1.035, 1.006, 0.999, 0.994, 0.892, 0.799, 0.773, 0.722, 0.658, 0.604, 0.0]	17.846217	[1.139, 1.0, 1.0, 1.0, 0.995, 0.984, 0.85, 0.735, 0.644, 0.572, 0.514, 0.467, 0.427, 0.0]
1000	24.560998	[1.365, 1.294, 1.144, 1.036, 1.007, 1.0, 0.995, 0.893, 0.799, 0.774, 0.722, 0.658, 0.605, 0.0]	18.08105	[1.149, 1.006, 1.0, 1.0, 1.0, 0.984, 0.861, 0.744, 0.653, 0.58, 0.521, 0.473, 0.433, 0.0]

Table 4.19: The variance of the quadratic forms for the two new minimal-variance methods with $k = 5$, for different parameters γ, for a degenerate matrix. The eigenvalues of $A_5\Sigma$ are also given for both methods.

4.7 Chapter summary

This chapter has presented the minimal-variance distances, which were first introduced in [85], a paper co-authored by myself and my supervisors. The distance measure aims to minimize the variance of the distances found, while producing results similar to the Mahalanobis distance. The benefits of using the minimal-variance distances include:

- The ability to use the minimal-variance distance measure in degenerate and near-degenerate data, something not available to the Mahalanobis distance;
- The adjustability of the distance: the degree parameter k allows for adjustment in computation time and applicability in degenerate data, and the parameter α controls the tradeoff between bias and low variance. The ability to change the constraint allows for further customisation of the distance;
- The ability to account for correlations and rotations in the data without imposing assumptions on the data, unlike many popular multivariate distance measures;
- The efficiency of the distance measure compared to other alternatives to the Mahalanobis distance.

The limitations of the minimal-variance distances include:

- Instability if k is chosen too high, as with many polynomial methods [101]. Possible methods to help improve this issue are given in Chapter 5;
- This method is reliant on an estimator of the sample covariance matrix, which is known to be problematic in high dimensions (see Section 2.5.1 in the literature review). However, examples with $d > N$ show this doesn't seem to cause issues, and the method could be extended to use different estimators of the covariance matrix.

Section 4.2 proposes two methods of constructing the minimal-variance distances. The polynomial method was first introduced in [85], and the weighted linear regression method was first suggested in Section 4.2.2. The distance was generalized to consider a new parameter α, which was not given in [85]. This parameter controls the constraint imposed on the distances, and has an effect on the variance and bias of the results. Both methods of construction produce the same minimal-variance distances, but offer different perspectives

which help to motivate the distance measure. The linear regression approach indicates that $\alpha = 1$ is a natural choice of the parameter α, and a theorem was given to prove that when $k = d$ and $\alpha = 1$, the minimal-variance matrix is equal to Σ^{-1} for full-rank Σ.

The choice of the parameter α is studied further in Section 4.3. It is shown that for $\alpha = 1$, the best linear unbiased estimator is obtained, whereas higher values of α produce a slightly biased but lower variance estimator. A correction for this biasedness is suggested in Section 4.3.2, and shown in practice on some empirical examples.

Section 4.4 explores the efficiency of the minimal-variance distances compared to the simplicial distances, where the efficiency of a distance is defined as the variance of the distance over the variance of the Mahalanobis distance. It is shown that the minimal-variance distances are more efficient, in this sense, than the simplicial distances, making them less computationally and time intensive than the simplicial distances. For small examples, the efficiency of both distances considered equals one for given values of k, showing the variances are equal to the Mahalanobis distance.

Numerical examples of applications using the minimal-variance distances are given in Section 4.5. The applications considered include a K-means clustering exercise, an outlier labelling example and a comparison of separation ability with $d > N$. All of these examples use real data, and the performance of the minimal-variance distances is compared to the simplicial distances, the Euclidean distance and the Mahalanobis distance across all examples. The minimal-variance distances often produce the best results out of the distance measures for all applications considered.

Finally, the flexibility of the minimal-variance distances is proven as Section 4.6 considers two new constraints. The constraints ensure that the minimal-variance matrix A behaves similarly to an inverse or a pseudoinverse, as appropriate. The two new constraints often have lower variance than the original constraint, but slightly less accuracy.

Overall, the minimal-variance distance is a strong alternative to the Mahalanobis distance in times when Σ is singular, or close to singularity. It performs well when there are small eigenvalues, and can outperform a variety of other distance measures on a wide range of tasks. The next chapter of this thesis introduces ‘minimal-variance whitening’, which takes the theory of minimal-variance distances and applies it to data transformation methods.

Chapter 5

Minimal-Variance Whitening

The research presented in this chapter is based on the contents of a publication I have co-authored [84], entitled **Polynomial whitening for high-dimensional data**, published in *Computational Statistics*, available at https://doi.org/10.1007/s00180-022-01277-6.

The differences between the published manuscript and the contents of this chapter are as follows:

- This chapter gives an extension to the minimal-variance whitening method in the form of 'iterative minimal-variance whitening', introduced in Section 5.4;
- Additional empirical examples are provided in this chapter, including examples utilising random projection for very high dimensional data, principal component analysis and iterative minimal-variance whitening;
- The 'fuzzy minimal-variance rank estimation' method is introduced in this chapter;
- Alternative constraints are suggested and explored in Section 5.6 of this chapter.

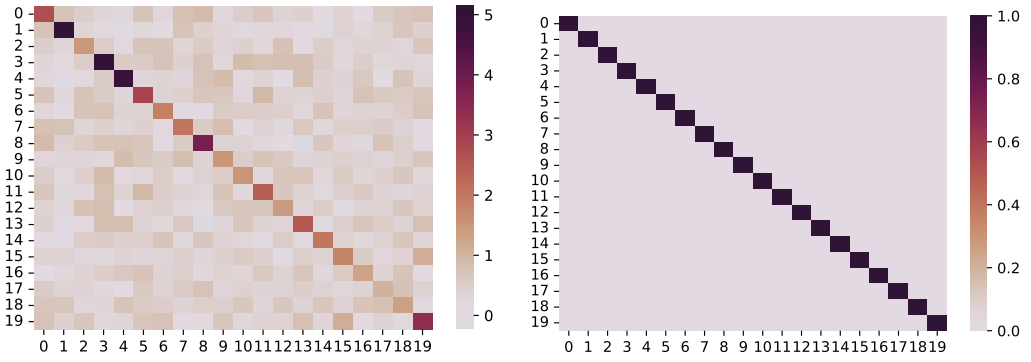
The aims of the research presented in this chapter are:

- To produce a method of data whitening which performs similarly to Mahalanobis whitening, particularly when Mahalanobis whitening is not available due to singularity in the covariance matrix;
- To minimize the total variation of the transformed data subject to a given constraint, following the trends of Chapter 4;

- To show how the minimal-variance whitening method can be used to detect singularity in the covariance matrix and approximate the matrix rank;
- To introduce the method of 'iterative minimal-variance whitening' for improved stability and performance;
- To highlight methods of applying minimal-variance whitening to extremely high dimensional data.

5.1 Introduction

Data whitening is a method of transforming a dataset by decorrelating and standardizing its variables. Let $X \in \mathbb{R}^{d \times N}$ be a d-dimensional dataset with N observations, with mean μ and covariance matrix Σ. Whitening transformations are typically of the form $A(X-\mu)$, where A is known as the whitening matrix. The most common method of whitening data is Mahalanobis whitening, which uses the whitening matrix $A = \Sigma^{-1/2}$. Figure 5.1 illustrates the effect that a whitening transformation has on the covariance matrix of a dataset. Figure 5.1a shows the heatmap of the covariance matrix of a dataset, and Figure 5.1b shows the heatmap of the covariance matrix of the same dataset after Mahalanobis whitening has been applied, with all correlations removed and equal values on the diagonal. More information on Mahalanobis whitening, and data whitening more generally, is given in Section 2.3 of the literature review.



(a) Original data (b) Whitened data

Figure 5.1: Heatmaps of the covariance matrices of (a) a dataset with correlations and variables of different scales (b) the same dataset after Mahalanobis whitening.

Much like the Mahalanobis distance seen in Chapter 4, whitening transformations are also reliant on the inverse of the covariance matrix (more specifically, the square root of the inverse). Due to singularity in the covariance matrix, which is common in high dimensions, the inverse does not always exist, making data whitening unavailable in such circumstances. The polynomial methods introduced in Chapter 4 have therefore inspired the production of similar methods with the aim of whitening data.

This chapter is structured as follows. Section 5.2 introduces the minimal-variance whitening method. More specifically, Section 5.2.1 details how the minimal-variance whitening method is formed, with the main result given in Theorem 4. Section 5.2.2 discusses the effect of the parameters in the minimal-variance polynomial, namely the degree parameter k and the constraint parameter α. Several numerical examples of the minimal-variance whitening method are given in Section 5.3, including outlier detection, dimension reduction and comparisons to other whitening methods. Section 5.4 introduces an extension to minimal-variance whitening called 'iterative minimal-variance whitening', and gives examples of this in practice. The final two subsections of the chapter give alternative ways that the minimal-variance whitening polynomial could be used: to estimate the rank of a matrix and to approximate Σ^{-1} (with comparisons to the method used to approximated Σ^{-1} in Chapter 5). A summary of the chapter is given in Section 5.8.

5.2 Constructing the minimal-variance whitening matrix

Much like the construction of the minimal-variance distances in Section 4.2.1, the minimal-variance whitening matrix is constructed through polynomials in the covariance matrix. A similar approach is used, now seeking to minimize the total variation of the whitened data. A constraint is again enforced on the polynomial to ensure the minimal-variance whitening matrix behaves like the square root of an inverse, where it exists.

5.2.1 Construction through polynomials

Let $X \in \mathbb{R}^{d\times N}$ be a matrix of data, with d dimensions and N observations. Denote the empirical mean vector and covariance matrix of X by $\mu \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d\times d}$, respectively. As in Chapter 4, let $S_\alpha = \text{trace}(\Sigma^\alpha)$ and $S_{(\alpha,k)} = (S_\alpha, S_{\alpha+1}, \ldots, S_{\alpha+k-1})$.

Define the vectors $\theta_\alpha = (\theta_0, \theta_1, \ldots, \theta_{k-1})^\top$ and $\Sigma_{(k)} = \left(\Sigma^0, \Sigma^1, \ldots, \Sigma^{k-1}\right)^\top$, where θ_α is dependent on a parameter α that will be discussed shortly. The matrix A_k is defined to be a $(k-1)$-degree polynomial in the covariance matrix Σ, of the form:

$$A_k = \sum_{i=0}^{k-1} \theta_i \Sigma^i = \theta_\alpha^\top \Sigma_{(k)}. \tag{5.1}$$

For a chosen integer k such that $k \leq d$, the objective is to find the vector of k coefficients of the matrix polynomial, denoted $\theta_\alpha = (\theta_0, \theta_1, \ldots, \theta_{k-1})^\top$ in Equation (5.1), so that the total variation of the transformed data $X_{A_k} = A_k(X - \mu)$ is minimized, subject to suitable constraints. The covariance matrix of the transformed data is $\mathcal{D}(X_{A_k}) = A_k \Sigma A_k^\top$, and the total variation of X_{A_k} is given by:

$$\begin{aligned}
\operatorname{trace}\left(\mathcal{D}(X_{A_k})\right) &= \operatorname{trace}\left(A_k \Sigma A_k^\top\right) \\
&= \operatorname{trace}\left(\sum_{i=0}^{k-1} \theta_i \Sigma^i \Sigma \sum_{j=0}^{k-1} \theta_j \Sigma^j\right) \\
&= \theta_\alpha^\top M_{(k)} \theta_\alpha,
\end{aligned}$$

where the matrix $M_{(k)}$ is defined as

$$M_{(k)} = \begin{pmatrix} S_1 & S_2 & \cdots & S_k \\ S_2 & S_3 & \cdots & S_{k+1} \\ \vdots & \vdots & \ddots & \vdots \\ S_k & S_{k+1} & \cdots & S_{2k-1} \end{pmatrix}.$$

To ensure non-trivial solutions to the minimization of the total variation, a constraint on the coefficient vector is needed. There are a number of options for this constraint, including constraints of the form

$$\operatorname{trace}\left(A_k \Sigma^\alpha\right) = \operatorname{trace}\left(\Sigma^{\alpha - 1/2}\right) \tag{5.2}$$

for some scalar value α. This can be written using the notation defined above as

$$\theta_\alpha^\top S_{(\alpha,k)} = S_{\alpha - 1/2}.$$

A constraint of this form ensures that the minimal-variance polynomial matrix A_k has similar qualities to $\Sigma^{-1/2}$, in the cases where $\Sigma^{-1/2}$ exists. Appropriate values of α in the constraint (5.2) will be considered in Section 5.2.2, after the following theorem. Theorem 4 derives the optimal coefficient vector θ_α to minimize total variation while adhering to the constraint (5.2).

Theorem 4. *Let $X \in \mathbb{R}^{d\times N}$ be a d-dimensional dataset with N observations, having empirical mean vector μ and empirical covariance matrix Σ. For $k \leq d$, $k \in \mathbb{Z}$, the matrix polynomial $A_k = \sum_{i=0}^{k-1} \theta_i \Sigma^i = \theta_\alpha^\top \Sigma_{(k)}$ that minimizes trace$\left(\mathcal{D}(X_{A_k})\right)$ subject to the constraint $\theta_\alpha^\top S_{(\alpha,k)} = S_{\alpha-1/2}$ has coefficients given by*

$$\hat{\theta}_\alpha = \frac{S_{\alpha-1/2}}{S_{(\alpha,k)}^\top M_{(k)}^{-1} S_{(\alpha,k)}} M_{(k)}^{-1} S_{(\alpha,k)}\,. \tag{5.3}$$

Proof. The aim is to minimize $\frac{1}{2}$trace$\left(\mathcal{D}\left(X_{A_k}\right)\right)$ subject to the constraint (5.2), where the constant $1/2$ is introduced to simplify calculations. The Lagrange function $\mathcal{L}(\theta_\alpha, \omega)$ with Lagrange multiplier ω is given by

$$\mathcal{L}(\theta_\alpha, \omega) = \frac{1}{2}\theta_\alpha^\top M_{(k)} \theta_\alpha - \omega\left(\theta_\alpha^\top S_{(\alpha,k)} - S_{\alpha-1/2}\right). \tag{5.4}$$

To minimize the Lagrange function, differentiate Equation (5.4) with respect to θ_α and set the result equal to 0, which gives:

$$M_{(k)}\theta_\alpha = \omega S_{(\alpha,k)}.$$

This can then be rearranged to find an estimator for θ_α:

$$\hat{\theta}_\alpha = \omega M_{(k)}^{-1} S_{(\alpha,k)}. \tag{5.5}$$

Let $\omega = \omega_{\alpha,k}$ to show the dependancy of the scalar on the parameters α and k. The value of $\omega_{\alpha,k}$ can be found by substituting (5.5) for θ_α into the constraint $\theta_\alpha^\top S_{(\alpha,k)} = S_{\alpha-1/2}$ and rearranging:

$$\omega_{\alpha,k} = \frac{S_{\alpha-1/2}}{S_{(\alpha,k)}^\top M_{(k)}^{-1} S_{(\alpha,k)}}.$$

Thus, the vector of coefficients which minimizes trace$\left(\mathcal{D}(X_{A_k})\right)$ subject to the constraint (5.2) is given by Equation (5.3). The polynomial with these coefficients is called the minimal-variance whitening polynomial, and the matrix produced by the polynomial is called the minimal-variance whitening matrix. □

An outline of how to produce and use the minimal-variance polynomial matrix to whiten a dataset is given in Algorithm 2.

Algorithm 2: Minimal-variance whitening method

Input: X: data; k: degree of polynomial; α: constraint parameter; μ: mean of X; Σ: covariance matrix of X

Output: X_{A_k}: transformed data

$$\hat{\theta}_\alpha = \frac{S_{\alpha-1/2}}{S_{(\alpha,k)}^\top M_{(k)}^{-1} S_{(\alpha,k)}} M_{(k)}^{-1} S_{(\alpha,k)}$$

$$A_k = \sum_{i=0}^{k-1} \theta_i \Sigma^i$$

$$X_{A_k} = A_k(X - \mu)$$

5.2.2 Parameter selection for minimal-variance whitening

The minimal-variance polynomial is reliant on the setting of two parameters. The parameter α relates to the constraint (5.2), which controls how the minimal-variance whitening matrix approximates $\Sigma^{-1/2}$. The parameter k controls the degree of the minimal-variance polynomial, which can be used as a trade-off between accuracy and time taken to calculate the polynomial. Both parameters will be explored in the following sections, and suggestions on how to set them are given.

Choice of the parameter α

The parameter α in used to set the constraint (5.2), and will affect the outcome of polynomial whitening. Theoretically, any value of α will produce an alternative whitening matrix to Σ^{-1}. Using $\alpha = 1$ produces the constraint

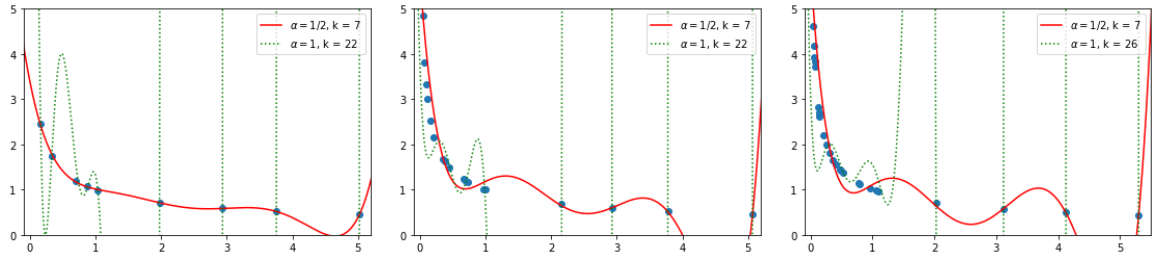
$$\operatorname{trace}(A_k\Sigma) = \operatorname{trace}\left(\Sigma^{1/2}\right),$$

while letting $\alpha = 1/2$ will give the constraint

$$\operatorname{trace}\left(A_k\Sigma^{1/2}\right) = \operatorname{trace}(I),$$

where I is the $d \times d$ identity matrix.

The outcomes of polynomial whitening using different values of α will be studied using empirical investigations. Figure 5.2 considers three different datasets, with $d = 10$, $d = 50$ and $d = 150$, respectively. The nonzero eigenvalues of the datasets (detailed in Appendix C.3) are plotted along the horizontal axes, and the reciprocal square root eigenvalues are plotted on the vertical axes. The minimal-variance whitening polynomials with different values of α are then plotted, using the following method.



(a) $d = 10$ (b) $d = 50$ (c) $d = 150$

Figure 5.2: Comparing the effect of constraints with $\alpha = 1/2$, $k = 7$ (red, solid line) and $\alpha = 1$, $k = 22$ (green, dotted line) on the fit of the minimal-variance polynomial to the inverse square root of the simulated eigenvalues (blue points). The eigenvalues are given in Appendix C.3, for datasets generated with (a) 10 dimensions, (b) 50 dimensions and (c) 150 dimensions.

Find the coefficients $\theta_\alpha = (\theta_0, \theta_1, \ldots, \theta_{k-1})^\top$ of the minimal-variance whitening polynomial using Equation (5.3), and write the polynomial as in Equation (5.1), replacing the matrix Σ with a symbol t:

$$p_k(t) = \theta_0 t^0 + \theta_1 t^1 + \cdots + \theta_{k-1} t^{k-1}. \tag{5.6}$$

This polynomial can then be plotted for different values of t. In Figure 5.2, the polynomials are plotted using $\alpha = 1$ and $\alpha = 1/2$, using the value of k that provided the best fit to the reciprocal square roots of the eigenvalues.

As demonstrated in Figure 5.2, using the parameter $\alpha = 1$ requires a much higher value of k to obtain the polynomial with the best fit. It has been shown in both Chapter 3 and Chapter 4 that using higher degree polynomials can induce instability, and therefore lower-degree polynomials are preferred. Furthermore, it can be shown for low values of the parameter k that θ_α is a vector with a scalar value in the first entry, and every other entry all zeros. See Appendix B.1 for a further explanation on this.

The polynomial with $\alpha = 1/2$ performed better in these experiments, in terms of data whitening success, stability and computational cost. This constraint works well in the case of non-degenerate data (when Σ is essentially non-singular). The method also performs well in the case of singular data, but requires an adjustment, much like the minimal-variance distances.

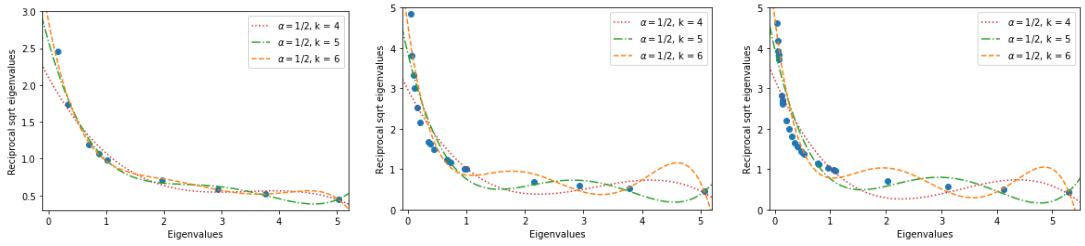
Using the parameter $\alpha = 1/2$ in the minimal-variance distances is equivalent to enforcing the constraint $\text{trace}\left(A_k\Sigma^{1/2}\right) = \text{trace}\,(I) = d$. It may be preferable to adjust the constraint to require $\text{trace}\left(A_k\Sigma^{1/2}\right) = r$, where r is the rank of the data. This constraint has been applied to all relevant examples, including those in Figure 5.2, and will be discussed in more detail in Section 5.2.3.

Choice of the parameter k

The true inverse square root of a full-rank covariance matrix can be written as a $(d-1)$-degree polynomial using the characteristic polynomial of the matrix. The minimal-variance whitening polynomial with parameter k forms a $(k-1)$-degree polynomial, as defined in Equation (5.1). As k increases, the polynomial can theoretically approximate the square root of the characteristic polynomial more accurately. However, not only is it more computationally intensive to compute a polynomial as the degree increases, but the opportunity for instability to occur is much greater, particularly in high dimensions (see the discussion in Section 4.2.4 and the examples given in Section 5.3.1). As such, keeping values of the parameter k relatively low is not only beneficial for cost, but for stability.

Figure 5.3 considers the same datasets as those in Figure 5.2, but uses parameters $\alpha = 1/2$ and $k = \{4, 5, 6\}$ to plot the polynomials. As can be seen by the polynomial fit to the reciprocal square roots of the eigenvalues in Figure 5.3, choosing low values of k can produce good approximations for the inverse square root of the covariance matrix. This will be further demonstrated in the numerical examples in Section 5.3.

To choose the best value of k, it may be appropriate to run the same experiment multiple times with different values of k and use a problem-specific metric to identify the best value of k for that dataset. For example, in Section 5.3.1, the Wasserstein metric is used to compare the whitened data to the standard normal distribution, as well as a sum-of-squares-based metric. The value of k chosen is the one which produces the lowest values for these metrics. This is similar to techniques used in many parameterized methods, such as using scree plots or silhouette scores to judge the best number of clusters to use in a clustering algorithm.



(a) $d = 10$ (b) $d = 50$ (c) $d = 150$

Figure 5.3: The minimal-variance polynomial fit to simulated eigenvalues (blue, values given in Appendix C.3) for datasets with (a) 10 dimensions, (b) 50 dimensions and (c) 150 dimensions. Parameters used are $\alpha = 1/2$ and $k = 4$ (red, dotted line), $k = 5$ (green, dash-dot line) and $k = 6$ (orange dashed line).

5.2.3 Constraint adjustment for rank-deficient data

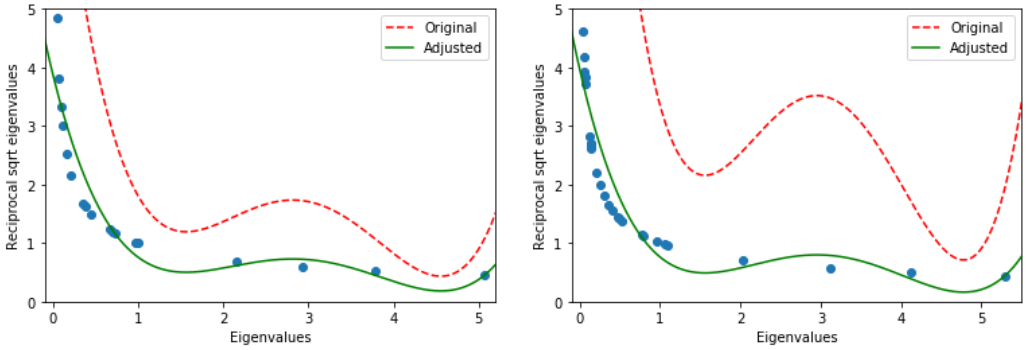
The Moore-Penrose pseudoinverse Σ^- has the property that

$$\text{trace}\left((\Sigma^-)^{1/2}\Sigma^{1/2}\right) = r, \tag{5.7}$$

where r is the rank of Σ. However, for matrices with many small eigenvalues, r is hard to calculate [241], and approximations of r are often based on arbitrary eigenvalue thresholding or subjective elbow plots [131]. In cases where Σ is not full-rank, an adjustment is proposed to modify the constraint (5.2) to be more similar to Equation (5.7). This discussion is analogous to the findings given in Section 4.3.2 regarding the minimal-variance distances.

Two examples are given in Figure 5.4 to illustrate the constraint adjustment, using the same datasets as in Section 5.2.2 (the eigenvalues of which can be found in Appendix C.3). The figures plot the eigenvalues of datasets against their reciprocal square root eigenvalues.

The minimal-variance polynomial using the original constraint (5.2) is shown in Figure 5.4a and Figure 5.4b as the red, dashed line. The method of plotting these polynomials is described in Section 5.2.2. Although the original polynomials take the correct shape, they are clearly placed too high and do not fit the plot of the reciprocal square root eigenvalues. Multiplying the polynomials by some constant c_k between 0 and 1 ensures a better fit of the polynomial. A method for finding an optimal value of c is as follows.



(a) $d = 50$ (b) $d = 150$

Figure 5.4: The minimal-variance polynomial with $k = 5$ fit to simulated eigenvalues (blue, the same as those in Figure 5.2) before (red, dashed line) and after (green, solid line) adjustment for rank-deficient data, as described in Section 5.2.3.

Let $\Lambda = \{\lambda_1, \ldots, \lambda_d\}$ be the set of all eigenvalues of Σ, and let $\tilde{\Lambda} = \{\lambda_i \in \Lambda : \lambda_i \neq 0\}$ be the set of all nonzero eigenvalues of Σ. In the case of very large dimensions, computation of eigenvalues λ_i is certainly out of reach; in this case, as will be discussed in Section 5.3.3, it is suggested to project the data to a low dimensional space and use the set of eigenvalues for the low dimensional projection of the data. The constant c_k can be found in any number of ways; here the goal is to minimize the distance between the minimal-variance polynomial $p_k(\lambda)$ (as in Equation (5.6)) and the target values $1/\lambda^{1/2}$, for $\lambda \in \tilde{\Lambda}$. Letting $w(\lambda)$ be a suitable weight function, the value c_k^* from

$$c_k^* = \underset{c_k \in (0,1]}{\arg\min} \sum_{\lambda \in \tilde{\Lambda}} w(\lambda)[c_k \cdot p_k(\lambda) - \lambda^{-0.5}]^2$$

minimizes the weighted sum of squares between the polynomial and the reciprocal square root of the nonzero eigenvalues. The optimal value of the adjustment constant c_k is then found to be

$$c_k^* = \frac{\sum_{\lambda \in \tilde{\Lambda}} w(\lambda)\lambda^{-0.5} p_k(\lambda)}{\sum_{\lambda \in \tilde{\Lambda}} w(\lambda) p_k(\lambda)^2}. \tag{5.8}$$

In Figure 5.4, the weight function $w(\lambda) = \lambda$ is used, and in general this weight function is recommended. However, the choice of $w(\lambda)$ can be altered to adjust the fit of the polynomial to the eigenvalues. If the user is more concerned about fitting the polynomial to the larger eigenvalues of the dataset, they may decide to use $w(\lambda) = \lambda^i$ with $i > 1$, for example.

The adjusted polynomials (given by the green solid line in Figure 5.4) clearly fit the desired points much more successfully than the original polynomials. However, if this adjustment is not performed, the data transformed by the minimal-variance whitening matrix A_k will still be approximately isotropic, as this adjustment is simply a multiplication of the whitening matrix by a scalar. For simplicity and continuity, this adjustment has been applied to all relevant examples that follow.

This adjustment to the constraint can also be used to detect the singularity of a matrix. Consider first the case with $d < N$. If Σ is full rank, and k is chosen appropriately, the value c_k^* will be equal to (or very close to) 1, as the minimal-variance polynomial is aiming to make $\text{trace}(A_k\Sigma) = d$, which is correct in the case of full-rank Σ. If the matrix Σ is not full-rank, c_k^* will be less than 1.

To illustrate this, Table 5.1 gives two $d < N$ examples. A d-dimensional dataset with N observations is generated using a covariance matrix with rank R. Further details on how these datasets were generated is available in Appendix C.3. The empirical covariance matrix of a dataset has rank $r = \min(d, N, R)$, and is used to find the minimal-variance whitening matrix with $k = 10$. Table 5.1 gives details of the dataset, as well as the constraint adjustment c_{10}^* from Equation (5.8). In dataset 1, the empirical matrix has full rank $r = d$, so $c_{10}^* = 1$. In dataset 2, the 'true' covariance matrix has rank $R = 50$, $d = 100$ and $N = 1000$, therefore the empirical covariance matrix has rank $r = \min(d, N, R) = 50$. This produces a constraint adjustment value of $c_{10}^* = 0.50 < 1$, implying that the empirical covariance matrix Σ is singular.

Three examples are used to consider datasets with $d > N$. Dataset 3 in Table 5.1 has 100 dimensions and only 50 observations. The 'true' covariance matrix used to generate this dataset is full-rank $R = 100$, but the empirical covariance matrix has rank $r = \min(d, N, R) = 50$. Therefore, the adjustment value is $c_{10}^* = 0.50$, indicating that this dataset is degenerate. Dataset 4 also has $d = 100$, $N = 50$, and the 'true' covariance matrix now has rank $R = 50$. The adjustment value is therefore less than 1: $c_{10}^* = 0.50$. The final example considered here has $d = 100$, $N = 50$, but the 'true' covariance matrix has rank $R = 30$. The empirical covariance matrix therefore has $r = \min(d, N, R) = 30$, and the adjustment value is $c_{10}^* = 0.30$. In all the examples with $d > N$, $c_{10}^* < 1$, as the empirical covariance matrix Σ will never be full-rank in datasets with $d > N$.

Dataset	d	N	R	r	c^*_{10}
1	100	1000	100	100	1.00
2	100	1000	50	50	0.50
3	100	50	100	50	0.50
4	100	50	50	50	0.50
5	100	50	30	30	0.30

Table 5.1: The adjustment value c^*_{10} for different configurations of the dimension d, number of observations N, rank of true population covariance matrix R and rank of sample covariance matrix r.

The rank of a covariance matrix is not always obvious due to the presence of small eigenvalues, particularly in high dimensions. Section 5.5 discusses how the adjustment value c^*_k can be used to approximate the rank of the covariance matrix.

5.3 Applications of minimal-variance whitening

Data whitening is used in many applications across multivariate data analysis, as it has been shown to improve both computation time and performance [112, 133]. In some applications, data whitening is desirable as it has been shown to improve results [111], but is not used due to the computational cost of whitening a dataset [118], or the inability to whiten a dataset due to degeneracy [121]. In such cases, the minimal-variance whitening method is extremely useful, as it is computationally inexpensive and can be used to whiten degenerate datasets.

This section gives an overview of the performance of the minimal-variance whitening method, with applications on datasets with both $d < N$ and $d \geq N$. The method is compared to other popular whitening methods in Section 5.3.2, and several considerations as to how to apply the method to extremely high dimensional data are given in Section 5.3.3. Sections 5.3.4 and 5.3.5 compare minimal-variance whitening to other pre-processing methods in the context of outlier detection and principal component analysis, respectively.

5.3.1 Whitening data using minimal-variance polynomials

The first set of numerical examples illustrate the performance of minimal-variance whitening on several synthetic and real datasets. Datasets with $d < N$ will be considered first, followed by examples using datasets with dimension greater than the number of observations.

Data with d < N

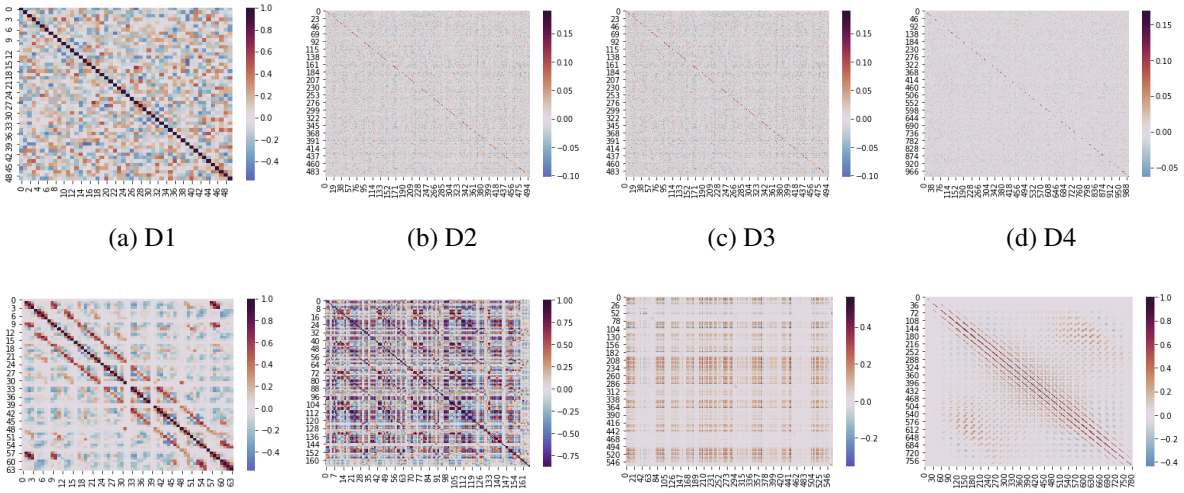
Table 5.2 gives the details of eight datasets with $d < N$, four of which are real datasets and the other four have been generated synthetically. The four synthetic datasets (D1, D2, D3, D4) are sampled from a Gaussian distribution $\mathcal{N}_d(0, \Sigma)$ with $N = 5 \times d$ observations, where the covariance matrices Σ are produced as follows. Generate d eigenvalues $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_d\}$ from the Wishart distribution and produce a random $d \times d$ orthogonal matrix Q. Let L be the matrix with the eigenvalues Λ on the diagonal and zeros elsewhere, then $\Sigma = Q^\top L Q$. See Appendix C.1 for more information.

The first three real datasets 'Digits', 'Musk' and 'HAR' (Human Activity Recognition) [18] were obtained from the UCI Machine Learning repository [68]. The 'MNIST' dataset [144] was obtained from the OpenML database [237].

Dataset	d	N
D1	50	250
D2	100	500
D3	500	2500
D4	1000	5000
Digits	64	1797
Musk	168	6598
HAR	561	10299
MNIST	784	70000

Table 5.2: Datasets used in Section 5.3.1 with $d < N$, their dimension d and number of observations N. The distribution of eigenvalues for each dataset is available in Appendix C.3.

The heatmaps in Figure 5.5 show the covariance matrices of the datasets, and the distribution of the eigenvalues of these covariance matrices are given in Appendix C.3. As Figure 5.5 shows, most of these datasets are highly correlated, with densely populated off-diagonal entries. In some cases, it can be beneficial to rescale the data so that each variable has zero mean and unit variance, before finding the minimal-variance polynomial matrix. If rescaling the data provides less extreme eigenvalues in the covariance matrix, this scaling is likely to improve the performance of the polynomial whitening. If a dataset has been rescaled, this is noted in the caption of Figure 5.5.



(a) D1 (b) D2 (c) D3 (d) D4

(e) Digits (f) Musk (g) HAR (h) MNIST

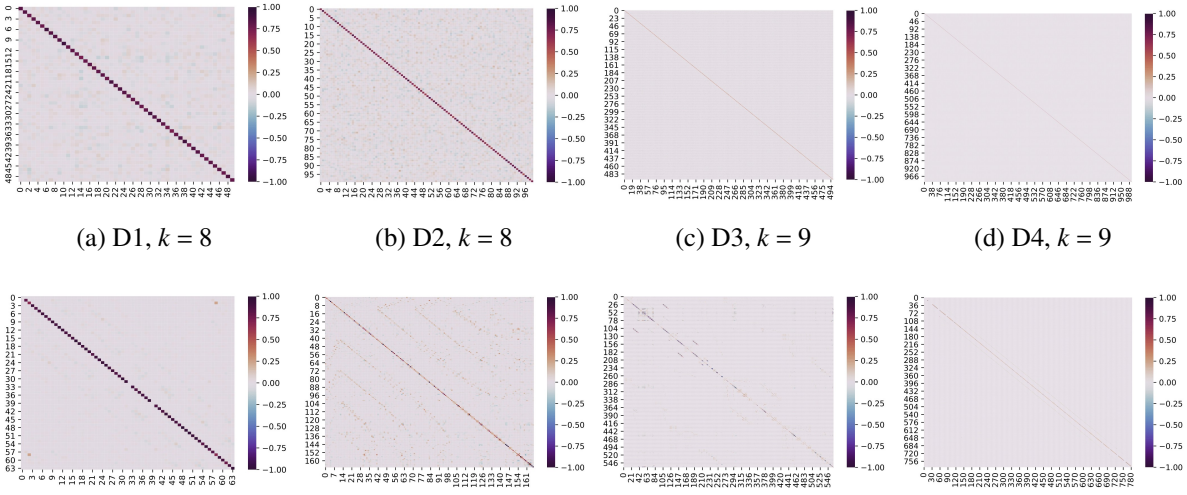
Figure 5.5: Heatmaps of the covariance matrix of each dataset detailed in Table 5.2 **before minimal-variance polynomial whitening**. Datasets corresponding to Figures (a), (e), (f) and (h) are scaled to have unit variance, to improve performance of polynomial whitening. The heatmaps show the covariance matrix after this scaling.

The proximity of the transformed data $X_{A_k} \sim \mathcal{N}_d(0, \mathcal{S})$ to the standard normal distribution $\mathcal{N}_d(0, I)$ can be measured using the Wasserstein metric [86]:

$$W(X_{A_k}) = \left(d + \text{trace}(\mathcal{S}) - 2\text{trace}\left(\mathcal{S}^{1/2}\right)\right)/d, \tag{5.9}$$

where the division by d accounts for the difference in the dimensions of each dataset.

The heatmaps in Figure 5.6 show the covariance matries of each dataset after whitening $X_{A_k} = A_k(X - \mu)$, illustrating that the correlations between variables have been approximately whitened. The value of k used in these heatmaps is chosen as the value of k which gives the lowest of the Wasserstein scores, which are given in Table 5.3.



(a) D1, $k = 8$ (b) D2, $k = 8$ (c) D3, $k = 9$ (d) D4, $k = 9$

(e) Digits, $k = 7$ (f) Musk, $k = 5$ (g) HAR, $k = 5$ (h) MNIST, $k = 5$

Figure 5.6: Heatmaps of the covariance matrix of the datasets in Table 5.2 **after minimal-variance polynomial whitening**. The value of k used in constructing the minimal-variance polynomial is given in the caption for each dataset.

The Wasserstein scores in Table 5.3 show that, in general, as the value of k increases, the transformed data is closer to the standard normal distribution, as desired. In some cases, such as the Musk dataset, higher values of k begin to show an increase in the Wasserstein score, indicating the decorrelation is less successful than when using lower values of k. This is due to numerical instability, as the minimal-variance polynomial aims to fit itself to extremely small eigenvalues, causing erratic behaviour in the polynomial. As such, it is recommended to use lower values of k which provide a more reliable alternative to the inverse square root of the covariance matrix, or to compute several minimal-variance polynomial matrices for different k and use the one that best satisfies some metric, such as the Wasserstein score.

The Wasserstein metric concerns itself only with the diagonal values of the covariance matrix, as it is calculated using traces. It can be considered as a measure of standardization, rather than whitening. A measure is therefore needed to evaluate the extent to which the data has been decorrelated. The heatmaps in Figure 5.6 show that the off diagonals of the covariance matrix of the transformed data are close to zero, indicating good decorrelation. Another way decorrelation can be measured is by considering the sum of squares of the off-diagonal entries of the covariance matrix (SSOD score) of the transformed data. In Table 5.4, let $\mathscr{S}_{A_kX}$ be the SSOD score of the whitened dataset $A_k(X-\mu)$.

Dataset	W_X	W_{A_3X}	W_{A_4X}	W_{A_5X}	W_{A_6X}	W_{A_7X}	W_{A_8X}	W_{A_9X}	$W_{A_{10}X}$
D1	0.455	0.179	0.119	0.090	0.075	0.060	**0.057**	0.071	0.139
D2	0.634	0.358	0.301	0.246	0.220	0.181	**0.160**	0.200	0.225
D3	0.866	0.718	0.678	0.631	0.601	0.578	0.544	**0.520**	0.552
D4	0.812	0.585	0.524	0.465	0.425	0.393	0.360	**0.336**	0.365
Digits	0.361	0.137	0.101	0.073	0.066	**0.058**	0.071	0.107	0.381
Musk	0.949	0.574	0.450	**0.373**	1.123	2.022	0.989	0.990	0.991
HAR	0.885	0.772	0.794	**0.586**	3.892	0.998	0.998	0.998	0.998
MNIST	0.612	0.405	0.341	**0.296**	0.597	1.077	1.039	1.566	4.563

Table 5.3: The Wasserstein scores (5.9), denoted W_{A_kX}, which measure the distance between the polynomial-whitened dataset $A_k(X-\mu)$ and the standard normal distribution $\mathcal{N}(0,I)$ for each dataset in Table 5.2. Values in bold indicate the lowest Wasserstein score W_{A_kX} over all k for a given dataset.

Dataset	$\mathscr{S}_X$	$\mathscr{S}_{A_3X}$	$\mathscr{S}_{A_4X}$	$\mathscr{S}_{A_5X}$	$\mathscr{S}_{A_6X}$	$\mathscr{S}_{A_7X}$	$\mathscr{S}_{A_8X}$	$\mathscr{S}_{A_9X}$	$\mathscr{S}_{A_{10}X}$
D1	10.10	2.89	2.80	2.44	2.12	2.10	**1.74**	1.85	2.97
D2	10.88	6.96	5.87	5.83	5.12	5.09	5.05	**4.72**	5.78
D3	20.04	19.69	18.55	18.44	16.14	15.76	15.65	**15.39**	16.44
D4	31.88	21.60	21.38	21.09	20.89	20.24	20.46	19.97	**19.24**
Digits	11.10	2.64	2.12	1.96	1.47	**1.18**	1.67	1.88	4.00
Musk	58.27	5.03	6.97	**6.64**	31.33	127.56	0.64	0.51	0.41
HAR	33.84	3.10	**1.35**	1.44	20.43	1.39	1.46	1.53	1.61
MNIST	74.75	11.02	11.02	**10.61**	13.61	58.67	38.45	280.83	1661.62

Table 5.4: The SSOD score, denoted $\mathscr{S}_{A_kX}$, of the polynomial-whitened dataset A_kX for each dataset in Table 5.2. Values in bold indicate the lowest value of $\mathscr{S}_{A_kX}$ over all k for a given dataset.

The sum of squares values in Table 5.4 decrease as k increases, until a certain value of k, much like the Wasserstein scores. The value of k that gives the optimum (smallest) SSOD score for each dataset is close to value of k that gives the optimum Wasserstein score for each dataset. Therefore, when the data has been successfully standardized, it has also been decorrelated well. Between the Wasserstein scores in Table 5.3, the SSOD scores in Table 5.4 and the heatmaps in Figure 5.6, it is evident that the minimal-variance method is able to produce an effective alternative to the inverse square root of the covariance matrix using a polynomial of degree significantly lower than the dimension of the dataset.

Table C.4 in Appendix C.3 shows the average time taken to produce the minimal-variance polynomial matrices for each dataset for each value of k considered, over 100 runs. The time taken increases as the dimensionality d of the dataset and the parameter k increase, but this only ever takes a matter of seconds, even for 1000-dimensional datasets.

Data with d > N

As discussed in the literature review, it is increasingly common for data to have higher dimensionality than number of observations in many fields, such as genetic microarrays, medical imaging and chemometrics [95]. Data with $d > N$ is clearly rank-deficient, with rank $r \leq N < d$, and thus the sample covariance matrix of such data is always singular, rendering many multivariate data analysis methods unusable, including data whitening. Minimal-variance polynomial whitening is applicable in such cases, as illustrated by the following examples.

Four synthetically generated datasets and four real datasets are given in Table 5.5. The first two synthetic datasets, E1 and E2, are sampled from a Gaussian distribution $\mathcal{N}_d(0, \Sigma)$, where the covariance matrices Σ are produced in the same way as detailed in Section 5.3.1. The third synthetic dataset, E3, is generated to copy the example in [244]: a multivariate Gaussian is generated using a population covariance matrix with diagonal entries $[50, 20, 10] + [1] * 47$. This creates a spiked eigenvalue model, which is of interest in 'high dimensional low sample size' (HDLSS) datasets [19]. The fourth dataset uses a covariance matrix with eigenvalues generated from a random uniform distribution between 0 and 1, to produce a non-sparse set of eigenvalues.

The madelon dataset was obtained from the UCI Machine Learning Repository [68]. The

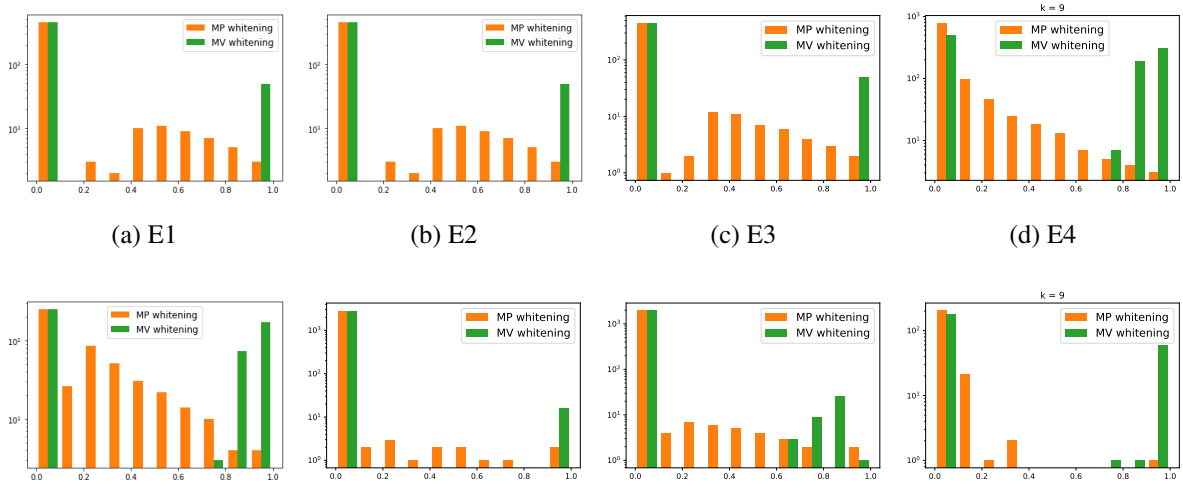
Dataset	d	N
E1	500	50
E2	1000	50
E3	500	50
E4	1000	500
Madelon†	500	250
Yeast	2884	17
Colon	2000	40
DB-emails	242	64

Table 5.5: Datasets used in Section 5.3.1, their dimension d and number of observations N. The madelon† dataset is a subsample of the true madelon dataset, with only the first 250 observations considered.

raw madelon dataset has 4400 observations, greater than the 500 features, so only the first 250 observations are used to create the madelon† dataset with $d > N$. The yeast dataset is a real genomic dataset with 2284 features and 17 observations [229, 237]. The third real dataset is another genomic dataset on colon cancer data [11], used by [263] as an example of a spiked eigenvalue model. This dataset includes two clusters which represent tumorous and non-tumorous colons; only the former cluster is considered here. The DB-emails dataset is a 'bag-of-words' representation of a collection of emails [76]. Note that the madelon† , yeast and colon datasets have been scaled to have unit variance. The empirical eigenvalues of all datasets are given in Appendix C.3.

Successful whitening of these datasets would result in a covariance matrix with r eigenvalues equal to 1, and $d - r$ eigenvalues equal to 0. Moore-Penrose Mahalanobis (MPM) whitening is performed on the four datasets in Table 5.5 by pre-multiplying the data by the square root of the Moore-Penrose pseudoinverse of the covariance matrix. The datasets are then whitened using the minimal-variance method as described in Section 5.2.1.

Figure 5.7 compares the distribution of the eigenvalues of the covariance matrices after MPM whitening and minimal-variance polynomial whitening. The eigenvalues are scaled such that the maximum eigenvalue is equal to 1. The first three synthetic datasets show that using minimal-variance whitening returns a dataset with eigenvalues only equal to 0



(a) E1 (b) E2 (c) E3 (d) E4

(e) Madelon† (f) Yeast (g) Colon (h) DB-emails

Figure 5.7: Log-scale histograms, showing the eigenvalues of the covariance matrix after the data has been whitened by Moore-Penrose (MP) Mahalanobis whitening (orange histogram), and minimal-variance polynomial (MV) whitening with $k = 9$ (green histogram), for each of the datasets given in Table 5.5.

and 1, whereas using MPM whitening gives a dataset with a spread of eigenvalues between 0 and 1. Figure 5.7d shows that minimal-variance whitening may not achieve perfect whitening, but that it is still more successful than the MPM whitening method.

Figure 5.7f considers the yeast dataset, and shows that the MPM whitening method does not perfectly whiten the data, while minimal-variance whitening returns only eigenvalues of value 0 or 1. The madelon† , colon and DB-emails datasets are not whitened perfectly by either method, but the eigenvalues are much more dispersed when using MPM whitening compared to minimal-variance whitening, whereas the eigenvalues should only be valued at 0 and 1, ideally. In all examples considered, the minimal-variance whitening method outperforms Moore-Penrose Mahalanobis whitening, even if it does not whiten the dataset perfectly.

In Section 5.4.2, the real datasets used here will again be considered to show that some of the above performances can be improved upon further by using an iterative minimal-variance whitening method.

5.3.2 Comparison to other whitening methods

Due to rotational freedom, there are infinitely many whitening matrices of the form $W = Q\Sigma^{-1/2}$, where Q is orthogonal and satisfies $Q^\top Q = I$ [130]. In the following examples, some of these different whitening matrices are compared to the minimal-variance whitening matrix.

There are many possible decompositions of the covariance matrix Σ, including the following:

- $\Sigma = V^{1/2}PV^{1/2}$, where V is the diagonal variance matrix and P is the correlation matrix,
- $\Sigma = U\Delta U^\top$, the eigendecomposition of the covariance matrix, with U the matrix of eigenvectors and Δ the diagonal matrix of eigenvalues.

Analogously, define the eigendecomposition $P = GOG^\top$ of the correlation matrix, where G is the matrix of eigenvectors of P and O is the diagonal matrix of eigenvalues. The Cholesky decomposition of the inverse covariance matrix is also defined as $LL^\top = \Sigma^{-1}$, when Σ^{-1} exists.

Five whitening procedures are given by Kessy et al. [130] to be unique in fulfilling different objective functions. Let W be a whitening matrix, and therefore let $X_W = W(X - \mu)$ be the transformed data. Most of the objective functions identified in the paper [130] are based on the cross-covariance matrix Φ and the cross-correlation matrix Ψ between the original data X with covariance Σ and the whitened data X_W:

$$\Phi = \operatorname{cov}(X_W, X) = W\Sigma,$$
$$\Psi = \operatorname{corr}(X_W, X) = \Phi V^{-1/2}.$$

In the following example, minimal-variance whitening is compared to three of these whitening procedures. The so-called 'PCA' and 'PCA-cor' whitening methods detailed in [130] are not considered, as these methods aim to maximize compression of variance into the first few variables of the whitened data. Although minimal-variance whitening performed relatively well in these scenarios, this is not the aim of the method. The three types of whitening considered alongside polynomial whitening are given below.

Mahalanobis whitening (MW): $W = \Sigma^{-1/2}$. Mahalanobis whitening is found to be the unique whitening procedure which maximizes **trace**(Φ), the average cross-covariance between each variable of the original and the newly transformed data. This is equivalent to minimizing the total squared distance between the original data X and the whitened data X_W, ensuring the whitened data is as similar as possible to the original data.

Mahalanobis-cor whitening (MCW): $W = P^{-1/2}V^{-1/2}$. Mahalanobis whitening can be affected by the different scales of the variables, so to avoid this issue the scale-invariant version can be used, known as Mahalanobis-correlation whitening. Mahalanobis-correlation whitening maximizes the cross-correlation **trace**(Ψ) between each variable of the standardized original data $V^{-1/2}X$ and the whitened data X_W. Doing this is shown to be equivalent to minimizing the squared distance between $V^{-1/2}X$ and X_W.

Cholesky whitening (CW): $W = L^\top$. Cholesky whitening is the only whitening procedure fulfilling the constraint of producing lower-triangular cross-covariance and cross-correlation matrices with positive diagonal entries. It does not result from fulfilling an objective function like the above methods, but rather from satisfying this constraint.

The performance of these different whitening procedures is evaluated by applying them to a dataset and considering the different objective functions in Φ and Ψ. First, as in [130], the whitening methods are applied to the 4-dimensional Iris dataset [79] in Table 5.6. Given the dataset's low dimension and well-conditioned covariance matrix, minimal-variance polynomial whitening (**MVW** in the table) with $k = d = 4$ produces exactly the same results as Mahalanobis whitening. Minimal-variance-cor whitening (**MVCW**) is also performed, where the data is standardized and minimal-variance polynomial whitening is performed using the correlation matrix P. This produces the same results as Mahalanobis-cor whitening.

The minimal-variance whitening method is more effectively used when applied to higher dimensional datasets with singular or near-singular covariance matrices. As such, the above exercise is repeated with a different dataset. For the purposes of this example, it is not possible to use a dataset which has a singular covariance matrix, as the Mahalanobis and Cholesky whitening methods are not usable in this case. The Wisconsin Breast Cancer dataset [253] is used, and has been pre-standardized to give improved results from all

	MW	MCW	CW	MVW $k = 4$	MVCW $k = 4$
$\text{tr}(\hat{\phi})$	**2.9829**	2.8495	1.9369	**2.9829**	2.8495
$\text{tr}(\hat{\psi})$	3.0742	**3.1914**	2.5331	3.0742	**3.1914**

Table 5.6: A comparison of different whitening methods applied to the Iris dataset, using metrics identified in [130]. Bold entries identify the best result for each metric.

	MW	MCW	CW	MVW $k = 6$	MVCW $k = 6$
$\text{tr}(\hat{\phi})$	21.0193	21.1282	14.5409	**24.8036**	23.1984
$\text{tr}(\hat{\psi})$	20.9651	21.0737	14.5034	21.7396	**24.7396**

Table 5.7: A comparison of different whitening methods applied to the Wisconsin Breast Cancer dataset, using metrics identified in [130]. Bold entries identify the best result for each metric.

methods. This dataset has dimension $d = 32$ and has a covariance matrix which could be considered ill-conditioned (see Appendix C.3 for details on the eigenvalues). Table 5.7 shows that minimal-variance whitening outperforms Mahalanobis whitening, using both the covariance and correlation matrix.

The minimal-variance polynomial method is therefore a good alternative to the aforementioned established whitening methods, particularly when the dataset is near-degeneracy. If the dataset has a singular covariance matrix, the other methods are not usable. Section 5.3.1 has already shown that minimal-variance polynomial whitening often performs better than Moore-Penrose Mahalanobis whitening, so the minimal-variance polynomial whitening method would be recommended in such cases.

5.3.3 Applications to extremely high dimensional data

Given a dataset X with extremely high dimension d, say $d = 1{,}000{,}000$, finding the minimal-variance polynomial matrix can be too costly and time-intensive. Typical dimension reduction methods such as principal component analysis are also prohibitively expensive in high dimensions. Instead, some other methods are suggested here to allow application of the minimal-variance polynomial in very high dimensional data.

Sampling variables

A simple method to reduce computational time is to sample some variables from X to produce a 'representative' dataset $\tilde{X}$ in a much smaller dimension $\tilde{d}$. This representative dataset can be found through random samples of the variables in X, or projection to a lower dimensional space [36, 38]. The covariance matrix $\tilde{\Sigma}$ of $\tilde{X}$ can be found and used to produce the minimal-variance polynomial alternative to $\tilde{\Sigma}^{-1/2}$:

$$\tilde{A_k} = \theta_0 I + \theta_1 \tilde{\Sigma} + \ldots + \theta_{k-1} \tilde{\Sigma}^{k-1}. \tag{5.10}$$

The $\tilde{d}$-dimensional matrix $\tilde{\Sigma}$ in Equation (5.10) can then be replaced with the d-dimensional covariance matrix Σ to obtain the minimal-variance polynomial matrix A_k, using the coefficients obtained to find $\tilde{A_k}$. This can be used to whiten the original high dimensional dataset X, and is much cheaper than finding the minimal-variance matrix directly.

Random projection

Alternatively, random projections of the dataset can be used while largely preserving pairwise distances between points. Let $f : \mathbb{R}^d \to \mathbb{R}^p$ be a projection where $p < d$. Choose p Gaussian vectors $\{u_1, u_2, \ldots, u_p\} \in \mathbb{R}^d$ with unit-variance coordinates. For any vector $x \in X$, define the projection

$$f(x) = \Big(u_1 \cdot x, u_2 \cdot x, \ldots, u_p \cdot x\Big).$$

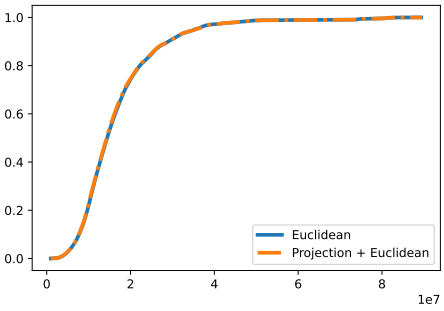
It can be shown that $\|f(x)\|_2 \approx \sqrt{p}\|x\|_2$ [38], where $\|x\|_2$ is the ℓ_2-norm of the vector x.

The Johnson-Lindenstrauss lemma states that any high dimensional dataset can be randomly projected using the above method while controlling the distortion in the pairwise distances between points. That is, for two points $x_1, x_2 \in X$ and a user-defined maximum distortion rate of $\epsilon \in (0, 1)$:

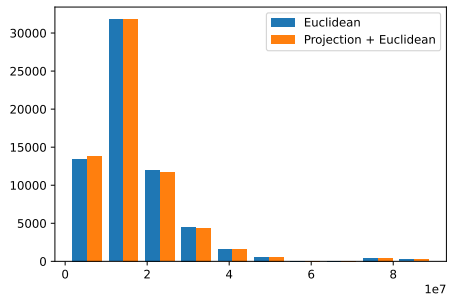
$$(1-\epsilon)\|x_1 - x_2\|^2 < \|f(x_1) - f(x_2)\|^2 < (1+\epsilon)\|x_1 - x_2\|^2.$$

The minimum number of dimensions p needed to guarantee a maximum distortion of ϵ is based only on the number of observations N, and is given by [61]:

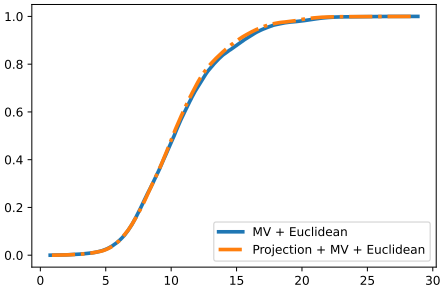
$$p \geq \frac{4\ln(N)}{(\epsilon^2/2) - (\epsilon^3/3)}.$$



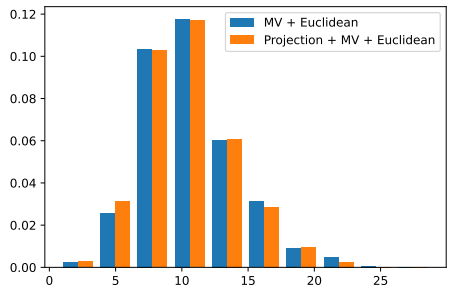
(a) Euclidean CDFs



(b) Euclidean histogram



(c) MV + Euclidean CDFs



(d) MV + Euclidean histogram

Figure 5.8: The CDFs and histograms of the distances produced between all points in the Micro-Mass dataset before and after random projection. Figures (a) and (b) consider only the Euclidean distance, Figures (c) and (d) consider the Euclidean distance after minimal-variance whitening has been applied.

An example is given which illustrates that random projection has minimal effect on the distribution of the Euclidean distances produced when using minimal-variance whitening. The dataset used is called 'Micro-Mass' [164] and represents different strains of bacteria. This dataset has 1300 dimensions and 360 observations. Although this dataset does not have a huge number of dimensions, it allows for computations of the minimal-variance whitening matrix before and after random projection for the sake of comparison.

Let $\epsilon = 0.3$, giving a minimum projection dimension $p = 654$. The dataset is projected from $d = 1300$ to $p = 654$ using the Python function `GaussianRandomProjection` from the `random_projection` package of Scikit-Learn [185]. All pairwise Euclidean distances are found before and after this projection, and the cumulative distribution functions (CDFs) of these distances is given in Figure 5.8a, showing the distances produced are very similar. Figure 5.8b also shows this, giving the histograms of the distances before and after the projection.

The blue line plotted in Figure 5.8c shows the CDF of the Euclidean distances found after performing minimal-variance whitening on the original 1300-dimensional Micro-Mass dataset. To test if random projection performs well with minimal-variance whitening, the data is first projected to $p = 654$. Minimal-variance whitening is then applied to the 654-dimensional dataset, and the Euclidean distances are found after this. The CDF of these distances is plotted in orange in Figure 5.8c, and shows very little difference to the non-projected distances. The comparison of distances when using minimal-variance whitening is also shown by the histogram in Figure 5.8d.

Approximating the distribution of eigenvalues

For large datasets, it may be that the eigenvalues of the covariance matrix are not known exactly, but their distribution can be approximated. If this is the case, d eigenvalues can be sampled from this distribution using the inverse cumulative distribution function and used to form an estimation of the covariance matrix. This is illustrated in Figure 5.9 using the Marchenko-Pastur distribution, as this distribution is known to model the eigenvalues of the sample covariance matrix of a random matrix as $d, N \to \infty$ and $\frac{d}{N} < 1$. Figure 5.9 considers an example with $d = 10,000$ and $N = 15,000$, and the probability density function (PDF) of the Marchenko-Pastur distribution with these parameters is shown by the red line. The histogram represents a random sample of 300 eigenvalues, and shows such a sample models the distribution well.

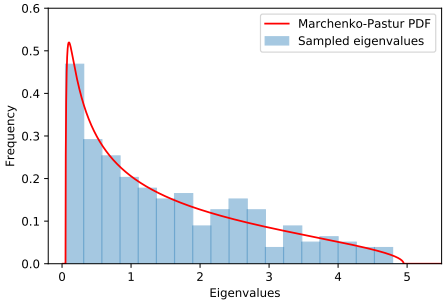


Figure 5.9: Sampling of eigenvalues from the Marchenko-Pastur distribution. The red line indicates the Marchenko-Pastur PDF, when $d = 10,000$ and $N = 15,000$. The histogram shows the spread of the 300 sampled values from this distribution.

Approximation of traces

One of the more time-consuming parts of computing the minimal-variance polynomial for high dimensional data is the need to compute the $S_j = \text{trace}\left(\Sigma^j\right)$ values, which are used to compute the coefficient vector of the polynomial.

The S_j values can be approximated using a range of methods to reduce computation time, including stochastic trace estimators [114], methods estimating Gauss-type quadrature formulas through Lanczos approximation process [89, 233] and a Chebyshev kernel polynomial method [21]. Further information and references on methods of approximating traces of functions of matrices can be found in [234].

5.3.4 The effect of different pre-processing methods on outlier detection algorithms

Outlier detection algorithms often require data to be pre-processed before the algorithm can be applied. It has been shown by Campos et al. [51] that the normalization of datasets will often lead to a better performance of outlier detection algorithms.

In this section, a study described in [127] is replicated. The authors produced a collection of labelled benchmark datasets to be used for evaluating outlier detection algorithm performance. They evaluated the performance of various algorithms when used after applying different normalization methods to these datasets. Performance of an algorithm was measured using the area under the receiver operator characteristic curve (AUC), which compares the labels of an observation ('inlier' or 'outlier') produced by the algorithm to the 'true' labels. They found that two types of normalization method performed differently (dependent on data set and outlier detection method):

'Min-Max' normalization: Each variable v of a dataset is normalized to only have values in the range [0, 1]: $\frac{v-\min(v)}{\max(v)-\min(v)}$, where $\min(v)$ and $\max(v)$ are the minimum and maximum values of the variable v, respectively.

'Median-IQR' normalization: Each variable v is transformed to $\frac{v-\text{median}(v)}{\text{IQR}(v)}$, where $\text{median}(v)$ and $\text{IQR}(v)$ are the median and inter-quartile range of the variable v, respectively.

The following four different outlier detection methods from the Python package PyOD [271] are considered here:

1. KNN: K-Nearest Neighbours
2. LOF: Local Outlier Factor
3. COF: Connectivity-based Outlier Factor
4. FastABOD: Fast Angle Based Outlier Detection

Further details of each of these algorithms are provided in [51]. All of the above methods require a parameter choice K (different to the polynomial degree parameter k referred to throughout this paper) to set the so-called neighbourhood size, and a contamination value C to indicate how many observations the algorithm should label as outliers. Let $K = 0.1 \times N$, where N is the number of observations in the dataset D. The parameter C is equal to the number of outliers given by the 'true' labels.

For a dataset D, an outlier detection method o and a pre-processing method z, denote the area under the receiver operating characteristic curve (AUC) as $AUC(D,o,z)$. A receiver operating characteristic curve is a graph used to show the performance of a classification model; the higher the AUC score, the better the classifier has performed [106]. For each outlier detection method o listed above, it is said a dataset D 'prefers' a pre-processing method z if $AUC(D,o,z) \geq AUC(D,o,y)$ for all other pre-processing methods y. The AUC score is evaluated for transformations $A_k D$ using Equation (5.3) by taking the maximum AUC score over all values of k considered.

Outlier Detection Method	Min-Var	Min-Max	Median-IQR
KNN	40.12%	30.70%	29.17%
LOF	41.29%	30.09%	28.61%
COF	42.26%	29.39%	28.34%
FastABOD	39.17%	31.16%	29.67%

Table 5.8: The percentage of the 7667 datasets considered that give higher AUC scores for the pre-processing technique (given in the column), by outlier detection method (given in the row).

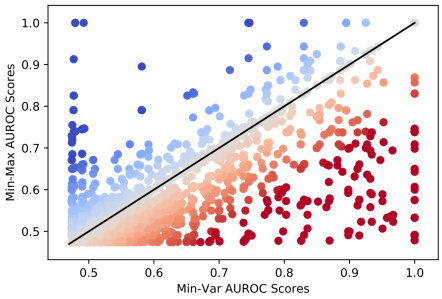
The outlier detection methods are tested with each pre-processing method on 7667 real datasets, as used in [127]. The datasets range from dimension 3 to dimension 359, and the number of observations in a dataset ranged from 44 to 5396. Table 5.8 shows the percentage of datasets that prefer each pre-processing method for each of the given outlier detection algorithms. The results in this table indicate that the polynomial whitening method outperforms the two normalization methods.

The scatter graphs in Figure 5.10 compare the minimal-variance polynomial whitening to the normalization methods considered individually. Each point represents a dataset, and the diagonal line indicates those datasets where the two methods give equal AUC scores. Points below this line, in red, indicate that the minimal-variance whitening method outperformed the other method considered. A numerical breakdown of these scatter graphs is given in Table 5.9.

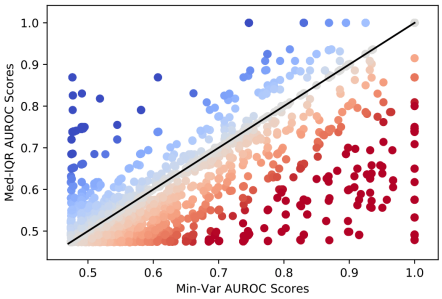
	Min-Var vs Min-Max			Min-Var vs Med-IQR		
	Min-Var	Min-Max	Equal	Min-Var	Med-IQR	Equal
KNN	34.4%	15.9%	49.7%	35.8%	14.0%	50.1%
LOF	37.3%	14.1%	48.6%	38.1%	12.8%	49.2%
COF	40.6%	16.1%	43.3%	41.8%	14.8%	43.4%
FastABOD	32.2%	15.1%	52.7%	33.8%	12.8%	53.4%

Table 5.9: The percentage of datasets for which the given pre-processing method produces AUC scores better than the alternative method in the column, for different outlier detection methods (given in the row). In particular, 34.4% of datasets produced higher AUC scores when using Min-Var than when using Min-Max, for the KNN outlier detection method.

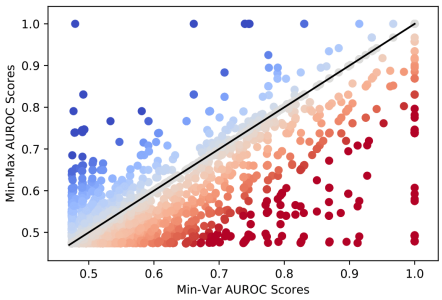
Table 5.9 highlights that, for a lot of datasets, the pre-processing methods considered here often produce equal AUC scores. Table 5.10 shows the amount of datasets out of the total 7667 (and the percentage) for which the pre-processing methods produce *strictly* better results, for each outlier detection method. It is clear that the minimal-variance method performs as well as (and often better than) the techniques often used to preprocess datasets before applying common outlier detection methods.



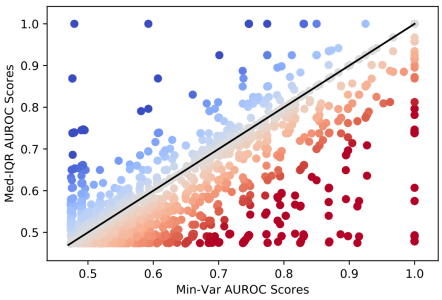
(a) KNN: Min-Var vs Min-Max



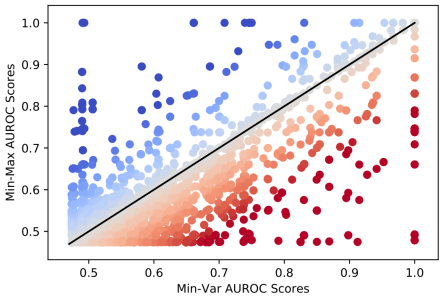
(b) KNN: Min-Var vs Med-IQR



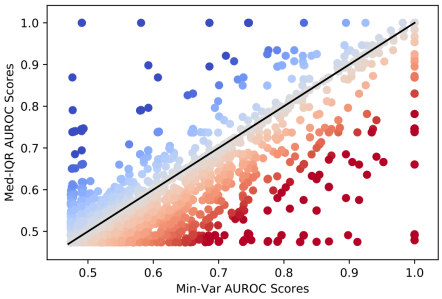
(c) LOF: Min-Var vs Min-Max



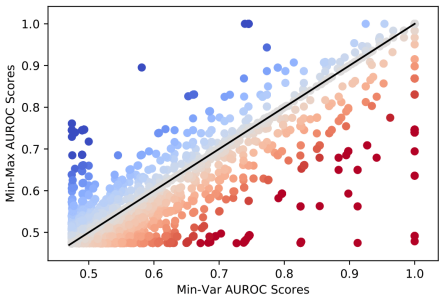
(d) LOF: Min-Var vs Med-IQR



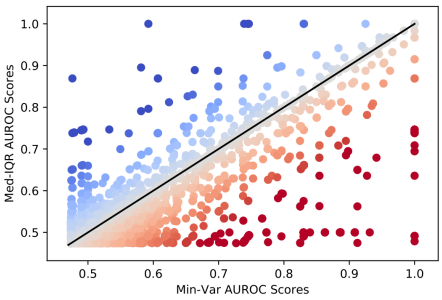
(e) COF: Min-Var vs Min-Max



(f) COF: Min-Var vs Med-IQR



(g) FastABOD: Min-Var vs Min-Max



(h) FastABOD: Min-Var vs Med-IQR

Figure 5.10: Scatter graphs plotting the AUC scores of outlier detection algorithms when performed using the minimal-variance polynomial whitening 'Min-Var' on the horizontal axis, and the AUC scores when using (a)-(d) 'Min-Max' or (e)-(h) 'Med-IQR' normalizations on the vertical axis. Points in red indicate a dataset where using Min-Var produced a better score than the alternative method, and points in blue indicate a dataset where using the alternative method produced a better score.

	Min-Var	Min-Max	Med-IQR
KNN	2195 (29%)	742 (10%)	632 (8%)
LOF	2338 (30%)	772 (10%)	604 (8%)
COF	2460 (32%)	811 (11%)	689 (9%)
FastABOD	1950 (25%)	705 (9%)	519 (7%)

Table 5.10: The number (and percentage) of datasets for which the given pre-processing method (in the column) produces AUC scores *strictly* better than the other methods, for each outlier detection method (in the row).

5.3.5 Principal component analysis

Principal component analysis (PCA) is a popular dimension-reduction technique, as it reduces a dataset to a chosen dimension p while retaining the greatest amount of variance (and therefore information) from the original dataset as possible. PCA finds p linear combinations of the variables of the dataset, giving p new compressed variables with maximal variance. As such, it is highly sensitive to the variances of the variables in the dataset. If one variable is measured on a much larger scale than the others, this variable will likely have much greater variance, and therefore be given much more weight in a linear combination than the other variables [126]. To prevent this, variables are often standardized to ensure they are all measured on the same scale.

Two methods of standardization are compared prior to performing PCA: Mahalanobis standardization, which is most commonly used before PCA, and minimal-variance standardization. Let the variable $v_i \in X$ have mean μ_i and standard deviation σ_i. The dataset transformed by Mahalanobis standardization is made up of the variables

$$z_i = \frac{(v_i - \mu_i)}{\sigma_i}$$

for $i \in \{1, \ldots, d\}$. To apply minimal-variance standardization, find the minimal-variance polynomial matrix A_k and use the values on the diagonal of A_k in place of σ_i (the choice of the parameter k will be discussed shortly):

$$w_i = \frac{(v_i - \mu_i)}{(A_k)_{i,i}}.$$

Note that this is different to minimal-variance whitening, in that only the diagonal of the minimal-variance polynomial matrix is used to perform the transformation. This is to align the minimal-variance method with the Mahalanobis standardization method.

First PCA example, data with d < N

In the first of the PCA examples, 1000 datasets are generated using the Python function `datasets.make_classification` from the Scikit-Learn package [185]. This function creates datasets with a chosen number of clusters, and a given number of 'informative' features, 'redundant' features (which are linear combinations of the informative features) and 'repeated' features (which are random duplicates of the informative features). The datasets used in this example have dimension d ranging from 10 to 26, with a random number of these being 'redundant' and 'repeated' features. Each dataset has N observations, with N between 100 to 300, and has between 2 and 5 clusters.

In these examples, parameter choices are made based on the relative size of the eigenvalues of the covariance matrix compared to the maximum eigenvalue. Let $\Lambda = \{\lambda_1, \ldots, \lambda_d\}$ be the set of eigenvalues of the covariance matrix of a dataset, let $\lambda_{\max}$ be the largest eigenvalue in Λ, and let $\bar{\lambda}$ be the mean of the eigenvalues in Λ. Let $p = p(\Lambda)$ be the desired number of principal components to be found (that is, the dimensionality of the dataset after applying PCA). The parameter $p(\Lambda)$ is chosen to be the number of eigenvalues Λ greater than the mean eigenvalue for that dataset, i.e:

$$p(\Lambda) = \sum_{i=1}^{d} \mathbb{1}_{\lambda_i > \bar{\lambda}},$$

as commonly used in practice [2].

The parameter $k = k(\Lambda)$ to be used in the minimal-variance polynomial is chosen based on the number of scaled eigenvalues $\pi_i = \lambda_i / \lambda_{\max}$ that are bigger than a given threshold t:

$$k(\Lambda) = \sum_{i=1}^{d} \mathbb{1}_{\pi_i > t}.$$

The thresholds $t \in \{0.1, 0.01, 0.001, 0.0001\}$ are used in Figure 5.11.

To evaluate the impact of the standardization methods on PCA, the total cumulative explained variance (CEV) of the transformed data is considered. That is, the amount of

variance from the original dataset that is still present after having implemented PCA. Figure 5.11 shows that using minimal-variance standardization gives much higher CEV than Mahalanobis whitening. Even for (relatively) low values of k (when $t = 0.1$) better results are achieved, and these continue to increase as t decreases (and therefore the value of k increases).

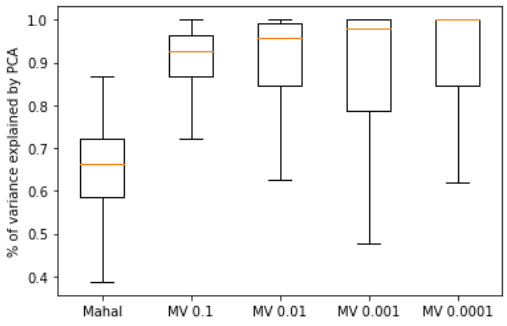


Figure 5.11: Boxplots of the CEV given by the PCA after different standardization methods for 1000 datasets. 'Mahal' denotes Mahalanobis standardization, and 'MV' denotes minimal-variance standardization. The number after MV indicates the threshold t used to choose the value of the parameter k.

PCA with K-means, data with d < N

In this example, 1000 new datasets are generated. For each of the 1000 datasets, 3 clusters are generated from multivariate Gaussian distributions $X^{(i)}$, $i = \{1, 2, 3\}$ with dimension $d = 100$, where the parameters $\mu^{(i)}$, $\Sigma^{(i)}$ and $N^{(i)}$ denote the mean, covariance matrix and number of observations in cluster $X^{(i)}$. The details of these parameters are given in Table 5.11. The eigenvalues of each $\Sigma^{(i)}$ taper off towards zero gradually. This creates a degenerate dataset with a rank that is hard to identify, a situation which the Moore-Penrose inverse struggles to deal with well.

The parameters p (the number of principal components) and k (the degree of the minimal-variance polynomial) will be set as they were in the previous example. In this section, the threshold used to set k is $t = 0.1$.

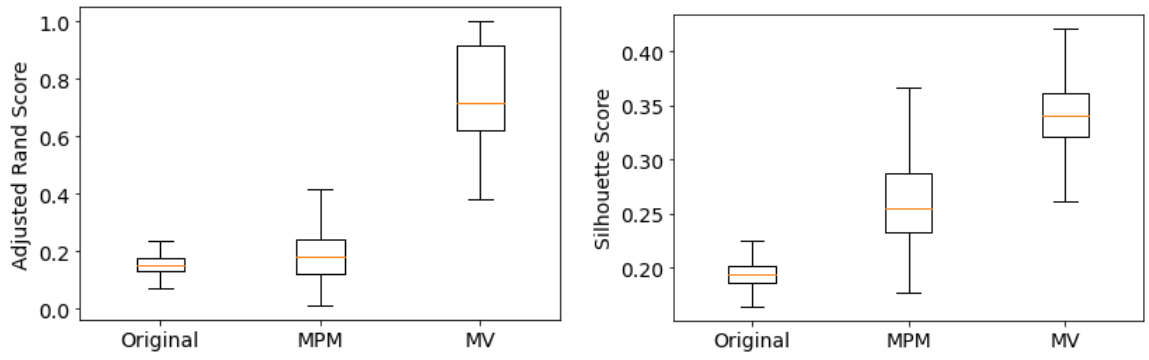
The K-means clustering algorithm, outlined previously in Section 3.5.2, aims to assign each point within a dataset to a cluster, by estimating the distances from each point to the estimated centre-point of a cluster of points. These examples compare the effect of dif-

i	$\mu^{(i)}$	Eigenvalues of $\Sigma^{(i)}$	$N^{(i)}$
1	$[0,\ldots,0]$	$[100,50,0.9^1,0.9^2,0.9^3,\ldots]$	166
2	$[1,\ldots,1]$	$[100,50,0.8^1,0.8^2,0.8^3,\ldots]$	166
3	$[0]*33+[1]*64$	$[100,50,0.8^1,0.8^2,0.8^3\ldots]$	168

Table 5.11: Details of clusters of datasets used for the $d < N$ PCA and K-means examples in Section 5.3.5. All datasets have dimension $d = 100$.

ferent standardization methods on the K-means clustering algorithm after applying PCA. The adjusted rand (AR) score [113, 220] is used to judge how well the algorithm has found the correct clusterings. An AR score of 0 indicates random labellings, and an AR score of 1 means the clusters were perfectly labelled by the algorithm. More information on the AR score is given in Appendix D.1.

The silhouette scores [203] are also used to compare the clusterings depending on the standardization methods. The silhouette score of a clustering indicates how well separated the clusters are. A score of 1 indicates well-distinguished clusters, whereas a score of -1 means that clusters have been incorrectly assigned. A higher silhouette score suggests that the standardization method and PCA have retained cluster structure well. More information on the silhouette score is available in Appendix D.3.



(a) AR Score

(b) Silhouette Score

Figure 5.12: (a) Adjusted rand (AR) scores and (b) silhouette scores of the labellings made by the K-means algorithm after PCA, which was applied to 1000 $d < N$ datasets with: no standardization (Original); Moore-Penrose Mahalanobis (MPM) standardization; minimal-variance (MV) standardization.

Figure 5.12 shows that using Mahalanobis standardization with the Moore-Penrose pseudoinverse (denoted MPM here) gives a slight improvement on using no standardization. However, using minimal-variance standardization before applying PCA and K-means clustering results in vastly better AR scores, as well as better silhouette scores, likely due to the tapering of the eigenvalues towards zero.

PCA with K-means, data with d > N

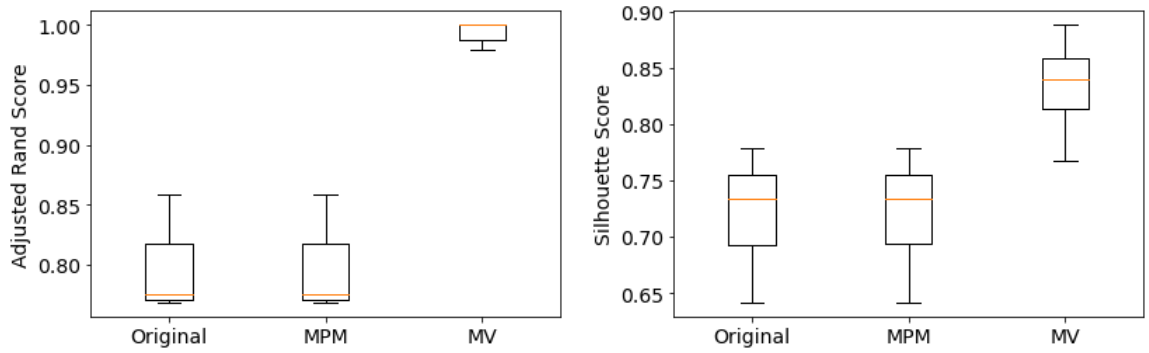
As in Section 5.3.5, 1000 different datasets are generated, each with dimension $d = 1000$ and number of observations $N = 430$. Each dataset is generated as a mixture of four multivariate Gaussian distributions $X_i \sim \mathcal{N}_d(\mu_i, \Sigma_i)$, $i = \{1,2,3,4\}$. The population parameters of each cluster are given in Table 5.12.

i	$\mu^{(i)}$	Eigenvalues of $\Sigma^{(i)}$	$N^{(i)}$
1	$[0,\dots,0]$	$[100,50,0.9^1,0.9^2,0.9^3,\dots]$	133
2	$[1,\dots,1]$	$[100,50,0.8^1,0.8^2,0.8^3,\dots]$	133
3	$[0]*333+[1]*667$	$[100,50,0.8^1,0.8^2,0.8^3\dots]$	134
4	$[1,\dots,1]$	$[100,50,0.1^1,0.1^2,0.1^3\dots]$	30

Table 5.12: Details of clusters of datasets used for the $d > N$ PCA and K-means examples in Section 5.3.5. The datasets have $d = 1000$ and $N = 430$.

Figure 5.13 shows boxplots of the AR scores and silhouette scores of the labels given by K-means clustering, after applying one of the standardization methods and PCA. It is clear that MPM standardization gives very similar results to the datasets with no standardization. The combination of MV standardization and PCA clearly performs better, as indicated by the boxplots of AR and silhouette scores in Figures 5.13.

The minimal-variance standardization method is clearly very useful in those cases where standardization would improve dimension reduction algorithms (or other multivariate data analysis methods), as it behaves similarly to the Moore-Penrose Mahalanobis standardization method, but does not struggle in cases where the datasets are degenerate, near-degeneracy or have unclear rank.



(a) AR Score

(b) Silhouette Score

Figure 5.13: (a) Adjusted rand (AR) scores and (b) silhouette scores of the labellings made by the K-means algorithm after PCA is applied to 1000 $d > N$ datasets with: no standardization (Original); Moore-Penrose Mahalanobis (MPM) standardization; minimal-variance (MV) standardization.

5.4 Iterative minimal-variance whitening

In this section, an extension to the minimal-variance whitening method is presented. As demonstrated in [101] and Section 4.2.4, producing a polynomial with a high degree can lead to numerical instability, particularly when applied to large dimensional matrices. This causes the breakdown seen in previous examples, like those given in Section 5.3.1 as k increases. Iterative procedures are known to be more robust and less vulnerable to numerical errors when used with high dimensional matrices [94]. To take advantage of this, lower degree polynomials can be applied iteratively to find an approximation to the inverse square root of the covariance matrix. This method is known as the 'iterative minimal-variance whitening' method. Section 5.4.1 will outline how the iterative method is applied, and Section 5.4.2 will give some examples of the method being used.

5.4.1 Constructing iterative minimal-variance whitening

If $X^{(0)}$ is the original dataset to be whitened, let $X^{(1)}$ be the dataset output by the minimal-variance whitening method. Perform minimal-variance whitening on the dataset $X^{(1)}$ to produce the dataset $X^{(2)}$, and repeat until the optimal or required results are found. Using this iterative procedure allows for a lower (and perhaps changing) value of k to be used, making the method less computationally-intensive and more stable.

The iterative algorithm converges to give a whitened dataset with identity covariance matrix, if the starting dataset is full rank. If the starting dataset is not full rank, the covariance matrix of the final whitened dataset aims to be approximately equal to a multiple of the identity matrix, with $d-r$ of the diagonal entries set to zero. Therefore, the eigenvalues of the covariance matrix of the transformed dataset will be (approximately) distributed at either one or two points: either all ones (in the nonsingular case) or a scalar value and zeros (in the singular case). Theorem 5 produces a moment condition to detect if the eigenvalues of a matrix are distributed at exactly one or two points.

Theorem 5. *Let Σ be a $d \times d$ matrix, and let $S_i = trace(\Sigma^i)$. Define the following moment condition:*

$$\Psi(\Sigma) = S_1 S_3 - S_2^2. \tag{5.11}$$

If all eigenvalues of Σ are distributed at either one or two points exactly, $\Psi(\Sigma) = 0$.

Proof. Define $\Lambda = \{0, \lambda, 1\}$ to be the support of the set of eigenvalues of Σ, with relative probabilities $P = \{1 - p_1 - p_2, p_1, p_2\}$, $p_1, p_2 > 0$. Define the non-central moments $\mu_i = \sum_{j=1}^{3} P_i \Lambda_i^j$, specifically:

$$\mu_0 = 1, \quad \mu_1 = p_1\lambda + p_2, \quad \mu_2 = p_1\lambda^2 + p_2, \quad \mu_3 = p_1\lambda^3 + p_2, \quad \mu_4 = p_1\lambda^4 + p_2.$$

If the values of the first three non-central moments are found empirically to be $\mu_1 = m_1$, $\mu_2 = m_2$, $\mu_3 = m_3$, the simultaneous equations

$$\begin{aligned} p_1\lambda + p_2 &= m_1 \\ p_1\lambda^2 + p_2 &= m_2 \\ p_2\lambda^3 + p_2 &= m_3 \end{aligned}$$

can be solved to find the variables p_1, p_2 and λ in terms of m_1, m_2 and m_3:

$$\begin{aligned} p_1 &= \frac{(m_1 - m_2)^3}{(m_2 - m_3)(-2m_2 + m_3 + m_1)} \\ p_2 &= \frac{m_1 m_3 - m_2^2}{-2m_2 + m_3 + m_1} \\ p_3 &= \frac{m_2 - m_3}{m_1 - m_2}. \end{aligned}$$

Define the moment matrix

$$\mathcal{M} = \begin{pmatrix} 1 & \mu_1 & \mu_2 \\ \mu_1 & \mu_2 & \mu_3 \\ \mu_2 & \mu_3 & \mu_4 \end{pmatrix} = \begin{pmatrix} 1 & p_1\lambda + p_2 & p_1\lambda^2 + p_2 \\ p_1\lambda + p_2 & p_1\lambda^2 + p_2 & p_1\lambda^3 + p_2 \\ p_1\lambda^2 + p_2 & p_1\lambda^3 + p_2 & p_1\lambda^4 + p_2 \end{pmatrix}.$$

The determinant of $\mathcal{M}$ can be found to be:

$$\begin{aligned} det(\mathcal{M}) &= p_1 p_2 \lambda^2 \left(-p_1\lambda^2 + 2p_1\lambda - p_1 - p_2\lambda^2 + 2p_2\lambda - p_2 + \lambda^2 - 2\lambda + 1\right) \\ &= -\frac{\left(m_1^2 - m_1 m_2 - m_1 m_3 + m_2^2 - m_2 + m_3\right)\left(m_1 m_3 - m_2^2\right)}{m_1 - m_2}. \end{aligned} \tag{5.12}$$

If $det(\mathcal{M}) = 0$, the distribution of eigenvalues has at most two points of support [158]. Re-write the first bracket on the numerator of (5.12) in terms of p_1, p_2 and λ:

$$m_1^2 - m_1 m_2 - m_1 m_3 + m_2^2 - m_2 + m_3 = -p_1\lambda^2(\lambda - 1)(p_1 + p_2 - 1). \tag{5.13}$$

If the distribution is supported at three points, the value of Equation (5.13) is nonzero. However, if the distribution is supported at exactly two points, Equation (5.13) will equal zero.

Consider the second bracket on the numerator of (5.12) in terms of p_1, p_2 and λ:

$$m_1 m_3 - m_2^2 = p_1 p_2 \lambda(\lambda - 1)^2. \tag{5.14}$$

If Equation (5.14) equals zero, then $\lambda = 0$ or $\lambda = 1$, so the support of the eigenvalues of Σ can only consist of two points. Note that the values 0 and 1 in the support X can be changed to any values, so this theorem generalizes to any two eigenvalues.

□

Using the results of Theorem 5, when the data has been fully whitened, $\Psi(\Sigma) = 0$. If the data whitening needs only to be approximate, a tolerance t can be set, and iterations can be halted when $\Psi(\Sigma) \leq t$. Otherwise, iterations are performed until $\Psi(\Sigma) = 0$ or $\Psi(\Sigma)$ converges. The iterative minimal-variance whitening procedure is outlined in Algorithm 3.

There are many alternative methods that could be used to terminate the iterations. Firstly, different conditions can be used in place of the moment condition in (5.11). If decorrelation is more important than whitening, for example, the sum of squares of the off-diagonal

Algorithm 3: Iterative minimal-variance whitening algorithm

Input: $X^{(0)}$: data; k: degree of polynomials; t: tolerance of moment condition;
μ: mean of X; Σ: covariance matrix of X

Output: X_{A_k}: transformed data

Set $i = 0$, $m = \infty$;

while $m > t$ **do**

$X^{(i+1)} = MV(X^{(i)}, k)$; Apply MV whitening, Algorithm 2

$\mu^{(i)} = \frac{1}{N}\sum_{j=0}^{d} x_j^{(i)}$;

$\Sigma^{(i)} = \frac{1}{N}(X^{(i)} - \mu^{(i)})(X^{(i)} - \mu^{(i)})^\top$;

$m = \Psi(\Sigma^{(i)})$; Moment condition, Equation (5.11)

$i = i + 1$

end

$X_{A_k} = X^{(i-1)}$

entries of the covariance matrix could be used instead. The Wasserstein metric between the whitened dataset X_{A_k} and the standard normal distribution could also be used, as defined in Equation (5.9). However, the Wasserstein metric is only equal to zero if the two distributions being compared are equal, so if X_{A_k} is not full rank this metric will never reach zero. Section 4.9.2 of [101] describes several methods of termination criteria for iterative procedures, including introducing a threshold on the relative difference of the norms between two successive iterates.

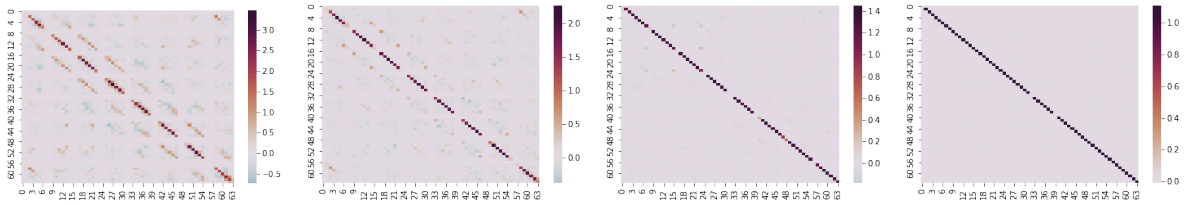
It may be of interest to recover the matrix A_k which transforms the original data $X^{(0)}$ to the final whitened data X_{A_k}. Let $A_k^{(i)}$ be the minimal-variance whitening matrix on the ith iteration of Algorithm 3, and let $\mathcal{I}$ be the total number of iterations performed. Then find the matrix $A_k = \Pi_{i=0}^{\mathcal{I}} A_k^{(\mathcal{I}-i)}$. That is, $A_k = A_k^{(\mathcal{I})} A_k^{(\mathcal{I}-1)} \dots A_k^{(1)} A_k^{(0)}$ and $X_{A_k} = A_k^{(\mathcal{I})} A_k^{(\mathcal{I}-1)} \dots A_k^{(1)} A_k^{(0)} X^{(0)}$.

5.4.2 Whitening data using iterative minimal-variance polynomials

The performance of the iterative minimal-variance whitening method will be evaluated here by applying Algorithm 3 to real datasets. This section begins by considering data with $d < N$, and will later give examples using examples with $d \geq N$.

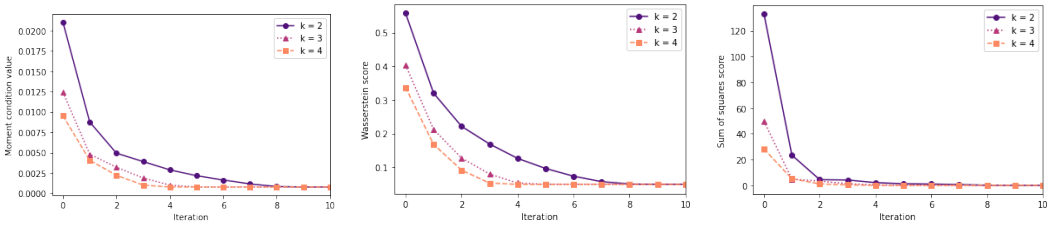
Data with d < N

The real datasets used in Section 5.3.1 are considered again here (see Table 5.2 for details of the datasets). Looking first at the Digits dataset, iterative whitening is applied using $k = 2$ in each iteration. Figure 5.14 shows that applying minimal-variance whitening with $k = 2$ once does not result in a whitened covariance matrix. However, after 5 iterations the dataset is approximately whitened, and after 10 iterations it is exactly whitened (that is, the covariance matrix is equal to the identity matrix with the exception of $d - r$ entries on the diagonal). This is an improvement on the minimal-variance whitening when performed without iterations with a higher value of k (see Figure 5.6e).



(a) Iteration 1 (b) Iteration 2 (c) Iteration 5 (d) Iteration 10

Figure 5.14: Heatmaps of the covariance matrix of the Digits dataset when using iterative whitening with $k = 2$.

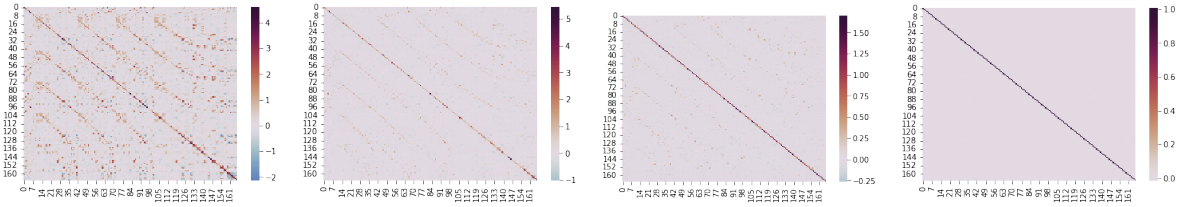


(a) Moment condition (b) Wasserstein score (c) SSOD score

Figure 5.15: The (a) moment condition values (b) Wasserstein scores (c) SSOD score of the Digits dataset for each iteration of iterative whitening, using $k = 2$ (solid line with circles), $k = 3$ (dotted line with triangles) and $k = 4$ (dashed line with squares).

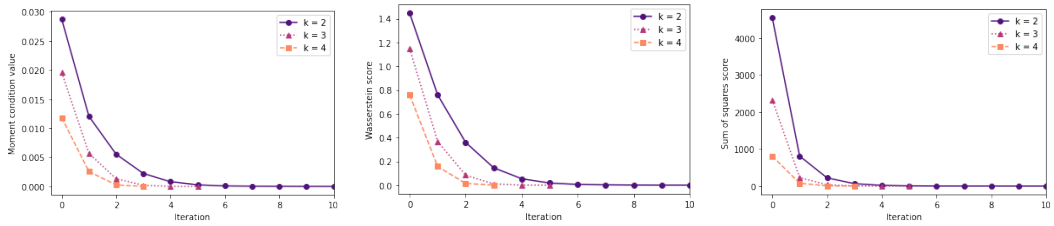
Figure 5.15 shows the progression of three metrics during the whitening iterations: the moment condition (Equation (5.11)), the Wasserstein score (Equation (5.9)) and the sum of squares of the off-diagonals of the covariance matrix (SSOD score). These metrics are considered for minimal-variance iterative whitening with $k = 2$, $k = 3$ and $k = 4$. As expected, higher values of k converge to a whitened dataset faster than lower values.

Recall that using the parameter k results in a $(k-1)$-degree polynomial. Using $k = 2$ requires 10 iterations to converge for the Digits example, which is equivalent to producing a polynomial of degree 10. Using $k = 3$ requires 7 iterations, equivalent to a polynomial of degree 14, and using $k = 4$ requires 5 iterations, equivalent to a polynomial of degree 15. Thus, despite requiring less iterations, using higher values of k can result in higher computational effort and more complex polynomials.



(a) Iteration 1 (b) Iteration 2 (c) Iteration 4 (d) Iteration 10

Figure 5.16: Heatmaps of the covariance matrix of the Musk dataset when using iterative whitening with $k = 2$.



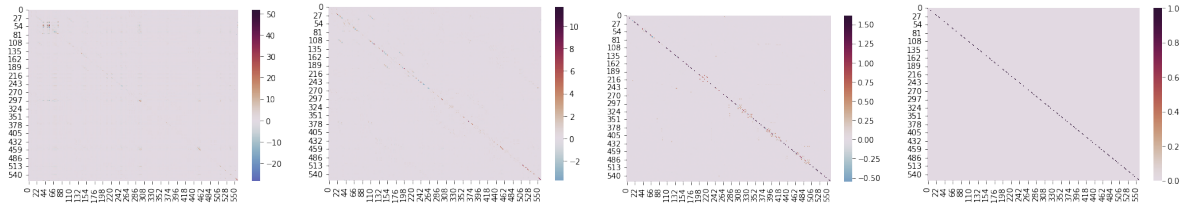
(a) Moment condition (b) Wasserstein score (c) SSOD score

Figure 5.17: The (a) moment condition values (b) Wasserstein scores (c) SSOD score of the Musk dataset for each iteration of iterative whitening, using $k = 2$ (solid line with circles), $k = 3$ (dotted line with triangles) and $k = 4$ (dashed line with squares).

A similar gradual convergence to a whitened dataset is seen with the Musk dataset in Figure 5.16 with $k = 2$, and with higher values of k in Figure 5.17. For $k = 2$, 10 iterations are used, equivalent to a 10-degree polynomial. For $k = 3$, 6 iterations are used, equivalent to a 12-degree polynomial. For $k = 4$, 4 iterations are used, again equivalent to a 12-degree polynomial.

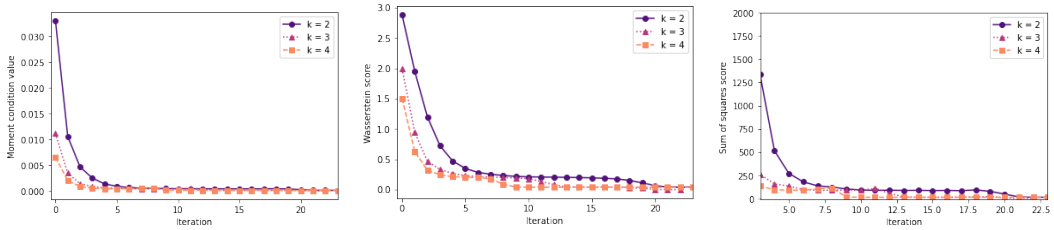
The HAR dataset requires more iterations than the datasets previously considered, as can be seen by Figure 5.18. Figure 5.19 shows that when $k = 2$, the dataset is whitened after 22 iterations (equivalent to a polynomial of degree 22). When $k = 3$, 14 iterations are

required (equivalent to a 28 degree polynomial) and when $k = 4$, 10 iterations are required (equivalent to a 30 degree polynomial). If this dataset was not singular, the 'true' square root inverse covariance matrix of this dataset would be a $d - 1 = 560$ degree polynomial, so the minimal-variance whitening polynomials are comparatively of very low degree.



(a) Iteration 1 (b) Iteration 3 (c) Iteration 10 (d) Iteration 22

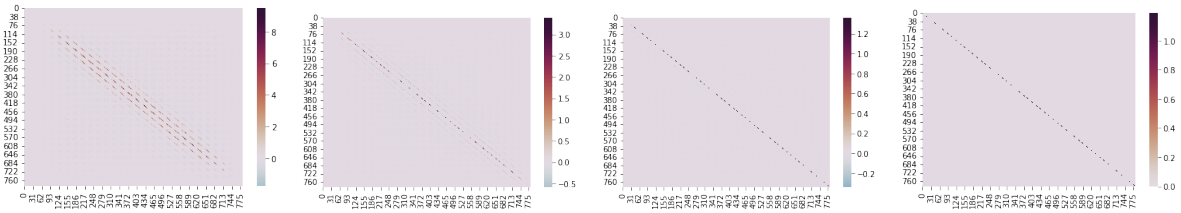
Figure 5.18: Heatmaps of the covariance matrix of the HAR dataset when using iterative whitening with $k = 2$.



(a) Moment condition (b) Wasserstein score (c) SSOD score

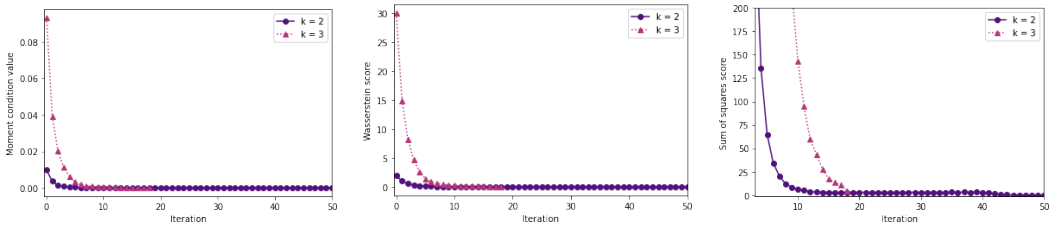
Figure 5.19: The (a) moment condition values (b) Wasserstein scores (c) SSOD score of the HAR dataset for each iteration of iterative whitening, using $k = 2$ (solid line with circles), $k = 3$ (dotted line with triangles) and $k = 4$ (dashed line with squares).

MNIST is a 784-dimensional dataset, with iterative whitening applied using $k = 2$ in Figure 5.20. For this example, only $k = 2$ and $k = 3$ are considered in Figure 5.21 as using $k = 4$ is too computationally intensive to use iteratively. For $k = 2$, 47 iterations are required for perfect whitening, equivalent to a 47-degree polynomial. For $k = 3$, the algorithm does not converge to a perfectly whitened dataset. However, it does get close to a perfectly whitened dataset within 20 iterations. Similarly, depending on the need for perfectly whitened data, the iterative procedure with $k = 2$ could be terminated earlier than 47 iterations, and a nearly-whitened dataset could be used, if this is satisfactory.



(a) Iteration 1 (b) Iteration 3 (c) Iteration 10 (d) Iteration 47

Figure 5.20: Heatmaps of the covariance matrix of the MNIST dataset when using iterative whitening with $k = 2$.



(a) Moment condition (b) Wasserstein score (c) SSOD score

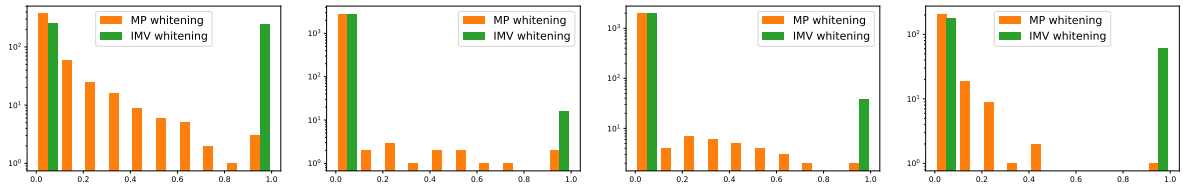
Figure 5.21: The (a) moment condition values (b) Wasserstein scores (c) SSOD score of the MNIST dataset for each iteration of iterative whitening, using $k = 2$ (solid line with circles), $k = 3$ (dotted line with triangles).

Different values of k can be used interchangeably throughout minimal-variance iterative whitening, but experiments have shown that using $k = 2$ is as effective as using higher values of k, and takes away the need to choose parameter values.

Data with d > N

The real datasets from Section 5.3.1 with $d > N$ (detailed in Table 5.5) are now used to show how iterative whitening works with high dimension low sample size data. Figure 5.7 showed that the non-iterative minimal-variance whitening method provides an advancement over the Moore-Penrose Mahalanobis whitening method, but does not always result in perfect whitening.

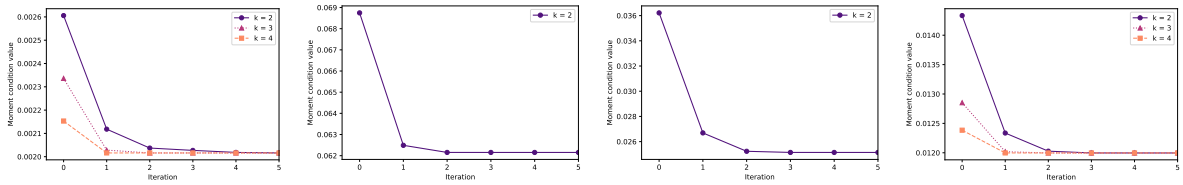
For each of the datasets considered, iterative minimal-variance whitening with parameter $k = 2$ perfectly whitens the data. Figure 5.22 demonstrates this by displaying the eigenvalues of the covariance matrices of the whitened datasets. Iterative minimal-variance whitening produces eigenvalues only equal to zero or one, exactly.



(a) Madelon† (b) Yeast (c) Colon (d) DB-emails

Figure 5.22: Log-scale histograms, showing the eigenvalues of the covariance matrix after the data has been whitened by Moore-Penrose (MP) whitening (orange histogram) and iterative minimal-variance (IMV) whitening with $k = 2$ (green histogram), for each of the real datasets considered in Table 5.5.

Furthermore, the correct rank of the datasets is retained using iterative minimal-variance whitening. The madelon† dataset has rank $r = 249$, the yeast dataset has rank $r = 16$, the colon dataset has rank $r = 39$ and the DB-emails dataset has rank $r = 62$.



(a) Madelon† (b) Yeast (c) Colon (d) DB-emails

Figure 5.23: The moment condition values (5.11) for each iteration of iterative whitening, using (where possible) $k = 2$ (solid line with circles), $k = 3$ (dotted line with triangles) and $k = 4$ (dashed line with squares), for each of the four datasets given in Table 5.5.

Figure 5.22 shows the moment condition values for each iteration of minimal-variance whitening. The yeast and colon datasets only show moment values for $k = 2$ as higher values of k caused instability in these datasets. The madelon† and DB-emails datasets show that using $k = 2$ does not require many more iterations than using higher values of k, often making it less computationally expensive than using the higher values of k. This, combined with the lower risk of instability and the good results produced, makes using $k = 2$ in iterative minimal-variance whitening an attractive parameter choice.

Overall, it is clear that iterative minimal-variance whitening is an effective way to whiten both $d < N$ and $d \geq N$ datasets, even when other methods like the Moore-Penrose Mahalanobis method cannot.

5.5 Matrix rank estimation

For rank-deficient matrices, it can often be hard to identify the true rank of the matrix due to the presence of gradually decreasing eigenvalues. In this section, a method using the constraint adjustment value (introduced in Section 5.2.3) will be used to detect the rank of a matrix. This will then be followed by a rank-detection method that combines the constraint adjustment value and the iterative minimal-variance whitening method.

5.5.1 Fuzzy minimal-variance rank estimation

Consider the minimal-variance whitening polynomial with $\alpha = 1/2$. As discussed in Section 5.2.3, the constraint would ideally be

$$\operatorname{trace}\left(A_k \Sigma^{1/2}\right) = r, \tag{5.15}$$

where r is the rank of Σ, so that A_k behaves similarly to $\Sigma^{-1/2}$ or the square root of the Moore-Penrose pseudoinverse. However, Section 5.2.3 shows that using $\alpha = 1/2$ actually enforces the constraint $\operatorname{trace}\left(A_k \Sigma^{1/2}\right) = d$. Section 5.2.3 discusses a method to alleviate this issue, by finding a value c_k^* (Equation 5.8) that adjusts A_k such that Equation (5.15) holds.

If $\operatorname{trace}\left(A_k \Sigma^{1/2}\right)$ is exactly equal to d, then $c_k^* = r/d$. However, the rank r of Σ may not always be known and can be hard to find; increasingly small eigenvalues can make distinguishing between non-zero and zero eigenvalues a difficult task. It is possible to use the constraint adjustment method to estimate the rank r, however, using $\tilde{r}_k = c_k^* \times d$, where c_k^* is the adjustment value given by Equation (5.8) when using degree parameter k. Let $\tilde{r}_k$ be called the fuzzy minimal-variance rank estimate (or fuzzy-MV rank).

Two examples of fuzzy-MV rank are given, with covariance matrices defined as Σ_1 and Σ_2. Let Λ_1 and Λ_2 be the eigenvalues of the two covariance matrices, respectively. Let $\Lambda_i^{(r)}$ be the first r eigenvalues of covariance matrix Σ_i. The eigenvalues are generated as follows:
$\Lambda_1 =$ `[5, 4, 3, 2, 1] + [0.01 ** i for i in range(11, 21)] + [0] * 5`,
$\Lambda_2 =$ `[5, 4, 3, 2, 1] + [0.01 ** i for i in range(11, 21)] + [0] * 85`,
using Python notation. Clearly, it would be expected for these two datasets to have the same rank, given they have equal eigenvalues except for some additional zeros in Λ_2. In Table 5.13, the rank r of each covariance matrix Σ_i is calculated using the function

`numpy.linalg.matrix_rank` in Python. This function returns the matrix rank of an array by identifying singular values less than a given threshold t. The default threshold (which is used here) is calculated using the SVD method. For more information on this method, see the `numpy` package documentation [98] or see [191] for a more thorough explanation. Table 5.13 shows that this method identifies different ranks for the two datasets, where equal ranks would be expected.

	Λ_i	r	$\Lambda_i^{(r)}$
Σ_1	[5, 4, 3, 2, 1, 1e-11, 1e-12, 1e-13, 1e-14, 1e-15, 1e-16, 1e-17, 1e-18, 1e-19, 1e-20] + [0]*5	8	[5, 4, 3, 2, 1, 1e-11, 1e-12, 1e-13]
Σ_2	[5, 4, 3, 2, 1, 1e-11, 1e-12, 1e-13, 1e-14, 1e-15, 1e-16, 1e-17, 1e-18, 1e-19, 1e-20] + [0]*85	7	[5, 4, 3, 2, 1, 1e-11, 1e-12]

Table 5.13: The eigenvalues Λ_i and ranks r of the two matrices Σ_1 and Σ_2, using the SVD method to compute ranks. $\Lambda_i^{(r)}$ denotes the r eigenvalues detected as nonzero by the rank identified.

	$\tilde{r}_2$	$\tilde{r}_3$	$\tilde{r}_4$	$\tilde{r}_5$	$\tilde{r}_6$	$\tilde{r}_7$
Σ_1	4.959	4.995	5.000	5.000	5.000	5.000
Σ_2	4.959	4.995	5.000	5.000	5.000	5.000

Table 5.14: The ranks of two matrices Σ_1 and Σ_2 using the fuzzy-MV rank method for different values of k.

Table 5.14 shows the fuzzy minimal-variance rank estimations for Σ_1 and Σ_2, for different values of k in the minimal-variance estimation. The rank values $\tilde{r}_k$ tend towards a value as k increases, and estimate that Σ_1 and Σ_2 have the same rank, as desired.

The rank estimator is named 'fuzzy' as the rank $\tilde{r}_k$ is not necessarily an integer. Consider a matrix Σ_3 with eigenvalues Λ_3 = `[5, 4, 3, 2, 1] + [0.01**i for i in range(1, 26)]`. Whereas the SVD method used in Table 5.13 can be affected by choice of threshold, the fuzzy-MV rank estimation method has no tolerance to be set. The only

variables to be considered are the degree parameter k and the weighting function in Equation (5.8). Much like the results shown in Table 5.3 and Table 5.4, Table 5.15 shows that as k increases there is a tendency towards a given rank value (shown in bold), until numerical instability starts to cause a decrease. The fuzzy-MV rank estimation is taken to be the maximum value of all $\tilde{r}_k$ values, before the values begin to decrease. This also provides a strong indication for the maximum value of k to be used in the minimal-variance polynomial before instability occurs.

$w(\lambda)$	$\tilde{r}_2$	$\tilde{r}_3$	$\tilde{r}_4$	$\tilde{r}_5$	$\tilde{r}_6$	$\tilde{r}_7$	$\tilde{r}_8$	$\tilde{r}_9$
λ	5.168	5.288	5.423	5.807	**6.227**	6.225	6.224	6.223
λ^2	5.083	5.240	5.341	5.620	**6.226**	6.225	6.224	6.224
λ^3	5.168	5.267	5.375	5.682	**6.227**	6.225	6.224	6.224

Table 5.15: The ranks of the matrix Σ_3 using the fuzzy-MV rank method for different values of k, and using different weight functions $w(\lambda)$ in calculating the adjustment value c_k^*. The bold values indicates the fuzzy-MV rank estimation for that weight function.

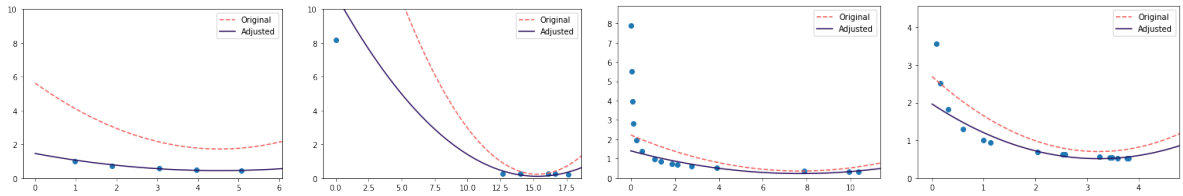
Regarding the weight function in Equation (5.8), using $w(\lambda) = \lambda^j$ for any value $j \geq 1$ seems to have little difference on the fuzzy-MV rank estimation, as illustrated in Table 5.15. The weight function $w(\lambda) = \lambda$ is recommended to be used by default.

5.5.2 Iterative fuzzy minimal-variance rank estimation

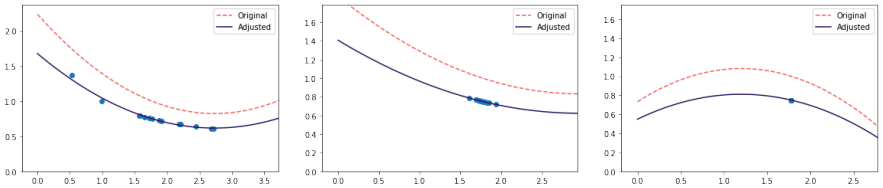
As seen in Section 5.3.1, the minimal-variance whitening polynomial does not always perfectly whiten a dataset. However, using the minimal-variance polynomial iteratively, as described in Section 5.4, can produce more successful results. To the same end, the iterative minimal-variance method can be used to provide a better estimation of the fuzzy-MV rank.

Recall that $X^{(i)}$ denotes the dataset during the ith iteration of the iterative minimal-variance algorithm (Algorithm 3), and that $\Sigma^{(i)}$ is the covariance matrix of $X^{(i)}$. Then for each iteration i, a new polynomial A_k is found to fit the inverse square-root eigenvalues of $\Sigma^{(i)}$. If $\Sigma^{(i)}$ has any zero-valued eigenvalues, this polynomial A_k will need to use the constraint adjustment (5.8). As an example, consider a dataset X, generated from a zero-mean multivariate Gaussian distribution using a diagonal covariance matrix Σ with entries

`[5, 4, 3, 2, 1] + [0.5 ** i for i in range(11, 21)] + [0] * 5`, again in Python notation. Figure 5.24a shows iteration 1 of the iterative minimal-variance whitening method. Using the notation defined in Section 5.4, the (largest) eigenvalues of the empirical covariance matrix $\Sigma^{(0)}$ are plotted in blue dots against the inverse square root eigenvalues. The dashed orange line shows the minimal-variance polynomial with $k = 3$ (that is, a polynomial of degree 2) for $\Sigma^{(0)}$, with the adjusted polynomial plotted by the solid purple line. Figure 5.24b depicts iteration 2, showing the eigenvalues of $\Sigma^{(1)}$ and the associated minimal-variance polynomials, and so forth for the rest of the subfigures in Figure 5.24.



(a) Iteration 1 (b) Iteration 2 (c) Iteration 3 (d) Iteration 4



(e) Iteration 5 (f) Iteration 6 (g) Iteration 7

Figure 5.24: Behaviour of the minimal-variance polynomials when used in iterative whitening. Blue dots: eigenvalues against inverse square root eigenvalues. Orange dashed line: minimal-variance polynomial using $k = 3$ for the given iteration. Purple solid line: adjusted polynomial using methods described in Section 5.2.3.

The notation for the constraint adjustment value c_k^* is adjusted to apply to iterative whitening. Let $c_{k,i}^*$ be the constraint adjustment value for iteration i. Let $\tilde{\Lambda}_i$ be the nonzero eigenvalues of $\Sigma^{(i)}$. Then the adjustment value is:

$$c_{k,i}^* = \frac{\sum_{\lambda \in \tilde{\Lambda}_i} w(\lambda) \lambda^{-0.5} p_k(\lambda)}{\sum_{\lambda \in \tilde{\Lambda}_i} w(\lambda) p_k(\lambda)^2}.$$

For each iteration i, the iterative fuzzy-MV rank estimation is found to be $\tilde{r}_{k,i} = c_{k,i}^* \times d$. Table 5.16 gives the values of $\tilde{r}_{k,i}$ for $k \in \{2, \ldots, 6\}$, $i = \{1, \ldots, 12\}$. For all values of k considered, the rank estimation converges towards the same value.

i \ k	2	3	4	5	6
1	5.259	9.854	10.384	10.501	14.467
2	5.596	12.768	13.776	14.188	15.000
3	6.401	14.628	14.964	14.998	15.000
4	8.053	14.995	15.000	15.000	15.000
5	10.291	15.000	15.000	15.000	15.000
6	12.089	15.000	15.000	15.000	15.000
7	13.579	15.000	15.000	15.000	15.000
8	14.554	15.000	15.000	15.000	15.000
9	14.932	15.000	15.000	15.000	15.000
10	14.998	15.000	15.000	15.000	15.000
11	15.000	15.000	15.000	15.000	15.000
12	15.000	15.000	15.000	15.000	15.000

Table 5.16: Ranks $\tilde{r}_{k,i}$ for a given matrix, given by iterative fuzzy-MV rank estimation, for polynomial degree parameter k and iteration i.

Further work on the iterative fuzzy-MV rank estimation is needed to assess the stability of the method, as well as more empirical investigations on different data distributions. However, the method clearly performs well and in a stable manner for Gaussian data with an unclear rank.

5.6 Alternative minimal-variance polynomial methods

Thus far, only polynomials in the form of Equation (5.1) have been considered, that is:

$$A_k = \sum_{i=0}^{k-1} \theta_i \Sigma^i = \theta_0 I + \theta_1 \Sigma + \theta_2 \Sigma^2 + \dots. \tag{5.16}$$

However, polynomials of different forms can also be considered alongside this. Given that the aim is to approximate (or replace) $\Sigma^{-1/2}$, it perhaps makes sense to make use of the square root of the covariance matrix by creating a polynomial of the form:

$$B_k = \sum_{i=0}^{k-1} \theta_i \Sigma^{i/2} = \theta_0 I + \theta_1 \Sigma^{1/2} + \theta_2 \Sigma + \dots. \tag{5.17}$$

Furthermore, the use of the identity matrix in Equations (5.16) and (5.17) can be seen as a regularization on the minimal-variance polynomial. Although regularization has many benefits (particularly when working with close-to-degenerate data), it could sometimes be useful to consider a non-regularized polynomial:

$$C_k = \sum_{i=1}^{k-1} \theta_i \Sigma^i = \theta_1 \Sigma + \theta_2 \Sigma^2 + \dots.$$

For completeness, consider the non-regularized polynomial using the square root of the covariance matrix:

$$D_k = \sum_{i=1}^{k-1} \theta_i \Sigma^{i/2} = \theta_1 \Sigma^{1/2} + \theta_2 \Sigma + \dots$$

Forming the minimal-variance polynomial matrix using these alternative polynomials is very similar to the method outlined in Theorem 4, with some slight alterations. Table 5.17 gives the formulae for the different coefficient vectors for each of the alternative polynomial methods. Note that $\alpha = 1/2$ is used by default in the definition of the constraint (5.2) as experiments show this parameter value gives the most desirable results. See Appendix B.2 for the full details on calculating the coefficient vectors in Table 5.17.

Section 5.2.1 defines the notation $S_j = \text{trace}(\Sigma^j)$ and $S_{(i,k)} = (S_i, S_{i+1}, \dots, S_{i+k-1})$. It is necessary to slightly alter the latter definition to include a new parameter γ:

$$S_{(i,k,\gamma)} = \left(S_i, S_{i+\gamma}, S_{i+2\gamma}, \dots, S_{i+k-\gamma}\right). \tag{5.18}$$

As explored in Section 5.2.3, the polynomials require a small adjustment if the rank of the dataset is less than d. This adjustment value is the same as c_k^* from Equation (5.8), but replaces the polynomial $p_k(\lambda)$ with the corresponding polynomial from Table 5.17.

One key difference between the different polynomials in Table 5.17 is the rank of the minimal-variance polynomial matrix. The rank of the whitening matrices A_k and B_k will always be equal to d, the dimension of the dataset, thanks to the presence of the identity matrix I in the polynomial. On the other hand, C_k and D_k will (likely) have equal rank to the covariance matrix Σ.

As shown in Table 5.18, polynomials A_k and B_k have k terms, whereas polynomials C_k and D_k have $k-1$ terms. Therefore, for a fair comparison in the examples that follow, the polynomials A_k and B_k use degree parameter k, and the polynomials C_k and D_k use degree parameter $k+1$.

Polynomial	Coefficient vector θ	Matrix $M_{(k)}$
$A_k = \sum_{i=0}^{k-1} \theta_i \Sigma^i$	$\theta_A = \frac{S_0}{S^\top_{\left(\frac{1}{2},k,1\right)} M^{-1}_{(k)} S_{\left(\frac{1}{2},k,1\right)}} M^{-1}_{(k)} S_{\left(\frac{1}{2},k,1\right)}$	$M_{(k)} = \begin{pmatrix} S_1 & S_2 & \cdots & S_k \\ S_2 & S_3 & \cdots & S_{k+1} \\ \vdots & \vdots & \ddots & \vdots \\ S_k & S_{k+1} & \cdots & S_{2k-1} \end{pmatrix}$
$B_k = \sum_{i=0}^{k-1} \theta_i \Sigma^{i/2}$	$\theta_B = \frac{S_0}{S^\top_{\left(\frac{1}{2},\frac{k}{2},\frac{1}{2}\right)} M^{-1}_{(k)} S_{\left(\frac{1}{2},\frac{k}{2},\frac{1}{2}\right)}} M^{-1}_{(k)} S_{\left(\frac{1}{2},\frac{k}{2},\frac{1}{2}\right)}$	$M_{(k)} = \begin{pmatrix} S_1 & S_{3/2} & \cdots & S_{(k+1)/2} \\ S_{3/2} & S_2 & \cdots & S_{(k+2)/2} \\ \vdots & \vdots & \ddots & \vdots \\ S_{(k+1)/2} & S_{(k+2)/2} & \cdots & S_k \end{pmatrix}$
$C_k = \sum_{i=1}^{k-1} \theta_i \Sigma^i$	$\theta_C = \frac{S_0}{S^\top_{\left(\frac{3}{2},k-1,1\right)} M^{-1}_{(k)} S_{\left(\frac{3}{2},k-1,1\right)}} M^{-1}_{(k)} S_{\left(\frac{3}{2},k-1,1\right)}$	$M_{(k)} = \begin{pmatrix} S_3 & S_4 & \cdots & S_{k+1} \\ S_4 & S_5 & \cdots & S_{k+2} \\ \vdots & \vdots & \ddots & \vdots \\ S_{k+1} & S_{k+2} & \cdots & S_{2k-1} \end{pmatrix}$
$D_k = \sum_{i=1}^{k-1} \theta_i \Sigma^{i/2}$	$\theta_D = \frac{S_0}{S^\top_{\left(1,\frac{k-1}{2},\frac{1}{2}\right)} M^{-1}_{(k)} S_{\left(1,\frac{k-1}{2},\frac{1}{2}\right)}} M^{-1}_{(k)} S_{\left(1,\frac{k-1}{2},\frac{1}{2}\right)}$	$M_{(k)} = \begin{pmatrix} S_2 & S_{5/2} & \cdots & S_{(k+2)/2} \\ S_{5/2} & S_3 & \cdots & S_{(k+3)/2} \\ \vdots & \vdots & \ddots & \vdots \\ S_{(k+2)/2} & S_{(k+3)/2} & \cdots & S_k \end{pmatrix}$

Table 5.17: Details to compute the four different minimal-variance polynomial matrices discussed in Section 5.6. Note the altered definition of $S_{(i,k,\gamma)}$ given by Equation (5.18). Derivations are detailed in Appendix B.2.

Polynomial	Number of terms	Degree of polynomial
A_k	k	$k-1$
B_k	k	$(k-1)/2$
C_k	$k-1$	$k-1$
D_k	$k-1$	$(k-1)/2$

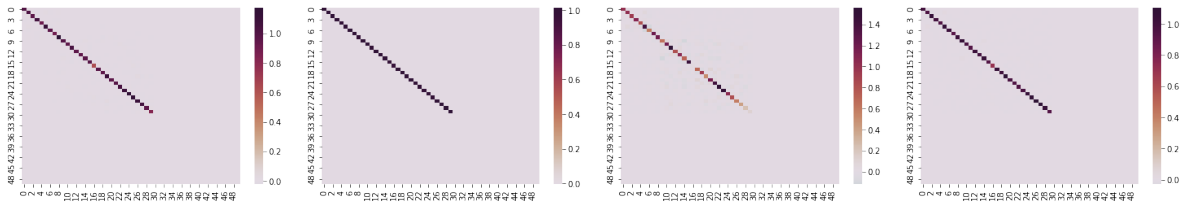
Table 5.18: The number of terms in the polynomial and the degree of the polynomial for each minimal-variance polynomial with degree parameter k.

Synthetic data examples

The first set of examples illustrating the differences between the original minimal-variance polynomial and the three new polynomials will use synthetically generated datasets.

Dataset 1 Let dataset 1 have $d = 50$ and $r = 30$, and produce a covariance matrix with diagonal entries `[10, 9, 8, 7, 6, 5, 4, 3, 2, 1] + [numpy.random.rand() for _ in range(10)] + [0.8 ** i for i in range(1, 11)] + [0] * 20`. This covariance matrix is used to produce a normally distributed dataset using the Python function `numpy.random.multivariate_normal` with mean zero and 1000 observations.

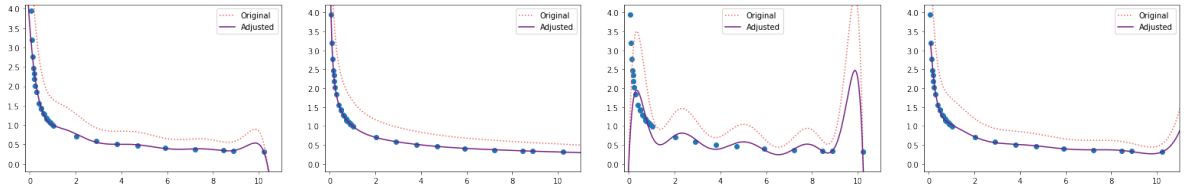
Figure 5.25 shows the heatmaps of the covariance matrices of the data after having been whitened by the four minimal-variance polynomial matrices. The heatmaps show polynomial B_{10} performs best, given it looks closest to the rank-deficient identity matrix.



(a) Polynomial A_{10} (b) Polynomial B_{10} (c) Polynomial C_{11} (d) Polynomial D_{11}

Figure 5.25: Heatmaps of the covariance matrix of dataset 1 after being whitened by the minimal-variance polynomial whitening method, using the four different polynomials detailed in Table 5.17.

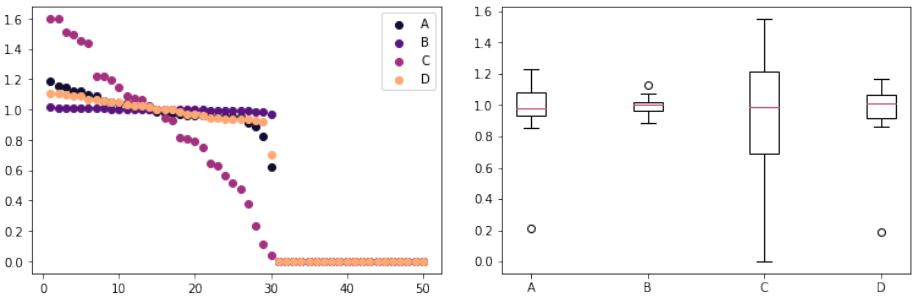
Figure 5.26 gives plots of the polynomials, comparing them to the inverse square root of the nonzero eigenvalues. The polynomials using $\Sigma^{1/2}$ (polynomials B_k and D_k) are smoother, possibly because they are of lower degree than polynomials A_k and C_k (see Table 5.18). The polynomials have been adjusted, following the methodology of Section 5.2.3 as discussed previously.



(a) Polynomial A_{10} (b) Polynomial B_{10} (c) Polynomial C_{11} (d) Polynomial D_{11}

Figure 5.26: Plots of the polynomials fit to the inverse square root eigenvalues of the covariance matrix of dataset 1, using the four different polynomials detailed in Table 5.17.

Polynomial C_k performs worst out of the polynomials considered. Figure 5.27 shows the eigenvalues of the covariance matrix of the transformed dataset through both a scatter plot and boxplots, and shows that polynomial C_k has a wide range of nonzero eigenvalues. The other polynomials perform relatively well, although polynomial B_k is clearly the most successful in whitening this dataset as the nonzero eigenvalues are closely centered around 1.

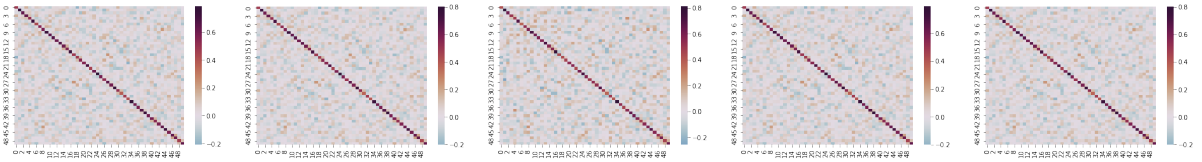


(a) All eigenvalues (b) Nonzero eigenvalues

Figure 5.27: Eigenvalues of the covariance matrix of dataset 1 after being transformed by a minimal-variance polynomial matrix. Figure (a) shows a plot of all eigenvalues in order, including zero eigenvalues. Figure (b) shows boxplots of the nonzero eigenvalues.

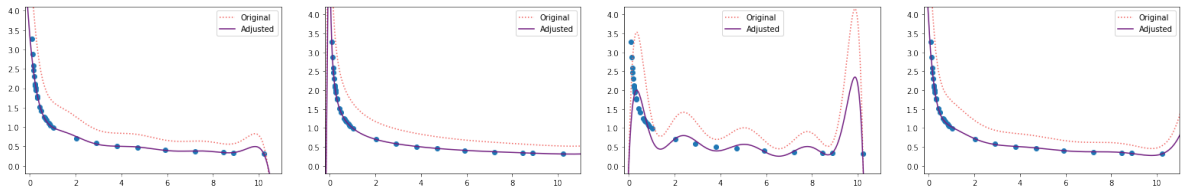
Dataset 1 Rotated Consider the same eigenvalues as dataset 1, but here introduce some rotation into the covariance matrix (see Appendix C.1 for the method used). There is no change in the behaviour of the polynomials or the value of the eigenvalues (see Figure 5.29 and Figure 5.30).

However, the rotation is not always removed easily when data is degenerate. This is illustrated by Figures 5.28a-5.28d, where the off-diagonals of the covariance matrix have nonzero values. However, this is also the case when using the Moore-Penrose pseudoinverse, as shown in Figure 5.28e. It would be recommended to apply iterative minimal-variance whitening in this case, as Section 5.4 has shown that this method is often more successful at decorrelating data.



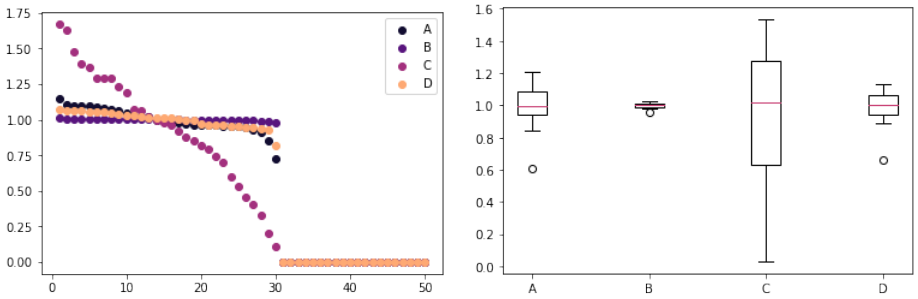
(a) Polynomial A_{10} (b) Polynomial B_{10} (c) Polynomial C_{11} (d) Polynomial D_{11} (e) MP Pseudoinverse

Figure 5.28: Heatmaps of the covariance matrix of dataset 1 with rotations (a)-(d) after being whitened by the minimal-variance polynomial whitening method, using the four different polynomials detailed in Table 5.17; (e) after being whitened by the square root of the Moore-Penrose pseudoinverse.



(a) Polynomial A_{10} (b) Polynomial B_{10} (c) Polynomial C_{11} (d) Polynomial D_{11}

Figure 5.29: Plots of the minimal-variance polynomials using the four different polynomials detailed in Table 5.17. The eigenvalues of the covariance matrix of dataset 1 with rotations are plotted against their reciprocal square root values.



(a) All eigenvalues (b) Nonzero eigenvalues

Figure 5.30: Eigenvalues of the covariance matrix of the dataset 1 with rotations after being transformed by a minimal-variance polynomial matrix. Figure (a) shows a plot of all eigenvalues in order, including zero eigenvalues. Figure (b) shows boxplots of the nonzero eigenvalues.

Dataset 2 Dataset 2 has been generated to have correlations, rather than introducing correlations later. Again, the `numpy.random.multivariate_normal` Python function is used, with mean equal to zero and number of observations equal to 1000. The covariance matrix for this example is computed using the code given in Snippet C.4 in Appendix C.3, and is illustrated by the heatmap in Figure 5.31. The final 20 dimensions of the covariance matrix are multiplied by increasingly smaller values to introduce degeneracy steadily. The eigenvalues of the empirical covariance matrix of this dataset are [260.96, 9.22, 7.8, 7.51, 6.26, 5.48, 5.16, 4.73, 4.45, 3.95, 3.31, 2.8, 2.72, 2.26, 2.03, 1.66, 1.55, 1.38, 1.29, 1.09, 0.87, 0.71, 0.47, 0.43, 0.33, 0.2, 0.13, 0.11, 0.08, 0.05, 0.01] + [0] * 19. This is a realistic scenario for a high dimensional dataset, as high dimensional datasets often have one large eigenvalue, and the rest much smaller [19].

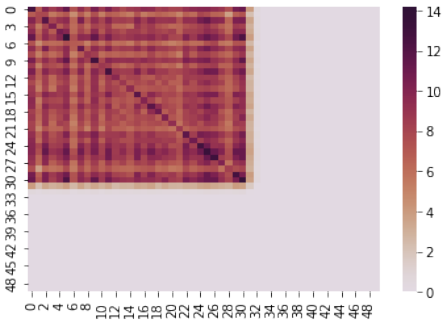
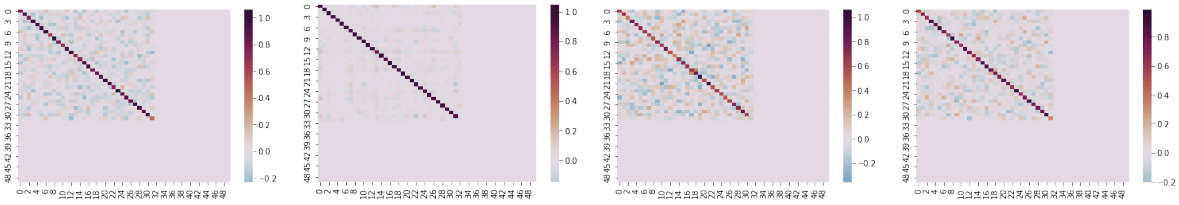
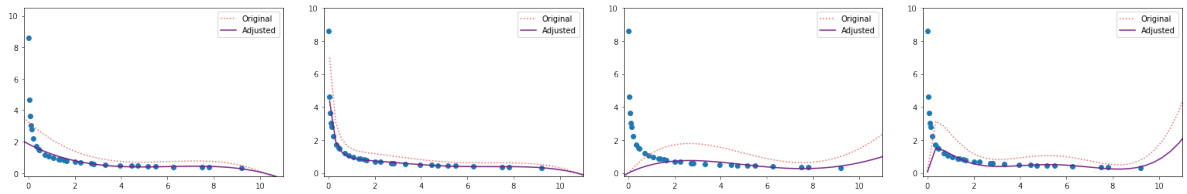


Figure 5.31: Heatmap of the empirical covariance matrix of dataset 2.



(a) Polynomial A_9 (b) Polynomial B_9 (c) Polynomial C_{10} (d) Polynomial D_{10}

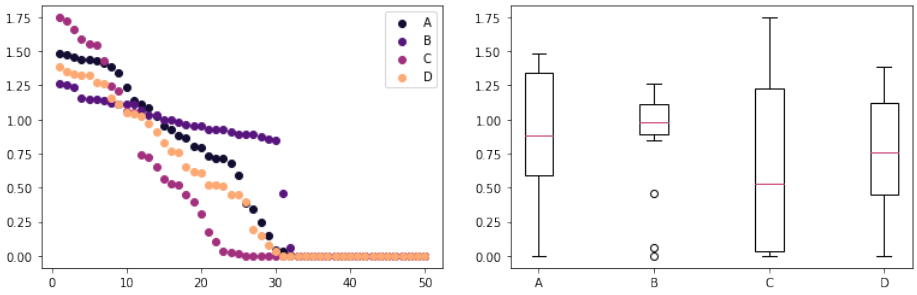
Figure 5.32: Heatmaps of the covariance matrix of dataset 2 after being whitened by the minimal-variance polynomial whitening method, using the four different polynomials detailed in Table 5.17.



(a) Polynomial A_9 (b) Polynomial B_9 (c) Polynomial C_{10} (d) Polynomial D_{10}

Figure 5.33: Plots of the minimal-variance polynomials using the four different polynomials detailed in Table 5.17. The eigenvalues of the covariance matrix of dataset 2 are plotted against their reciprocal square root values.

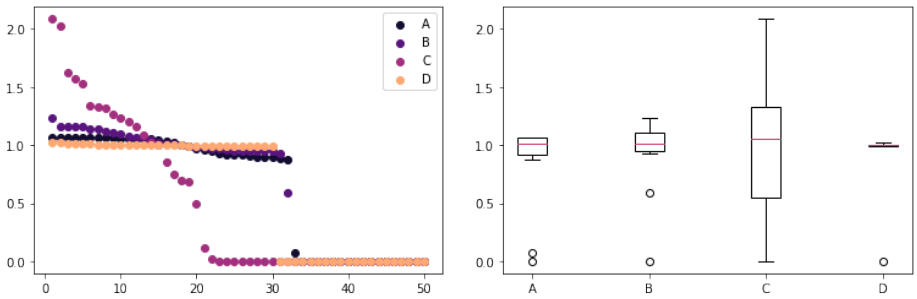
Figure 5.32 shows that the minimal-variance whitening polynomials remove most of the correlations seen in the original covariance matrix (Figure 5.31), but do not whiten the dataset perfectly. Polynomial B_9 performs best according to the heatmaps and the fit of the polynomial B in Figure 5.33. The spread of eigenvalues in Figure 5.34 also corroborates this, although shows that there is improvement to be made. Improvements can be made on the whitening performance by using iterative whitening, as in Section 5.4, with the different polynomials.



(a) All eigenvalues (b) Nonzero eigenvalues

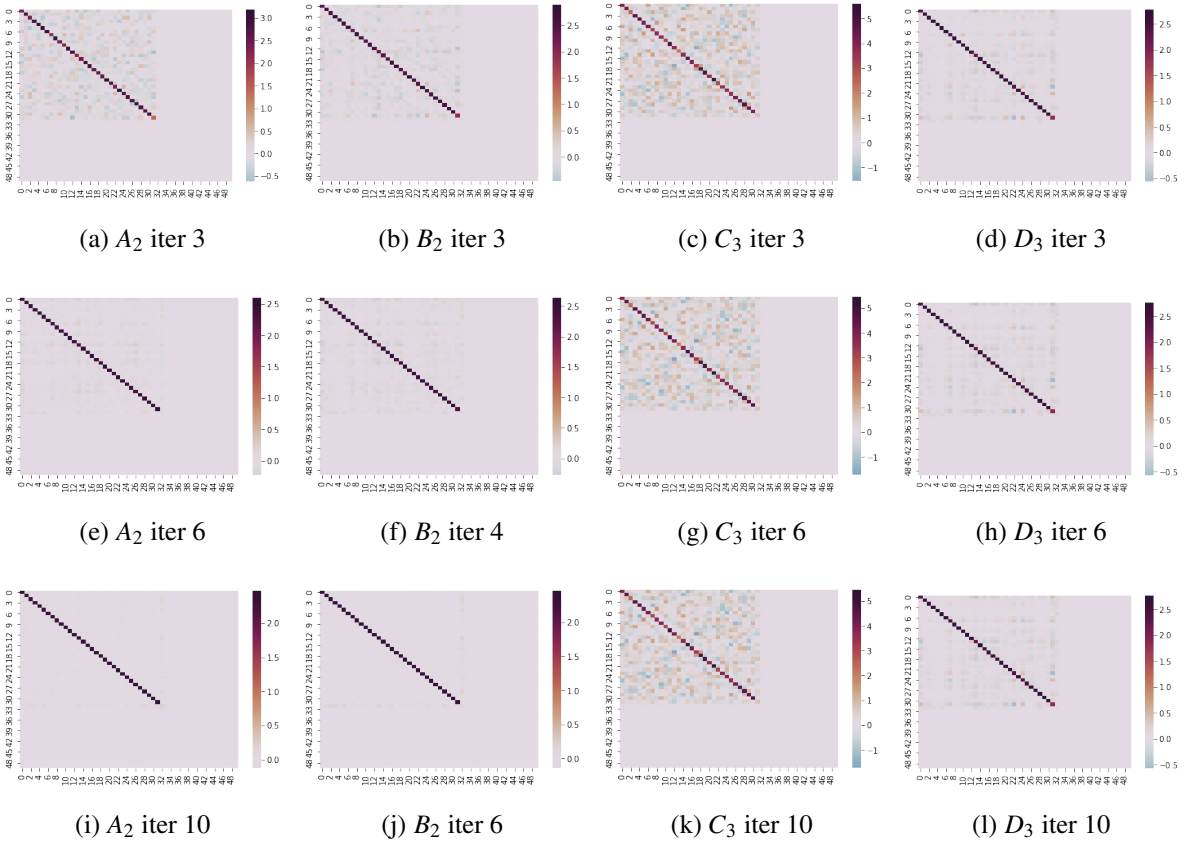
Figure 5.34: Eigenvalues of the covariance matrix of dataset 2 after being transformed by a minimal-variance polynomial matrix. Figure (a) shows a plot of all eigenvalues in order, including zero eigenvalues. Figure (b) shows boxplots of the nonzero eigenvalues.

Dataset 2 with iterative whitening The parameter $k = 2$ is used for polynomials A and B, and $k = 3$ for polynomials C and D, to ensure the polynomials have 2 terms in each iteration (see Table 5.18). Figure 5.36 shows that polynomials A and B whiten the dataset well when used with iterative whitening, but polynomials C and D struggle to remove correlations due to the lack of a regularizing term (the identity matrix). Figure 5.35 shows a large improvement in the resulting eigenvalues of the whitened dataset when using polynomials A, B and D when compared with Figure 5.34. Polynomial D does particularly well in retrieving the desired eigenvalues, but does not remove correlations as well as polynomials A and B, so one of the regularized polynomials would likely be recommended for this example.



(a) All eigenvalues (b) Nonzero eigenvalues

Figure 5.35: Eigenvalues of the covariance matrix of dataset 2 after being transformed by iterative whitening with the four different minimal-variance polynomial matrices. Figure (a) shows a plot of all eigenvalues in order, including nonzero eigenvalues. Figure (b) shows boxplots of the nonzero eigenvalues.



(a) A_2 iter 3 (b) B_2 iter 3 (c) C_3 iter 3 (d) D_3 iter 3

(e) A_2 iter 6 (f) B_2 iter 4 (g) C_3 iter 6 (h) D_3 iter 6

(i) A_2 iter 10 (j) B_2 iter 6 (k) C_3 iter 10 (l) D_3 iter 10

Figure 5.36: Iterative whitening on dataset 2 with the four minimal-variance polynomials given in Table 5.17.

Real data examples

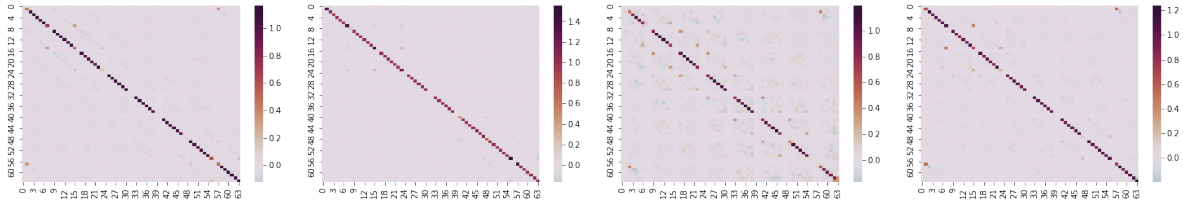
Two of the real datasets introduced in Table 5.2 will now be considered with the new polynomial methods, using both non-iterative and iterative whitening methods.

Digits The Digits dataset is whitened with the four different polynomials, and the heatmaps of the resulting covariance matrices are shown in Figure 5.37. Polynomials A, B and D provide approximate whitening, whereas polynomial C doesn't perform as well.

All polynomials have a good fit to the larger eigenvalues (mostly eigenvalues greater than 1), as shown in Figure 5.38. Polynomial C_{11} sees more oscillation than the other polynomials, suggesting it may be slightly less stable than the others. Polynomial B_{10} performs best in approximating the majority of the eigenvalues well.

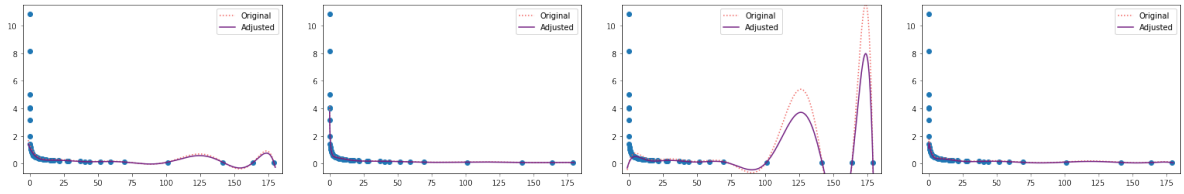
Regarding the eigenvalues of the covariance matrix of the transformed data, Figure 5.39a shows there is a steady decline to the zero eigenvalues for all four methods. Figure 5.39b

shows the distribution of the nonzero eigenvalues for each method considered. Polynomial B has nonzero eigenvalues largely concentrated around 1, whereas the other polynomials see a much larger spread. In this case, polynomial B performs best. However, iterative whitening may be able to improve this whitening.



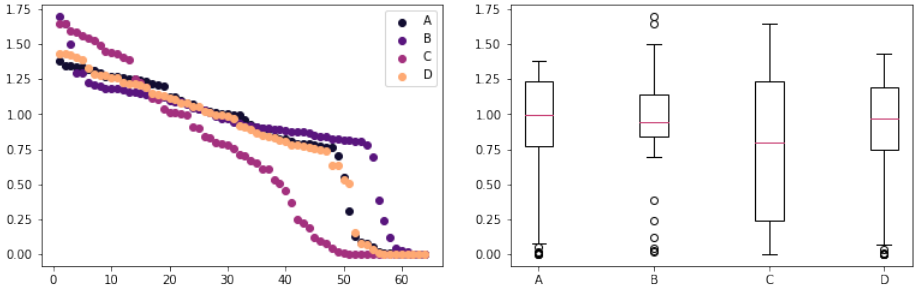
(a) Polynomial A_{10} (b) Polynomial B_{10} (c) Polynomial C_{11} (d) Polynomial D_{11}

Figure 5.37: Heatmaps of the covariance matrix of the Digits dataset after being whitened by the minimal-variance polynomial whitening method, using the four different polynomials detailed in Table 5.17.



(a) Polynomial A_{10} (b) Polynomial B_{10} (c) Polynomial C_{11} (d) Polynomial D_{11}

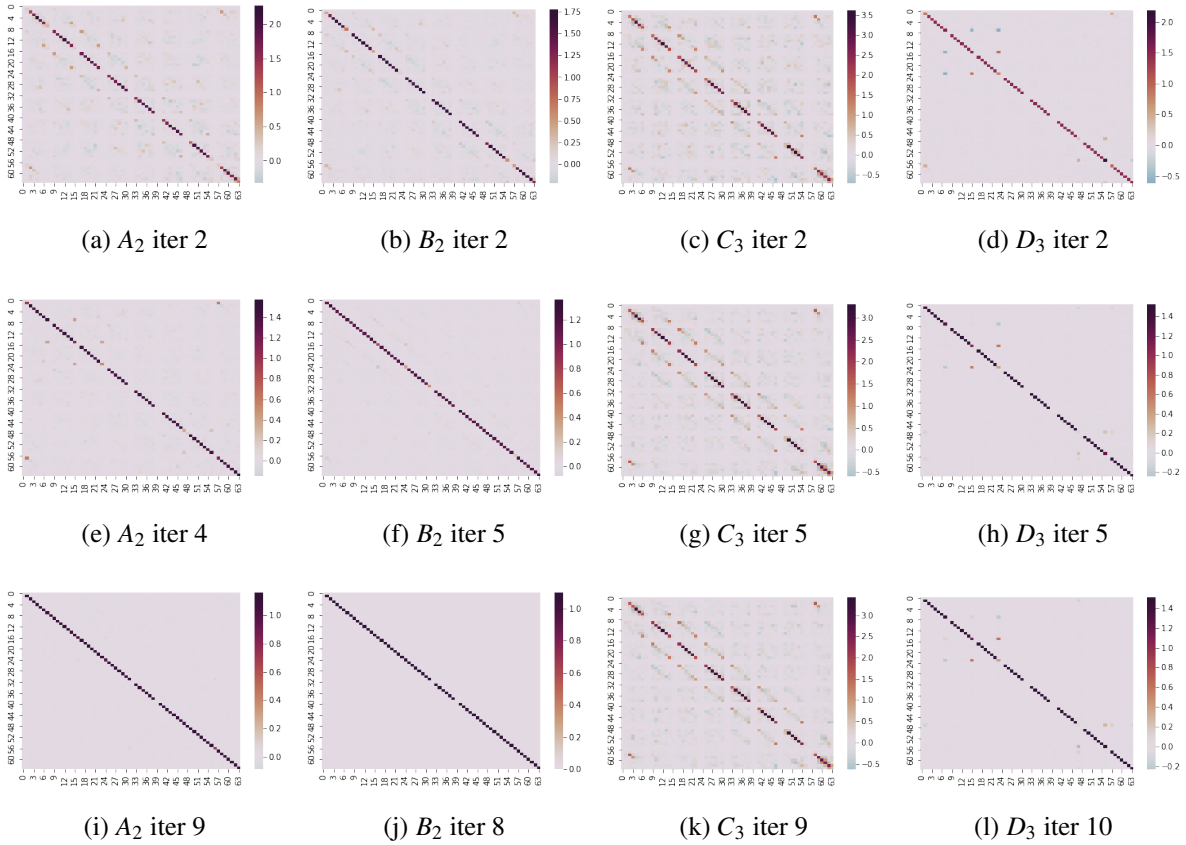
Figure 5.38: Plots of the polynomials fit to the inverse square root eigenvalues of the covariance matrix of the Digits dataset, using the four different polynomials detailed in Table 5.17.



(a) All eigenvalues (b) Nonzero eigenvalues

Figure 5.39: Eigenvalues of the covariance matrix of the Digits dataset after being transformed by a minimal-variance polynomial matrix. Figure (a) shows a plot of all eigenvalues in order, including zero eigenvalues. Figure (b) shows boxplots of the nonzero eigenvalues.

Digits with iterative whitening Minimal-variance whitening can now be compared with iterative minimal-variance whitening on the Digits dataset. The degree parameter $k = 2$ is used for polynomials A and B, and $k = 3$ for polynomials C and D. Polynomials A and B whiten the data perfectly after several iterations, as seen in Figure 5.40. Polynomial C does not see much of an improvement using iterative whitening. The heatmap relating to polynomial D indicates a good whitening transformation after several iterations. However, the rank of the dataset has been changed when using this method.



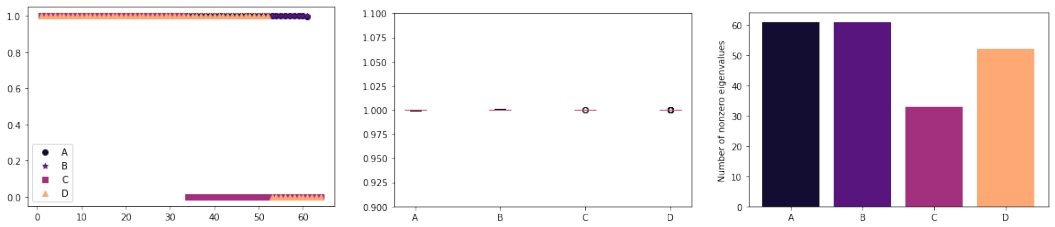
(a) A_2 iter 2 (b) B_2 iter 2 (c) C_3 iter 2 (d) D_3 iter 2

(e) A_2 iter 4 (f) B_2 iter 5 (g) C_3 iter 5 (h) D_3 iter 5

(i) A_2 iter 9 (j) B_2 iter 8 (k) C_3 iter 9 (l) D_3 iter 10

Figure 5.40: Iterative whitening on the Digits dataset with the four minimal-variance polynomials given in Table 5.17.

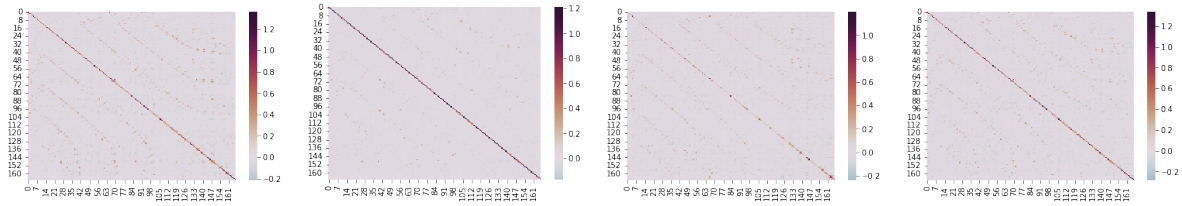
Figure 5.41b shows that when using each polynomial, the eigenvalues are all zeros and ones. However, Figure 5.41a also shows that there are different amounts of zeros and ones depending on which polynomial is used, and this is illustrated more clearly in the bar chart given in Figure 5.41c. Therefore, despite polynomial D's seemingly good performance in the heatmaps and boxplots, it does not perform as well as polynomials A and B due to lack of regularization.



(a) All eigenvalues (b) Nonzero eigenvalues (c) Count of nonzero eigenvalues

Figure 5.41: Eigenvalues of the covariance matrix of the Digits dataset after being transformed by iterative whitening with the four different minimal-variance polynomial matrices. Figure (a) shows a plot of all eigenvalues in order, including nonzero eigenvalues. Figure (b) shows boxplots of the nonzero eigenvalues. Figure (c) shows the number of nonzero eigenvalues of the covariance matrix after iterative whitening using each polynomial.

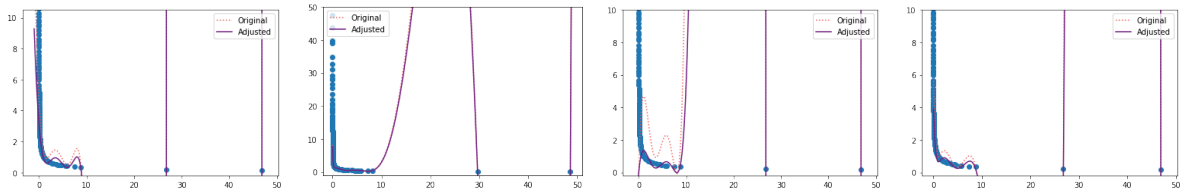
Musk The above exercise is repeated on the Musk dataset. Clearly, considering the heatmaps in Figure 5.42, polynomial B_9 performs the best at removing correlations (see Figure 5.5f for the heatmap of the original covariance matrix).



(a) Polynomial A_9 (b) Polynomial B_9 (c) Polynomial C_{10} (d) Polynomial D_{10}

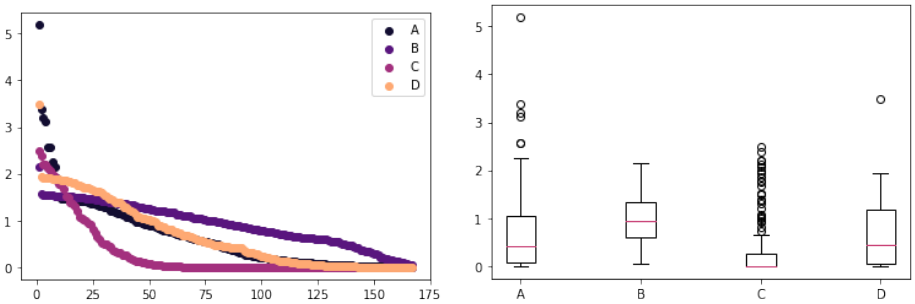
Figure 5.42: Heatmaps of the covariance matrix of the Musk dataset after being whitened by the minimal-variance polynomial whitening method, using the four different polynomials detailed in Table 5.17.

Polynomial B_9 also provides the best polynomial fit in Figure 5.43, as the other polynomials do not fit the eigenvalues between 0 and 10 very closely. However, Figure 5.44 shows that none of the polynomials perform particularly well when considering the eigenvalues of the transformed data. As with the other examples, iterative whitening can be used to improve these results.



(a) Polynomial A_9 (b) Polynomial B_9 (c) Polynomial C_{10} (d) Polynomial D_{10}

Figure 5.43: Plots of the polynomials fit to the inverse square root eigenvalues of the covariance matrix of the Musk dataset, using the four different polynomials detailed in Table 5.17.

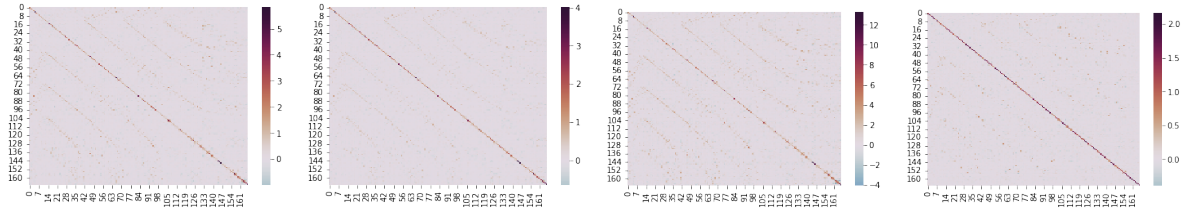


(a) All eigenvalues (b) Nonzero eigenvalues

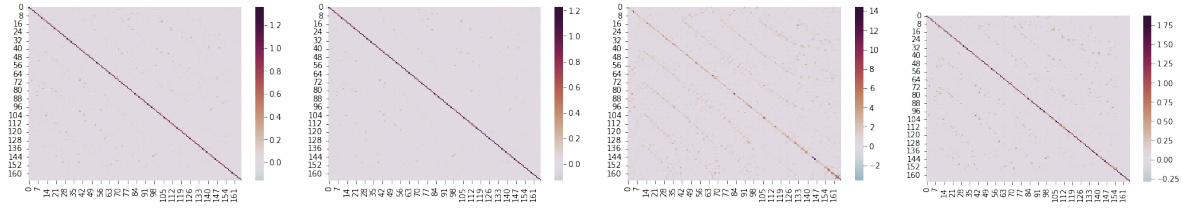
Figure 5.44: Eigenvalues of the covariance matrix of the Musk dataset after being transformed by a minimal-variance polynomial matrix. Figure (a) shows a plot of all eigenvalues in order, including zero eigenvalues. Figure (b) shows boxplots of the nonzero eigenvalues.

Musk with iterative whitening As in previous examples, $k = 2$ is used for polynomials A and B, and $k = 3$ for polynomials C and D. Polynomials A and B clearly perform well in Figure 5.45. Polynomials C and D retain some correlations after the iterative-whitening transformation. Like the Digits dataset, the eigenvalues for all polynomials become exclusively zeros and ones, but polynomials C and D have the incorrect rank (Figure 5.46). Thus, polynomials A and B should be used with iterative whitening.

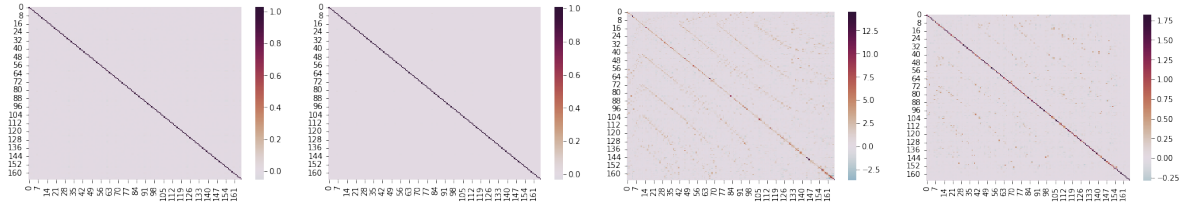
The overriding message is that polynomials A and B seems to work most consistently in whitening data successfully, particularly when combined with iterative whitening. Polynomials C and D do not behave as hoped in theory, particularly when combined with iterative whitening. This suggests that regularization by the identity matrix is important for minimal-variance whitening polynomials.



(a) A_2 iter 2 (b) B_2 iter 2 (c) C_3 iter 2 (d) D_3 iter 2

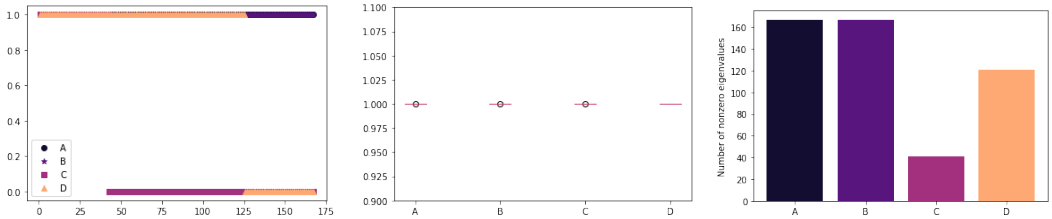


(e) A_2 iter 5 (f) B_2 iter 4 (g) C_3 iter 5 (h) D_3 iter 5



(i) A_2 iter 8 (j) B_2 iter 6 (k) C_3 iter 9 (l) D_3 iter 10

Figure 5.45: Iterative whitening on the Musk dataset with the four minimal-variance polynomials given in Table 5.17.



(a) All eigenvalues (b) Nonzero eigenvalues (c) Count of nonzero eigenvalues

Figure 5.46: Eigenvalues of the covariance matrix of the Musk dataset after being transformed by iterative whitening with the four different minimal-variance polynomial matrices. Figure (a) shows a plot of all eigenvalues in order, including nonzero eigenvalues. Figure (b) shows boxplots of the nonzero eigenvalues. Figure (c) shows the number of nonzero eigenvalues of the covariance matrix after iterative whitening using each polynomial.

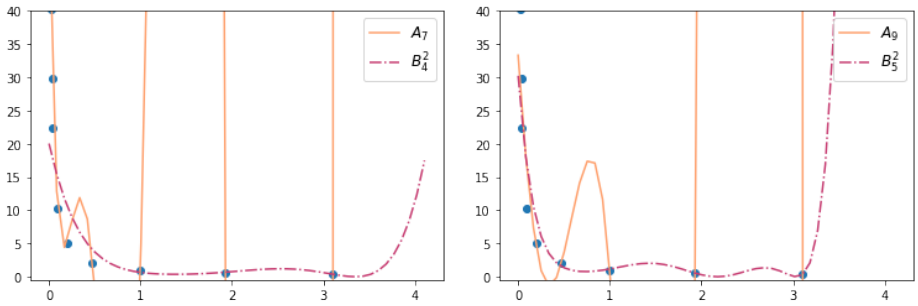
5.7 Squaring the polynomial to produce an alternative to Σ^{-1}

In Chapter 4, a minimal-variance polynomial approximation to Σ^{-1} is constructed and used to find the minimal-variance distances. For this section, denote this approximation to Σ^{-1} as A_k. An alternative method to finding such an approximation is to square the minimal-variance polynomial approximation to $\Sigma^{-1/2}$ that has been discussed in Chapter 5. For this section, denote the approximation to $\Sigma^{-1/2}$ as B_k, such that the approximation to Σ^{-1} using this method will be given by B_k^2. If the approximation B_k to $\Sigma^{-1/2}$ is a good one, the approximation B_k^2 to Σ^{-1} should also be good.

Using this method has a few potential benefits. Firstly, positive-definiteness of the approximation B_k^2 is guaranteed as it is the square of a matrix. Secondly, there is an iterative method of finding an approximation to $\Sigma^{-1/2}$ as detailed in Section 5.4, which often produces a more effective approximation than the standard minimal-variance polynomial. An equivalent iterative method for an approximation to Σ^{-1} has not been constructed, so using the iterative method for $\Sigma^{-1/2}$ and then squaring the approximation may be more effective.

Furthermore, finding B_k requires calculating a $(k-1)$-degree polynomial, and B_k^2 will therefore be a $2(k-1)$-degree polynomial. In theory, this should therefore be equivalent to calculating A_{2k-1} (since A_{2k-1} will be a $2k-2$ degree polynomial), but with much less computational cost and threat of instability. The comparisons are therefore between A_{2k-1} and B_k^2 in the examples that follow.

A simple example will be used first to explore this method. Let Σ be a diagonal matrix with eigenvalues [3, 2, 1, 1/2, 1/5, 1/10, 1/20, 1/30, 1/40, 1/50]. A dataset X is generated from a multivariate Gaussian distribution with zero mean, covariance matrix Σ and 1000 observations. In Figure 5.47, the eigenvalues are plotted against the inverse eigenvalues. The A_k polynomials are a closer fit to the inverse eigenvalues, but have higher oscillation. The B_k^2 polynomials provide a close fit, with a more simple polynomial shape. Table 5.19 shows the variances of the quadratic forms, using the formula from (4.1). In this example, a lower variance can be found using B_k^2, along with the guarantee of a positive-definite approximation to Σ^{-1}.



(a) A_7, B_4^2 (b) A_9, B_5^2

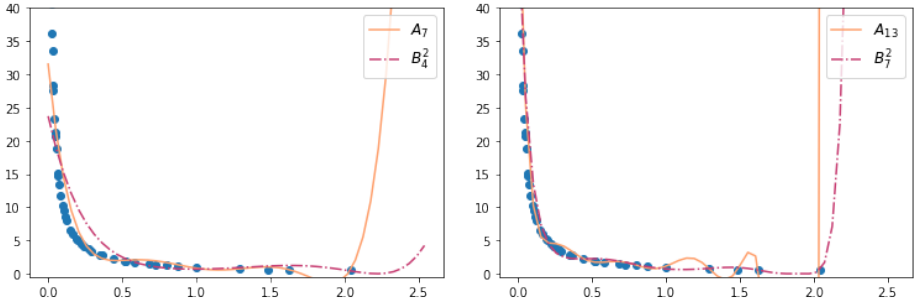
Figure 5.47: Plot of polynomials, where A_k (orange, solid line) approximates Σ^{-1}, and B_k^2 (pink dashed line) approximates $\left(\Sigma^{-1/2}\right)^2$. The blue dots are the eigenvalues plotted against their reciprocals.

Deg	A_{2k-1}	$\text{var}(y^\top A_{2k-1} y)$	B_k^2	$\text{var}\left(y^\top B_k^2 y\right)$
4	A_5	23.851	B_3^2	42.243
6	A_7	**20.223**	B_4^2	30.122
8	A_9	21.335	B_5^2	24.148
10	A_{11}	22.241	B_6^2	20.972
12	A_{13}	33.014	B_7^2	**20.198**
14	A_{15}	90.387	B_8^2	20.213
16	A_{17}	28.150	B_9^2	21.289

Table 5.19: Variances of the quadratic forms $y^\top A_{2k-1} y$ and $y^\top B_k^2 y$ for different values of k, where $y = x - \mu$, for a 10-dimensional dataset. The 'Deg' column denotes the degree of the polynomial.

The iterated minimal-variance approximation to $\Sigma^{-1/2}$ is detailed in Section 5.4. Use this iterative method to find the approximation to $\Sigma^{-1/2}$ and square this to find the approximation to Σ^{-1}. The variance when using this method with $k = 2$ and 6 iterations is 20.000002, which is lower than any values in Table 5.19.

A second dataset is generated as above, with eigenvalues `[2, 1.7, 1.5, 1.2, 1] + [0.9 ** i for i in range(1, 46)]`. Figure 5.48 shows the fit of the polynomials A_{2k-1} and B_k^2 for $k = 4$ and $k = 7$. As the eigenvalues degenerate consistently, the polynomial fit of B_k^2 is very good, particularly for $k = 7$.



(a) A_7, B_4^2 (b) A_{13}, B_7^2

Figure 5.48: Plot of polynomials, where A_k (orange, solid line) approximates Σ^{-1}, and B_k^2 (pink dashed line) approximates $\left(\Sigma^{-1/2}\right)^2$. The blue dots are the eigenvalues plotted against their reciprocals.

Deg	A_{2k-1}	$\text{var}(y^\top A_{2k-1} y)$	B_k^2	$\text{var}\left(y^\top B_k^2 y\right)$
4	A_5	128.543	B_3^2	191.548
6	A_7	114.937	B_4^2	154.558
8	A_9	108.711	B_5^2	132.676
10	A_{11}	109.746	B_6^2	121.473
12	A_{13}	**105.532**	B_7^2	114.404
14	A_{15}	127.455	B_8^2	109.997
16	A_{17}	152.759	B_9^2	106.843
18	A_{19}	426.547	B_{10}^2	**104.546**
20	A_{21}	727.745	B_{11}^2	105.415

Table 5.20: Variances of the quadratic forms $y^\top A_{2k-1} y$ and $y^\top B_k^2 y$ for different values of k, where $y = x - \mu$, for a 50-dimensional dataset. The 'Deg' column denotes the degree of the polynomial.

Table 5.20 gives the variance of the quadratic forms using A_{2k-1} and B_k^2 for different values of k. A lower variance is achieved when using B_k^2. Using the iterative minimal-variance whitening method with $k = 2$ gives a variance of exactly 100, a further improvement on the two non-iterative methods.

These examples suggest that it may often be beneficial to use the minimal-variance polynomial approximation to $\Sigma^{-1/2}$ squared to find an approximation to Σ^{-1}, particularly by using the iterative minimal-variance polynomial method.

5.8 Chapter summary

This chapter has introduced the concept of minimal-variance whitening, a polynomial-based method intended to decorrelate and standardize datasets. The method was first introduced in a paper published by my supervisors and I [85]. The stand-out benefits of using the minimal-variance whitening method include:

- The ability to whiten datasets that have singular covariance matrices. This is not possible using many classical whitening methods, such as Mahalanobis whitening and Cholesky whitening;
- The improvement in whitening for degenerate and near-degenerate datasets using this method when compared to the Mahalanobis whitening method using the Moore-Penrose pseudoinverse;
- The adjustability offered by the method, thanks to different parameter settings, including the degree parameter k. There are many other avenues for flexibility, such as different constraints, different polynomial forms and different weightings in the constraint adjustment method;
- The improvement offered to the method by the iterative minimal-variance whitening methods;
- The potential improvement for minimal-variance distances by using whitening and the Euclidean distance, or the squared approximation to Σ^{-1} given in Section 5.7.

Some of the limitations of this method include:

- The potential for instability if k is chosen too high, as is the case with most polynomial-based methods. The iterative method introduced in Section 5.4 reduces this issue greatly;
- Perfect whitening is not always achieved, but again this is improved greatly by the iterative whitening method;
- The computational expense of calculating powers and traces of powers of matrices is high, particularly in high dimensions. Some techniques to alleviate this cost are given in Section 5.3.3.

Section 5.2 constructs the minimal-variance polynomial through Lagrangian methods, and has a thorough discussion of how to set the different parameters used in the minimal-variance polynomial. It is concluded that low values of the parameter k should be used to provide more stable and low-cost outputs by producing lower degree polynomials. The best value for the parameter α is seen to be $\alpha = 1/2$, as it produces an unbiased estimator and good empirical results. However, using this parameter enforces a need to adjust the constraint when whitening rank-deficient data. Recommendations for such an adjustment are given, and are shown to help identify whether a matrix is singular or not.

Several numerical examples of the minimal-variance whitening method are given in Section 5.3. It is shown that the method is useful in settings where $d < N$ and $d \geq N$ for a range of applicatons, including outlier detection and dimension reduction. The method is compared to a number of different whitening methods, including Mahalanobis whitening, Mahalanobis whitening using the Moore-Penrose pseudoinverse and other pre-processing methods. Several methods are suggested in this section to make the minimal-variance whitening method accessible when using extremely high dimensional datasets, including sampling, random projection and trace approximation methods. Overall, this section illustrates the varied applicability and good empirical performance of the minimal-variance whitening method, and that it can often outperform established whitening methods.

Section 5.4 introduces an extension to minimal-variance whitening called the 'iterative minimal-variance whitening' method. The foundations of this method are to repeatedly whiten the output of the minimal-variance whitening procedure, until some stopping criterion or convergence is reached. Several real datasets are used to illustrate the success of this method, and vast improvements are made on both the original minimal-variance and Moore-Penrose Mahalanobis whitening methods. Through these empirical investigations, it is noted that the iterative minimal-variance whitening method should be applied using the parameter $k = 2$, for the sake of simplicity, stability and performance. However, different values of k can be used interchangeably should the user want.

Section 5.5 illustrates how the constraint adjustment introduced in Section 5.2.3 can be used to estimate the rank of a singular matrix, using the original minimal-variance whitening method or the iterative version. This method of rank estimation is shown to be more consistent than other methods, such as eigenvalue thresholding.

Different forms of the minimal-variance polynomial are explored in Section 5.6 for whitening. Polynomials that capitalize on the need to calculate $\Sigma^{1/2}$ are used, and some non-regularized polynomials are experimented with. It is shown that the original minimal-variance polynomial and a regularized polynomial using $\Sigma^{1/2}$ both perform well, but that the non-regularized methods struggle, particularly with degenerate data and when applied in iterative whitening.

In the final section of this chapter, comparisons are made between the approximation to Σ^{-1} given in Chapter 4 and using the square of the approximation to $\Sigma^{-1/2}$ given in Chapter 5 to approximate Σ^{-1}. Using an iterative method to approximate $\Sigma^{-1/2}$ means more accurate results are often found, and as such an improvement can be made in approximating the inverse of the covariance matrix. This method also ensures that such an approximation to Σ^{-1} is positive-definite, which may not otherwise be guaranteed.

Overall, the minimal-variance polynomial approximation to $\Sigma^{-1/2}$ is highly adjustable and provides good data whitening capabilities, which are needed for a wide range of applications. It is available for use in the case of degeneracy, unlike the popular methods of whitening outlined in [130]. Minimal-variance whitening often outperforms these established methods when they are available, especially when it is used iteratively. The method has other applications outside of data whitening, such as estimation of the rank of a matrix and approximating the inverse of a covariance matrix. There are many possible avenues for further research into the methods and applications of the minimal-variance polynomials, which will be explored in Section 6.2.

Chapter 6

Conclusion

This chapter provides a overview of the research reported in this thesis. An in-depth conclusion can be found at the end of each chapter, so Section 6.1 is made up of summaries which repeat the key points of these conclusions. Potential avenues for future research related to the topics in this thesis are then provided in Section 6.2.

6.1 Summary of research contributions

In this section, an outline of the novel contributions of this thesis is given. Chapter 1 provides an introduction to the topics considered, and gives an overview of the structure of the thesis. Chapter 2 is a review of the literature related to the topics discussed in this thesis. The reliance on distance measures in multivariate analysis is highlighted, as well as the need for methods of data whitening across many fields and applications. The implications of working with high dimensional data for distance measures are also explored. A discussion of estimators for the covariance and inverse covariance matrix is given, illustrating the reliance on structural assumptions for such estimators. This highlights the need for new methodology to produce multivariate distance measures and methods of data whitening in high dimensions, with no such structural assumptions.

The following paragraphs explore Chapters 3-5, as these chapters contain the novel contributions of the thesis.

Chapter 3: Simplicial distances Chapter 3 considers the simplicial distances, which were first proposed by Pronzato et al. [192] but have been extended in [85] and in this thesis. The distance from a point x to a set of points X is found by calculating the average volume of all k-dimensional simplices formed by x and points in X. As such, the simplicial distances are free from the assumptions and limitations imposed by estimations of the covariance matrix, and can be used in the case of degenerate data by choosing the parameter k to be less than the rank of the data.

The parameter δ controls the behaviour of the distances, with comparisons drawn to the ℓ_δ distances for different parameter choices. The parameter k controls the dimension of the simplices used. It is shown that distances proportional to the Euclidean distance and the Mahalanobis distance can be found using $k = 1$ and $k = d$ respectively. All other values of k form a spectrum of distances that fall between these two measures in the sense of both behaviour and variances.

Computing the distances directly through simplex volumes can be computationally expensive, so alternative methods of computation are suggested to improve speed through elementary symmetric functions with polynomials and subsampling of simplices. It is shown that there is some instability in the elementary symmetric functional method, but only when the degree parameter k is chosen too high, which is not a recommended parameter choice for most polynomial methods.

The distances account for interactions between variables and varying scales, while also being well defined in the case of degenerate data, unlike the Mahalanobis distance. Through empirical examples, it is shown that the simplicial distances are robust and can outperform the Mahalanobis and Euclidean distances in circumstances with correlated, high dimensional data.

Chapter 4: Minimal-variance distances The family of minimal-variance distance measures were discussed in Chapter 4, having first been proposed in [85]. The minimal-variance matrix A_k is derived by finding a set of coefficients for a degree-$(k-1)$ polynomial in the covariance matrix Σ, where k is a user-defined parameter. The polynomial seeks to minimize the variance of the distances produced when using A_k in a quadratic form, subject to a constraint which ensures that A_k behaves like a (pseudo)inverse of Σ.

The method of finding the coefficients of this polynomial is approached from two directions: constrained polynomial approximation and weighted linear regression. The parameter α controls the constraint imposed on the minimal-variance matrix, and can alter the variance-bias trade-off of the estimator. It is shown that using $\alpha = 1$ produces an unbiased estimator. The chapter illustrates that the method is highly amenable, with comparisons between the constraint used throughout Chapter 4 and two other constraints, all of which have various benefits and drawbacks.

Much like the simplicial distances, it is recommended to use a lower value of the parameter k to avoid instability and to achieve practically good results. Through numerical examples, the minimal-variance distances are shown to be more efficient than the simplicial distances at reducing variance (and therefore behaving similarly to the Mahalanobis distance). Further numerical examples show that the minimal-variance distances perform well when applied to a variety of applications compared to the Euclidean, Mahalanobis and simplicial distances. The minimal-variance distances are shown to be a good alternative to the Mahalanobis distance, with the key advantage of being well-defined when the data considered is degenerate.

Chapter 5: Minimal-variance whitening Minimal-variance whitening was first introduced in [84]. The method was inspired by the results of the minimal-variance distances, and the desire to use Mahalanobis whitening in the case of degenerate data, which is not possible due to the non-existence of the inverse covariance matrix. A $(k-1)$-degree matrix polynomial A_k is constructed in Σ, with the coefficients found such that the total variation of the data transformed by this matrix A_k would be minimized. A constraint is imposed to ensure A_k behaves similarly to the inverse square root of the covariance matrix Σ, where it exists. The coefficients of the minimal-variance whitening polynomial are produced using a Lagrangian minimization method.

There are two (main) parameters for the minimal-variance whitening matrix A_k. The parameter α controls the behaviour of the constraint imposed, and it is shown that $\alpha = 1/2$ is the best choice for this parameter. The parameter k specifies the degree of the polynomial to be calculated. As k increases, so does the potential for more accurate results. However, as with many polynomial methods, the opportunity for instability also rises with k, meaning lower choices of k are recommended.

An iterative method is introduced, which can vastly improve the results and stability of minimal-variance whitening. Using a low value of k (often $k = 2$) the minimal-variance whitening method is applied repeatedly, until convergence or a stopping criterion is reached. Empirical results show this method is highly successful, and can improve on the results produced by the Mahalanobis whitening method with the Moore-Penrose inverse, particularly for datasets with $d > N$.

The minimal-variance whitening method is highly flexible. The constraint can be modified, the polynomial can take a different form, and a varying number of iterations with different values of k can be applied. This makes the method adaptable for different applications. Many applications of the minimal-variance whitening method other than data whitening are given in Chapter 5, including singularity detection, rank estimation and improved estimation of Σ^{-1}. The minimal-variance whitening method is therefore a usable and adaptive substitute for the inverse square root covariance matrix when working with singular data.

6.2 Future research directions

The potential for future avenues of research related to the contributions of this thesis is great. First and foremost, fully-tested software packages could be produced to allow for easy implementations of the methods. All methods could also benefit from more precise floating point computation to improve stablility with high values of k, but this is likely to be unrealistic for widespread implementations of the methods. Further applications of all methods can also be considered. For the two distance measures, more research could be done considering the interaction of these new distance measures with various data analysis methods, including density-based clustering, distance-based clustering, support vector machines and more. Regarding minimal-variance whitening, there is a huge range of applications to be experimented with, including approximate Bayesian computation, neural networks and image recognition.

The rest of this section considers more specific research directions for each of the novel methods presented in this thesis.

Chapter 3: Simplicial distances The potential for computational speed improvement in finding the simplicial distances is great. Implementations of this distance measure could make use of parallel processing methods, particularly when computing the distance directly through simplex volumes. The volumes of simplices could be computed asynchronously, or the distances between points could be found using a synchronous process. Furthermore, more research could be done on the distance measure with $\delta = 1$ in high dimensions, as the literature implies that ℓ_1 distances are more effective than ℓ_2 distances in such cases.

Chapter 4: Minimal-variance distances One of the possible limitations of the minimal-variance distances is the reliance on the sample covariance matrix, particularly for HDLSS data. A potential avenue of future research is therefore the application of minimal-variance distances using an alternative estimator to the covariance matrix. However, as touched upon in Section 2.5, one must be mindful of the assumptions imposed on the data by many covariance estimators.

Chapter 5: Minimal-variance whitening It is evident that using a different polynomial form could provide improvements in performance. Polynomial B presented in Section 5.6 appears to be stable and have good performance, so further research into this particular polynomial is suggested. Much like the minimal-variance distances, the minimal-variance whitening method could also be considered with estimators other than the sample covariance matrix. Furthermore, it has been shown that the minimal-variance whitening matrix can be squared to approximate Σ^{-1} well. As such, more comparisons to the Mahalanobis distance and the minimal-variance distances could be performed, particularly using the polynomial B from Section 5.6 and the iterative minimal-variance whitening method.

Appendix A

Moments and Distributions of Distances

A.1 Moments of quadratic forms

From Lemma 6.1 of [162], let y be a random vector with mean 0 and covariance matrix Σ, and let A be a symmetric matrix. The expectation of the quadratic form $y^\top Ay$ is given by

$$\mathrm{E}\left(y^\top Ay\right) = \mathrm{trace}(A\Sigma).$$

If y is normally distributed with sample mean 0 and sample covariance matrix Σ, then the variance of the quadratic form is given by

$$\mathrm{Var}\left(y^\top Ay\right) = 2\mathrm{trace}\left([A\Sigma]^2\right).$$

If the above conditions hold, the skewness and kurtosis of the quadratic form can also be defined respectively as:

$$\mathrm{Skew}\left(y^\top Ay\right) = 2\sqrt{2}\mathrm{trace}\left([A\Sigma]^3\right)\left\{\mathrm{trace}\left([A\Sigma]^2\right)\right\}^{-3/2},$$

$$\mathrm{Kurt}\left(y^\top Ay\right) = 12\mathrm{trace}\left([A\Sigma]^4\right)\left\{\mathrm{trace}\left([A\Sigma]^2\right)\right\}^{-2}.$$

The distances considered in this thesis are generalized squared distances of the form

$$\rho_A^2(x, X) = (x-\mu)^\top A(x-\mu),$$

where x is assumed to be normally distributed with sample mean μ and sample covariance matrix Σ. Therefore y is replaced by $(x-\mu)$, which is normally distributed with zero mean. Throughout this thesis, the matrix A is often defined to be a weighted sum of the covariance matrix Σ, so is a symmetric matrix by definition. Therefore, the distances satisfy the conditions for the moments above to hold.

If the covariance matrix Σ is of full-rank, Corollary 3.2a.1 of [171] states that the moment generating function (MGF) of the quadratic form $\rho_A^2(x,X)$ (where $(x-\mu) \sim \mathcal{N}(0,\Sigma)$) is given by

$$M_{\rho_A^2(x,X)}(t) = \det(I - 2tA\Sigma)^{-1/2},$$

where I is the $d \times d$ covariance matrix and d is the dimensionality of the dataset.

If the covariance matrix is singular with rank $r < d$, the MGF needs to be adapted. Consider the rank decomposition $\Sigma = BB^\top$, where B is a $d \times r$ matrix of rank r. Corollary 3.2a.2 of [171] states that, if $B^\top AB$ is not singular, the MGF of the quadratic form $\rho_A^2(x,X)$ is given by

$$M_{\rho_A^2(x,X)}(t) = \det\left(I - 2tB^\top AB\right)^{-1/2}.$$

A.2 Moments of the simplicial distances with $\delta = 2$

For a point $x \in X$, where X is normally distributed with sample mean μ and sample covariance matrix Σ, consider $\rho_{k,2}^2(x,X) = (x-\mu)^\top \frac{S_k}{k}(x-\mu)$, i.e. the k-simplicial distance from the sample mean μ to a point x with $\delta = 2$. Replace the matrix A in the moment conditions above with the matrix S_k/k to retrieve the following formulae:

$$\begin{aligned}
\mathrm{E}\left(\rho_{k,2}^2(x,X)\right) &= \frac{1}{k}\mathrm{trace}(S_k\Sigma) \\
\mathrm{Var}\left(\rho_{k,2}^2(x,X)\right) &= \frac{2}{k^2}\mathrm{trace}\left([S_k\Sigma]^2\right) \\
\mathrm{Skew}\left(\rho_{k,2}^2(x,X)\right) &= 2\sqrt{2}\mathrm{trace}\left([S_k\Sigma]^3\right)\left\{\mathrm{trace}\left([S_k\Sigma]^2\right)\right\}^{-3/2} \\
\mathrm{Kurt}\left(\rho_{k,2}^2(x,X)\right) &= 12\mathrm{trace}\left([S_k\Sigma]^4\right)\left\{\mathrm{trace}\left([S_k\Sigma]^2\right)\right\}^{-2}.
\end{aligned} \tag{A.1}$$

Note that for the third and fourth standardized moments, the scalar value $1/k$ cancels out.

A.3 Moments of the squared Euclidean distances

The formulae for the moments of quadratic forms given in Section A.1 is applied to the squared Euclidean distance. The squared Euclidean distance is divided by trace(Σ) here, to make it equivalent to the simplicial distance with $\delta = 2$, $k = 1$. Let λ_i, $i \in \{1, 2, \ldots, d\}$ be the eigenvalues of Σ, and let $A = \frac{I}{\text{trace}(\Sigma)}$. The moments of $\rho^2_{1,2}(x, X)$ are as followed:

$$\begin{aligned}
\mathrm{E}\left(\rho^2_{1,2}(x,X)\right) &= \text{trace}\left(\frac{\Sigma}{\text{trace}(\Sigma)}\right) = \frac{\text{trace}(\Sigma)}{\text{trace}(\Sigma)} = 1 \\
\mathrm{Var}\left(\rho^2_{1,2}(x,X)\right) &= 2\text{trace}\left(\frac{\Sigma^2}{\text{trace}(\Sigma)^2}\right) = \frac{2\text{trace}(\Sigma^2)}{\text{trace}(\Sigma)^2} = \frac{2\sum_{i=1}^d \lambda_i^2}{(\sum_{i=1}^d \lambda_i)^2} \\
\mathrm{Skew}\left(\rho^2_{1,2}(x,X)\right) &= 2\sqrt{2}\text{trace}\left(\frac{\Sigma^3}{\text{trace}(\Sigma)^3}\right)\left\{\text{trace}\left(\frac{\Sigma^2}{\text{trace}(\Sigma)^2}\right)\right\}^{-3/2} = \frac{2\sqrt{2}\sum_{i=1}^d \lambda_i^3}{\left(\sum_{i=1}^d \lambda_i^2\right)^{3/2}} \\
\mathrm{Kurt}\left(\rho^2_{1,2}(x,X)\right) &= 12\text{trace}\left(\frac{\Sigma^4}{\text{trace}(\Sigma)^4}\right)\left\{\text{trace}\left(\frac{\Sigma^2}{\text{trace}(\Sigma)^2}\right)\right\}^{-2} = \frac{12\sum_{i=1}^d \lambda_i^4}{\left(\sum_{i=1}^d \lambda_i^2\right)^2}
\end{aligned} \tag{A.2}$$

A.4 Moments of the squared Mahalanobis distances

For a full-rank d-dimensional dataset, choosing $\delta = 2$ and $k = d$ in the simplicial distances produces values equal to the squared Mahalanobis distance divided by d. As such, this section provides details of the moments of the squared Mahalanobis distance divided by d. The distribution of the Mahalanobis distance is dependent on whether population statistics are used, or sample estimates of the statistics are used.

Using true Σ and μ

Let X be a d-dimensional dataset with population mean μ and population covariance matrix Σ. The Mahalanobis distances from all points $x \in X$ to the population mean μ are known to follow a chi-square distribution with d degrees of freedom [168]. When the distances are divided by the scalar d for compatibility with the simplicial distances, the distribution of the distances is:

$$\rho^2_{d,2}(x, X) \sim \frac{1}{d}\chi^2.$$

This produces the following moments:

$$\mathrm{E}\left(\rho^2_{d,2}(x,X)\right) = \frac{d}{d} = 1$$

$$\mathrm{Var}\left(\rho^2_{d,2}(x,X)\right) = \frac{2d}{d} = \frac{2}{d}$$

$$\mathrm{Skew}\left(\rho^2_{d,2}(x,X)\right) = \frac{1}{d}\sqrt{\frac{8}{d}} = \sqrt{\frac{8}{d^3}}$$

$$\mathrm{Kurt}\left(\rho^2_{d,2}(x,X)\right) = \frac{1}{d}\frac{12}{d} = \frac{12}{d^2}$$

Using sample $\tilde{\Sigma}$ and $\tilde{\mu}$

Consider now the sample mean $\tilde{\mu}$ and sample covariance matrix $\tilde{\Sigma}$ of the d-dimensional dataset X with N observations. Note that the biased sample covariance matrix is used, defined $\tilde{\Sigma} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \tilde{\mu})(x_i - \tilde{\mu})^\top$, as in (3.1). Define the squared Mahalanobis distance with these sample parameters as $d \times \rho^2_{d,2}(x,X)$, as the simplicial distance with $k = d$, $\delta = 2$ is equal to the squared Mahalanobis distance divided by d. This distance follows a scaled Beta distribution with parameters $\alpha = d/2$, $\beta = (N-d-1)/2$ [88, 240]:

$$\frac{d}{(N-1)}\rho^2_{d,2}(x,X) \sim Beta\left(\frac{d}{2}, \frac{N-d-1}{2}\right)$$

If the squared Mahalanobis distance is divided by d for compatibility with the simplicial distances, the following moments are produced using the sample mean and sample covariance matrix:

$$\mathrm{E}\left(\rho^2_{d,2}(x,X)\right) = \frac{N-1}{d}\frac{\alpha}{\alpha+\beta} = \frac{N-1}{d}\frac{d}{2}\frac{1}{\frac{d}{2}+\frac{N-d-1}{2}} = 1$$

$$\mathrm{Var}\left(\rho^2_{d,2}(x,X)\right) = \frac{(N-1)^2}{d^2}\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} = \frac{2(N-d-1)}{d(N+1)}$$

$$\mathrm{Skew}\left(\rho^2_{d,2}(x,X)\right) = \frac{2(\beta-\alpha)\sqrt{\alpha+\beta+1}}{(\alpha+\beta+2)\sqrt{\alpha\beta}} = \frac{2\sqrt{2}(N-2d-1)\sqrt{N+1}}{(N+3)\sqrt{d(N-d-1)}}$$

$$\mathrm{Kurt}\left(\rho^2_{d,2}(x,X)\right) = \frac{6\left[(\alpha-\beta)^2(\alpha+\beta+1) - \alpha\beta(\alpha+\beta+2)\right]}{\alpha\beta(\alpha+\beta+2)(\alpha+\beta+3)}$$

$$= \frac{12\left[(N+1)(2d-N+1)^2 - d(N+3)(N-d-1)\right]}{d(N-d-1)(N+3)(N+5)}$$

A.5 Moments of the minimal-variance distances

The minimal-variance distances with parameter k are found using the quadratic form $\rho^2_{A_k}(x,X) = (x-\mu)^\top A_k (x-\mu)$, where $A_k = \sum_{i=0}^{k-1} \theta_i \Sigma^i$. Following the equations given in Section A.1, the expectation of the minimal-variance distances with parameter k is given by

$$\mathrm{E}\left(\rho^2_{A_k}(x,X)\right) = \sum_{i=0}^{k-1} \theta_i \mathrm{trace}\left(\Sigma^{i+1}\right) = \theta_\alpha^\top S_{(1,k)},$$

where the vectors θ_α and $S_{(1,k)}$ are as defined in Section 4.2. Let V be the Vandermonde matrix defined by Equation (4.4). Then, by the formulae in Section A.1, as well as the justification in Section 4.2,

$$\mathrm{Var}\left(\rho^2_{A_k}(x,X)\right) = \sum_{i=0}^{k-1}\sum_{j=0}^{k-1} \theta_i\theta_j \mathrm{trace}\left(\Sigma^{i+j+1}\right) = \theta_\alpha^\top V^\top V \theta_\alpha.$$

The skewness and kurtosis of the minimal-variance distances are respectively given as:

$$\begin{aligned}\mathrm{Skew}\left(\rho^2_{A_k}(x,X)\right) &= \frac{2\sqrt{2}\sum_{i=0}^{k-1}\sum_{j=0}^{k-1}\sum_{l=0}^{k-1}\theta_i\theta_j\theta_k \mathrm{trace}\left(\Sigma^{i+j+l+1}\right)}{\left(\sum_{i=0}^{k-1}\sum_{j=0}^{k-1}\theta_i\theta_j \mathrm{trace}\left(\Sigma^{i+j+1}\right)\right)^{3/2}} \\ &= \frac{2\sqrt{2}\sum_{i=0}^{k-1}\sum_{j=0}^{k-1}\sum_{l=0}^{k-1}\theta_i\theta_j\theta_k \mathrm{trace}\left(\Sigma^{i+j+l+1}\right)}{\left(\theta_\alpha^\top V^\top V \theta_\alpha\right)^{3/2}},\end{aligned}$$

$$\begin{aligned}\mathrm{Kurt}\left(\rho^2_{A_k}(x,X)\right) &= \frac{12\sum_{i=0}^{k-1}\sum_{j=0}^{k-1}\sum_{l=0}^{k-1}\sum_{m=0}^{k-1}\theta_i\theta_j\theta_l\theta_m \mathrm{trace}\left(\Sigma^{i+j+l+m+1}\right)}{\left(\sum_{i=0}^{k-1}\sum_{j=0}^{k-1}\theta_i\theta_j \mathrm{trace}\left(\Sigma^{i+j+1}\right)\right)^{2}} \\ &= \frac{12\sum_{i=0}^{k-1}\sum_{j=0}^{k-1}\sum_{l=0}^{k-1}\sum_{m=0}^{k-1}\theta_i\theta_j\theta_l\theta_m \mathrm{trace}\left(\Sigma^{i+j+l+m+1}\right)}{\left(\theta_\alpha^\top V^\top V \theta_\alpha\right)^{2}}.\end{aligned}$$

Appendix B

Alternative Minimal-Variance Whitening Polynomials

B.1 Minimal-variance whitening constraint with $\alpha = 1$

In Section 5.2.2, the choice of α in the constraint $\theta_\alpha^\top S_{(\alpha,k)} = S_{\alpha-1/2}$ is discussed. Although using $\alpha = 1$ is beneficial in the sense that polynomials do not require adjustment in rank-deficient cases (see Section 5.2.3), using $\alpha = 1$ has some downfalls which prevent it from being the default parameter choice. It is shown in Section 5.2.2 that a high value of k is needed to achieve good results from the polynomial with $\alpha = 1$. When k is not high, the coefficient vector will often be of the form $\theta_1 = \left(\frac{S_{1/2}}{S_1}, 0, \ldots, 0\right)^\top$, which makes the polynomial equal to a scaled version of the identity matrix.

As an example, consider the minimal-variance whitening polynomial with $\alpha = 1$ and $k = 2$. Using these parameters, the equation for the coefficient vector θ_α is

$$\theta_1 = \frac{S_{1/2}}{S_{(1,2)}^\top M_{(2)}^{-1} S_{(1,2)}} M_{(2)}^{-1} S_{(1,2)}. \tag{B.1}$$

Considering only $M_{(2)}^{-1}S_{(1,2)}$:

$$\begin{aligned} M_{(2)}^{-1}S_{(1,2)} &= \frac{1}{S_1S_3 - S_2^2}\begin{pmatrix} S_3 & -S_2 \\ -S_2 & S_1 \end{pmatrix}\begin{pmatrix} S_1 \\ S_2 \end{pmatrix} \\ &= \frac{1}{S_1S_3 - S_2^2}\begin{pmatrix} S_1S_3 - S_2^2 \\ -S_1S_2 + S_1S_2 \end{pmatrix} \\ &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{aligned}$$

The denominator of the fraction in Equation (B.1) is then

$$S_{(1,2)}^{\top}M_{(2)}^{-1}S_{(1,2)} = \begin{pmatrix} S_1 & S_2 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = S_1,$$

which makes the fraction in Equation (B.1) equal $S_{1/2}/S_1$. The coefficient vector for the minimal-variance whitening polynomial is therefore

$$\theta_1 = \begin{pmatrix} S_{1/2}/S_1 & 0 \end{pmatrix}^{\top},$$

producing the matrix $A_2 = \frac{S_{1/2}}{S_1}I$. It can be shown using the same method that this holds for other low values of k too.

B.2 Alternative minimal-variance polynomial methods

In Section 5.6, three alternative polynomials are derived for use in the construction of minimal-variance whitening. The coefficient vectors of these alternative polynomials are given in Table 5.17, along with the different definitions needed for the matrix $M_{(k)}$ in each polynomial method. Here, the derivations of these coefficient vectors are given. For all of these polynomials, the constraint (5.2) is used with $\alpha = 1/2$.

Polynomial $A_k = \sum_{i=0}^{k-1}\theta_i\Sigma^i$

This is the original polynomial used throughout the minimal-variance whitening methods of this thesis. The details of this method are given in Theorem 4 in Section 5.2.1, but here the steps are broken down further.

Begin by considering $k = 2$. Consider the function to be minimize:

$$\begin{aligned}
\operatorname{trace}\left(\mathcal{D}(X_{A_2})\right) &= \operatorname{trace}\left(A_2^\top \Sigma^{1/2} A_2\right) \\
&= \operatorname{trace}\left(\sum_{i=0}^{1} \theta_i \Sigma^i \Sigma^{1/2} \sum_{j=0}^{1} \theta_j \Sigma^j\right) \\
&= \operatorname{trace}\left(\theta_0^2 \Sigma + 2\theta_0\theta_1 \Sigma^2 + \theta_1^2 \Sigma^3\right) \\
&= \theta_0^2 S_1 + 2\theta_0\theta_1 S_2 + \theta_1^2 S_3 \\
&= \begin{pmatrix} \theta_0 & \theta_1 \end{pmatrix} \begin{pmatrix} S_1 & S_2 \\ S_2 & S_3 \end{pmatrix} \begin{pmatrix} \theta_0 \\ \theta_1 \end{pmatrix} \\
&= \theta_{1/2}^T M_{(2)} \theta_{1/2},
\end{aligned}$$

where $\theta_{1/2}^\top = (\theta_0, \theta_1)$ and $M_{(2)} = \begin{pmatrix} S_1 & S_2 \\ S_2 & S_3 \end{pmatrix}$. For general values of k in the polynomial A_k, the coefficient vector is $\theta_{1/2}^\top = (\theta_0, \theta_1, \ldots, \theta_{k-1})$ and

$$M_{(k)} = \begin{pmatrix} S_1 & S_2 & \cdots & S_k \\ S_2 & S_3 & \cdots & S_{k+1} \\ \vdots & \vdots & \ddots & \vdots \\ S_k & S_{k+1} & \cdots & S_{2k-1} \end{pmatrix}.$$

The constraint with $\alpha = 1/2$ is $\operatorname{trace}\left(A_k \Sigma^{1/2}\right) = \operatorname{trace}\left(\Sigma^0\right)$. The left side of this constraint can be rewritten as follows:

$$\begin{aligned}
\operatorname{trace}\left(A_k \Sigma^{1/2}\right) &= \operatorname{trace}\left(\sum_{i=0}^{k-1} \theta_i \Sigma^i \Sigma^{1/2}\right) \\
&= \sum_{i=0}^{k-1} \theta_i S_{i+1/2} \\
&= (\theta_0, \theta_1, \ldots, \theta_{k-1}) (S_{1/2}, S_{3/2}, \ldots, S_{1/2+k-1})^\top \\
&= \theta_{1/2}^\top S_{\left(\frac{1}{2}, k, 1\right)},
\end{aligned}$$

recalling from (5.18) that $S_{(i,k,\gamma)} = \left(S_i, S_{i+\gamma}, S_{i+2\gamma}, \ldots, S_{i+k-\gamma}\right)$.

Thus, a value of $\theta_{1/2}$ that minimizes $\theta_{1/2}^T M_{(k)} \theta_{1/2}$ while the constraint $\theta_{1/2}^\top S_{\left(\frac{1}{2}, k, 1\right)} = S_0$ holds is sought after. Theorem 4 details how to find this value of θ_α using the Lagrange function for general values of α.

Polynomial $B_k = \sum_{i=0}^{k-1} \theta_i \Sigma^{i/2}$

This polynomial takes advantage of the square root of the covariance matrix, given that the inverse of the square root is the matrix being approximated. To begin, consider $k = 2$.

$$\begin{aligned}
\operatorname{trace}\left(\mathcal{D}(X_{B_2})\right) &= \operatorname{trace}\left(B_2^\top \Sigma^{1/2} B_2\right) \\
&= \operatorname{trace}\left(\sum_{i=0}^{1} \theta_i \Sigma^{i/2} \Sigma^{1/2} \sum_{j=0}^{1} \theta_j \Sigma^{j/2}\right) \\
&= \operatorname{trace}\left(\theta_0^2 \Sigma + 2\theta_0\theta_1 \Sigma^{3/2} + \theta_1^2 \Sigma^2\right) \\
&= \theta_0^2 S_1 + 2\theta_0\theta_1 S_{3/2} + \theta_1^2 S_2 \\
&= \begin{pmatrix} \theta_0 & \theta_1 \end{pmatrix} \begin{pmatrix} S_1 & S_{3/2} \\ S_{3/2} & S_2 \end{pmatrix} \begin{pmatrix} \theta_0 \\ \theta_1 \end{pmatrix} \\
&= \theta_{1/2}^\top M_{(2)} \theta_{1/2},
\end{aligned}$$

where $\theta_{1/2}^\top = (\theta_0, \theta_1)$ and $M_{(2)} = \begin{pmatrix} S_1 & S_{3/2} \\ S_{3/2} & S_2 \end{pmatrix}$. For general k in polynomial B_k, the coefficient vector is $\theta_{1/2}^\top = (\theta_0, \theta_1, \ldots, \theta_{k-1})$ and

$$M_{(k)} = \begin{pmatrix} S_1 & S_{3/2} & \cdots & S_{(k+1)/2} \\ S_{3/2} & S_2 & \cdots & S_{(k+2)/2} \\ \vdots & \vdots & \ddots & \vdots \\ S_{(k+1)/2} & S_{(k+2)/2} & \cdots & S_k \end{pmatrix}.$$

The left side of the constraint $\operatorname{trace}\left(B_k \Sigma^{1/2}\right) = \operatorname{trace}\left(\Sigma^0\right)$ can be written as:

$$\begin{aligned}
\operatorname{trace}\left(B_k \Sigma^{1/2}\right) &= \operatorname{trace}\left(\sum_{i=0}^{k-1} \theta_i \Sigma^{i/2} \Sigma^{1/2}\right) \\
&= \sum_{i=0}^{k-1} \theta_i S_{(i+1)/2} \\
&= (\theta_0, \theta_1, \ldots, \theta_{k-1}) (S_{1/2}, S_1, \ldots, S_{k/2})^\top \\
&= \theta_{1/2}^\top S_{(\frac{1}{2}, \frac{k}{2}, \frac{1}{2})}.
\end{aligned}$$

The optimal value of $\theta_{1/2}$ is therefore the one that minimizes $\theta_{1/2}^\top M_{(k)} \theta_{1/2}$, while the constraint $\theta_{1/2}^\top S_{(\frac{1}{2}, \frac{k}{2}, \frac{1}{2})} = S_0$ holds. The same steps as those used for polynomial A in Theorem 4 (using the Lagrangian) are followed to find the values of the coefficient vector $\theta_{1/2}$, given in Table 5.17.

Polynomial $C_k = \sum_{i=1}^{k-1} \theta_i \Sigma^i$

This polynomial aims to be a non-regularized version of polynomial A_k. Whereas the previous polynomials have k coefficients, this polynomial has $k-1$ thanks to the summation index starting at 1. Therefore, the empirical calculation using $k = 3$ is considered for this polynomial.

$$\begin{aligned}
\operatorname{trace}\left(\mathcal{D}(X_{C_3})\right) &= \operatorname{trace}\left(C_3^\top \Sigma^{1/2} C_3\right) \\
&= \operatorname{trace}\left(\sum_{i=1}^{2} \theta_i \Sigma^i \Sigma^{1/2} \sum_{j=1}^{2} \theta_j \Sigma^j\right) \\
&= \operatorname{trace}\left(\theta_1^2 \Sigma^3 + 2\theta_1\theta_2 \Sigma^4 + \theta_2^2 \Sigma^5\right) \\
&= \theta_1^2 S_3 + 2\theta_1\theta_2 S_4 + \theta_2^2 S_5 \\
&= \begin{pmatrix} \theta_1 & \theta_2 \end{pmatrix} \begin{pmatrix} S_3 & S_4 \\ S_4 & S_5 \end{pmatrix} \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} \\
&= \theta_{1/2}^\top M_{(3)} \theta_{1/2},
\end{aligned}$$

where $\theta_{1/2}^\top = (\theta_1, \theta_2)$ and $M_{(3)} = \begin{pmatrix} S_3 & S_4 \\ S_4 & S_5 \end{pmatrix}$. For general k in polynomial C_k, the coefficient vector is $\theta_{1/2}^\top = (\theta_1, \theta_2, \ldots, \theta_{k-1})$ and

$$M_{(k)} = \begin{pmatrix} S_3 & S_4 & \cdots & S_{k+1} \\ S_4 & S_5 & \cdots & S_{k+2} \\ \vdots & \vdots & \ddots & \vdots \\ S_{k+1} & S_{k+2} & \cdots & S_{2k-1} \end{pmatrix}.$$

The left side of the constraint $\operatorname{trace}\left(C_k \Sigma^{1/2}\right) = \operatorname{trace}\left(\Sigma^0\right)$ can be written as:

$$\begin{aligned}
\operatorname{trace}\left(C_k \Sigma^{1/2}\right) &= \operatorname{trace}\left(\sum_{i=1}^{k-1} \theta_i \Sigma^i \Sigma^{1/2}\right) \\
&= \sum_{i=1}^{k-1} \theta_i S_{i+1/2} \\
&= (\theta_1, \theta_2, \ldots, \theta_{k-1})^\top \left(S_{3/2}, S_{5/2}, \ldots, S_{(k-1)/2}\right) \\
&= \theta_{1/2}^\top S_{\left(\frac{3}{2}, k-1, 1\right)}.
\end{aligned}$$

Then following the steps of Theorem 4 finds the coefficient vector θ which minimizes $\theta_{1/2}^\top M_{(k)} \theta_{1/2}$, while $\theta_{1/2}^\top S_{\left(\frac{3}{2}, k-1, 1\right)} = S_0$ holds. The formula of the coefficient vector θ is given in Table 5.17.

Polynomial $D_k = \sum_{i=1}^{k-1} \theta_i \Sigma^{i/2}$

This is the non-regularized polynomial that makes use of the square root of the covariance matrix. Like the previous polynomial, D_k requires $k-1$ coefficients, so $k=3$ is used in the following example.

$$\begin{aligned}
\operatorname{trace}\left(\mathcal{D}(X_{D_3})\right) &= \operatorname{trace}\left(D_3^\top \Sigma^{1/2} D_3\right) \\
&= \operatorname{trace}\left(\sum_{i=1}^{2} \theta_i \Sigma^{i/2} \Sigma^{1/2} \sum_{j=1}^{2} \theta_j \Sigma^{j/2}\right) \\
&= \operatorname{trace}\left(\theta_1^2 \Sigma^2 + 2\theta_1\theta_2 \Sigma^{5/2} + \theta_2^2 \Sigma^3\right) \\
&= \theta_1^2 S_2 + 2\theta_1\theta_2 S_{5/2} + \theta_2^2 S_3 \\
&= \begin{pmatrix} \theta_1 & \theta_2 \end{pmatrix} \begin{pmatrix} S_2 & S_{5/2} \\ S_{5/2} & S_3 \end{pmatrix} \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} \\
&= \theta_{1/2} M_{(3)} \theta_{1/2},
\end{aligned}$$

where $\theta_{1/2}^\top = (\theta_0, \theta_1)$ and $M_{(3)} = \begin{pmatrix} S_2 & S_{5/2} \\ S_{5/2} & S_3 \end{pmatrix}$. For general k in polynomial D_k, the coefficient vector is $\theta_{1/2}^\top = (\theta_1, \theta_2, \ldots, \theta_{k-1})$ and

$$M_{(k)} = \begin{pmatrix} S_2 & S_{5/2} & \cdots & S_{(k+2)/2} \\ S_{5/2} & S_3 & \cdots & S_{(k+3)/2} \\ \vdots & \vdots & \ddots & \vdots \\ S_{(k+2)/2} & S_{(k+3)/2} & \cdots & S_k \end{pmatrix}.$$

The left side of the constraint $\operatorname{trace}\left(D_k \Sigma^{1/2}\right) = \operatorname{trace}\left(\Sigma^0\right)$ can be written as:

$$\begin{aligned}
\operatorname{trace}\left(D_k \Sigma^{1/2}\right) &= \operatorname{trace}\left(\sum_{i=1}^{k-1} \theta_i \Sigma^{i/2} \Sigma^{1/2}\right) \\
&= \sum_{i=1}^{k-1} \theta_i S_{(i+1)/2} \\
&= (\theta_1, \theta_2, \ldots, \theta_{k-1})^\top (S_1, S_{3/2}, \ldots, S_{k/2}) \\
&= \theta_{1/2}^\top S_{\left(1, \frac{k-1}{2}, \frac{1}{2}\right)}.
\end{aligned}$$

As with the previous polynomials, follow the method using in Theorem 4 to compute the value of the vector $\theta_{1/2}$ which minimizes $\theta_{1/2}^\top M_{(k)} \theta_{1/2}$ while $\operatorname{trace}\left(D_k \Sigma^{1/2}\right) = \operatorname{trace}\left(\Sigma^0\right)$ holds.

Appendix C

Details of Datasets

C.1 Rotating a matrix

When producing synthetic datasets, it is often useful to rotate the covariance matrix before using it to create a multivariate normal distribution, to ensure there are correlations in the data as desired. A rotation matrix is defined to be a square orthonormal matrix with determinant equal to ± 1 [77]. To find such a matrix, a random matrix of size $d \times d$ is generated, and the QR decomposition of this matrix is found. The matrix to be rotated is then pre- and post-multiplied by the rotation matrix, as seen in Snippet C.1.

```
import numpy as np
from scipy.linalg import qr

def rotation_matrix(d):
    """Produce a rotation matrix of size d by d, using the QR
    decomposition"""
    M = np.random.rand(d, d)
    Q, _ = qr(M)
    return Q

#let unrotated_matrix be a pre-defined d by d matrix to be
rotated
d = unrotated_matrix.shape[0]
Q = rotation_matrix(d)
rotated_matrix = Q.T @ unrotated_matrix @ Q
```

Snippet C.1: Producing a rotation matrix and using it to rotate a matrix.

C.2 Datasets used in Chapter 4

The datasets used in Section 4.5.1 and Section 4.5.2 are detailed in Table C.1 and Table C.2, respectively.

Dataset	Original eigenvalues	Normalized eigenvalues
Iris	4.2, 0.24, 0.08, 0.02	0.0431 0.0012 0.0004 0.0001
Wine	99201.8, 172.5, 9.4, 5.0, 1.2, 0.84, 0.28, 0.15, 0.11, 0.07, 0.04, 0.02, 0.008	3.25e-03, 4.47e-05, 1.51e-05, 5.36e-06, 4.24e-06, 3.06e-06, 2.10e-06, 3.98e-07, 2.96e-07, 2.16e-07, 1.05e-07, 7.55e-08, 2.87e-08
Image Seg.	11393.8, 9183.6, 5479.5, 2300.1, 217.2, 161.7, 55.7, 14.4, 3.6, 1.2, 0.2, 0.02, 0.001, 0.001, 0, 0, 0, 0, 0	0.135, 0.0869, 0.0178, 0.00683, 0.0034, 0.00155, 0.00146, 0.000351, 0.000105, 6.21e-05, 2.97e-05, 3.78e-06, 5.03e-07, 2.37e-08, 0, 0, 0, 0, 0
Digits	179.0, 163.7, 141.8, 101.1, 69.5, 59.1, 51.9, 44.0, 40.3, 37.0, 28.5, 27.3, 21.9, 21.3, 17.6, 16.9, 15.9, 15.0, 12.2, 10.9, 10.7, 9.6, 9.2, 8.7, 8.4, 7.2, 6.9, 6.2, 5.9, 5.2, 4.5, 4.2, 4.0, 3.9, 3.7, 3.5, 3.1, 2.7, 2.7, 2.5, 2.3, 1.9, 1.8, 1.7, 1.4, 1.3, 1.3, 0.93, 0.67, 0.49, 0.25, 0.09, 0.06, 0.06, 0.04, 0.02, 0.008, 0.004, 0.001, 0.001, 0, 0, 0, 0	0.0473, 0.044, 0.037, 0.0266, 0.0184, 0.0153, 0.0134, 0.0115, 0.0107, 0.00796, 0.00741, 0.00678, 0.0058, 0.00514, 0.00452, 0.00437, 0.00407, 0.00333, 0.0032, 0.00286, 0.00258, 0.00248, 0.00234, 0.00225, 0.00195, 0.00184, 0.00168, 0.00159, 0.00139, 0.00121, 0.00116, 0.0011, 0.00106, 0.000988, 0.000951, 0.000844, 0.000809, 0.000732, 0.00071, 0.000649, 0.000608, 0.000499, 0.00047, 0.000445, 0.000367, 0.00034, 0.000305, 0.000239, 0.000171, 0.000122, 6.35e-05, 2.59e-05, 1.6e-05, 1.49e-05, 9.08e-06, 3.55e-06, 1.99e-06, 1.01e-06, 3.15e-07, 1.63e-07, 1.04e-07, 0, 0, 0
Protein	2.59, 1.44, 0.604, 0.223, 0.136, 0.107, 0.074, 0.0661, 0.05, 0.0412, 0.0302, 0.0261, 0.023, 0.0187, 0.0136, 0.0108, 0.00858, 0.00818, 0.00549, 0.00427, 0.0041, 0.00386, 0.00327, 0.00301, 0.00259, 0.00247, 0.00173, 0.00159, 0.00131, 0.00114, 0.00105, 0.000993, 0.000869, 0.000761, 0.000726, 0.000682, 0.0006, 0.000527, 0.000468, 0.000419, 0.000397, 0.00029, 0.000283, 0.00025, 0.000239, 0.000209, 0.000202, 0.000182, 0.000168, 0.000159, 0.000143, 0.000133, 0.000127, 0.000118, 0.000109, 0.000106, 9.27e-05, 8.78e-05, 8.6e-05, 7.72e-05, 7.42e-05, 6.57e-05, 6.37e-05, 5.81e-05, 5.73e-05, 5.41e-05, 5.11e-05, 4.64e-05, 4.22e-05, 3.92e-05, 3.56e-05, 3.35e-05, 2.97e-05, 2.59e-05, 2.15e-05, 1.74e-05, 0	0.0169, 0.00896, 0.00488, 0.00269, 0.00149, 0.00124, 0.00084, 0.00069, 0.000541, 0.000514, 0.00032, 0.000285, 0.000232, 0.000171, 0.000142, 0.000116, 0.00011, 9.47e-05, 6.49e-05, 6.17e-05, 5.42e-05, 4.89e-05, 4.67e-05, 3.56e-05, 3.26e-05, 2.78e-05, 2.02e-05, 1.79e-05, 1.65e-05, 1.37e-05, 1.32e-05, 1.25e-05, 1.1e-05, 1.08e-05, 9.03e-06, 8.59e-06, 7.64e-06, 7.29e-06, 6.03e-06, 5.6e-06, 5.09e-06, 4.12e-06, 3.55e-06, 3.47e-06, 3.2e-06, 2.85e-06, 2.56e-06, 2.37e-06, 2.2e-06, 2.05e-06, 1.91e-06, 1.82e-06, 1.63e-06, 1.57e-06, 1.45e-06, 1.41e-06, 1.25e-06, 1.19e-06, 1.13e-06, 1.06e-06, 9.65e-07, 9.4e-07, 8.73e-07, 7.73e-07, 7.34e-07, 7.22e-07, 6.84e-07, 6.21e-07, 5.48e-07, 5.23e-07, 4.97e-07, 4.47e-07, 3.9e-07, 3.48e-07, 2.85e-07, 2.4e-07, 0

Table C.1: Eigenvalues of the datasets given in Table 4.11, used in the K-Means clustering examples in Section 4.5.1. Eigenvalues of the raw and normalized datasets are given.

Dataset	d	N	#outliers (%)
Lympho	18	148	6 (4.1%)
WBC	30	278	21 (5.6%)
Glass	9	214	9 (4.2%)
Vowels	12	1456	50 (3.4%)
Cardio	21	1831	176 (9.6%)
Thyroid	6	3772	93 (2.5%)
Musk	166	3062	97 (3.2%)
Satimage-2	36	5803	71 (1.2%)
Letter	32	1600	100 (6.25%)
Speech	400	3686	61 (1.65%)
Pima	8	768	268 (35%)
Satellite	36	6435	2036 (32%)
Shuttle	9	49097	3511 (7%)
BreastW	9	683	239 (35%)
Arrhythmia	274	452	66 (15%)
Ionosphere	33	351	126 (36%)
MNIST	100	7603	700 (9.2%)
Optdigits	64	5216	150 (3%)
ForestCover	10	286048	2747 (0.9%)
Mammography	6	11183	260 (2.32%)
Annthyroid	6	7200	534 (7.42%)
Pendigits	16	6870	156 (2.27%)
Wine	13	129	10 (7.7%)

Table C.2: Details of the datasets used in the outlier labelling example in Section 4.5.2, given in Table 4.15.

C.3 Datasets used in Chapter 5

Datasets used in Section 5.2.2

The purpose of the middle set of eigenvalues in each dataset is to create a slow taper towards the zero eigenvalue(s). The eigenvalues used to generate the datasets in Section 5.2.2 (specifically, Figure 5.2 and Figure 5.3) are given below, in Python notation. The data is formed using a multivariate Gaussian distribution (the Python function `numpy.random.multivariate_normal`), with zero mean and $5 \times d$ observations. The covariance matrices used to generate the data have the following values on the diagonal, and all other entries zero.

d	Eigenvalues
10	`[5, 4, 3, 2, 1] + [numpy.random.rand() ** i for i in range(4)] + [0]`
50	`[5, 4, 3, 2, 1] + [numpy.random.rand() ** i for i in range(30)] + [0] * 15`
150	`[5, 4, 3, 2, 1] + [numpy.random.rand() ** (i/2) for i in range(100)] + [0] * 45`

Table C.3: Eigenvalues used to generate the three datasets used in examples in Section 5.2.2. The eigenvalues are given in Python notation.

Datasets used in Section 5.2.3

The datasets used in Figure 5.4 are the same as those given in Table C.3, for $d = 50$ and $d = 150$.

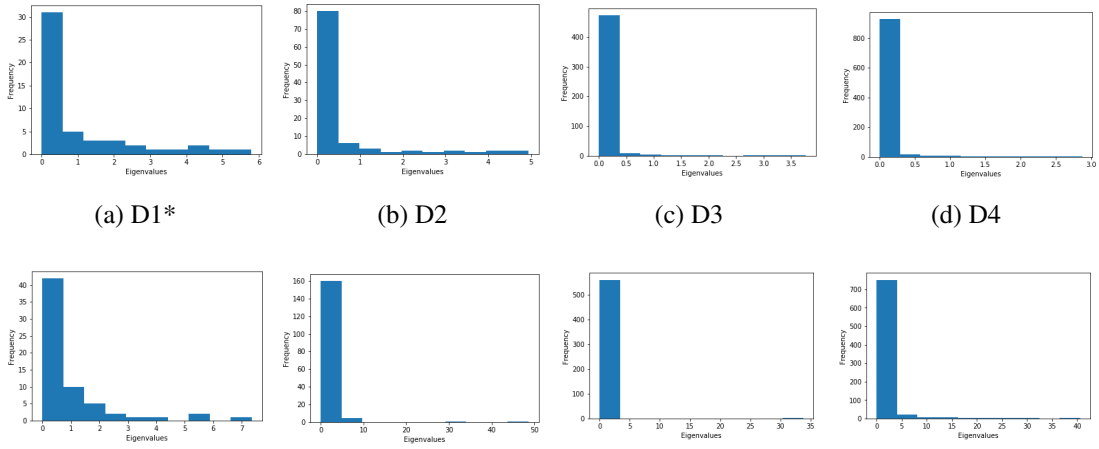
The datasets used in Table 5.1 were generated using the following Python code:

```
true_sigma = numpy.diag([numpy.random.rand() for _ in range(R)] +
[0] * (d - R))
X = numpy.random.multivariate_normal(np.zeros(d), true_sigma, N).T
```

The empirical covariance matrix can then be found using the `numpy.cov` function.

Datasets used in Section 5.3.1, $d < N$

The histograms in Figure C.2 give the distributions of the eigenvalues of the datasets with $d < N$ used in Section 5.3.1. Summary information about the datasets is given in Table 5.2.



(a) D1* (b) D2 (c) D3 (d) D4

(e) Digits* (f) Musk* (g) HAR (h) MNIST*

Figure C.2: Eigenvalues of the datasets used in Section 5.3.1 with $d < N$. Datasets marked with * in the caption have been rescaled such that each variable has zero mean and unit variance, and the eigenvalues are taken after this rescaling.

Computation time of examples in Section 5.3.1, $d < N$

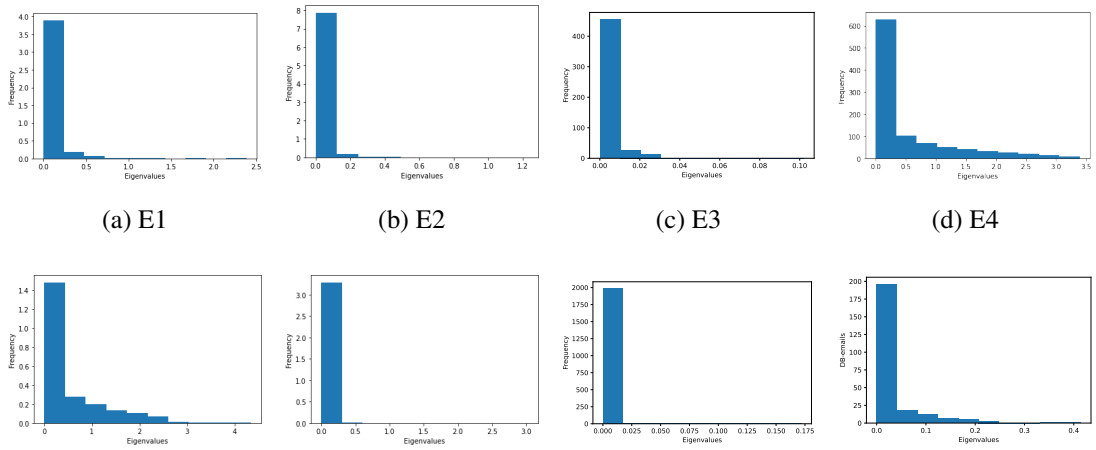
Table C.4 gives the average time taken to calculate the minimal-variance polynomial (in seconds) over 100 runs, for each dataset used in Section 5.3.1 for data with $d < N$, for each different value of k.

Dataset	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 7$	$k = 8$	$k = 9$	$k = 10$
D1	0.04	0.04	0.05	0.04	0.04	0.06	0.04	0.05
D2	0.25	0.23	0.23	0.21	0.20	0.23	0.26	0.27
D3	1.35	1.43	1.72	2.08	1.98	2.00	2.59	3.11
D4	4.16	4.73	5.91	6.62	8.47	9.77	12.02	14.36
Digits	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
Musk	0.21	0.22	0.22	0.23	0.25	0.26	0.29	0.31
HAR	1.98	2.15	2.34	2.62	3.81	3.54	3.83	4.42
MNIST	6.24	6.55	7.24	8.20	10.11	12.29	12.71	14.35

Table C.4: Time taken to calculate A_k in seconds for each dataset in Section 5.3.1 with $d < N$ (average over 100 runs).

Datasets used in Section 5.3.1, $d > N$

The histograms in Figure C.3 give the distributions of the eigenvalues of the datasets with $d > N$ used in Section 5.3.1. Summary information about the datasets is given in Table 5.5.



(a) E1 (b) E2 (c) E3 (d) E4

(e) Madelon† * (f) Yeast Gene* (g) Colon* (h) DB-emails

Figure C.3: Eigenvalues of the datasets used in Section 5.3.1 with $d > N$. Datasets marked with * in the caption have been rescaled such that each variable has zero mean and unit variance, and the eigenvalues are taken after this rescaling.

Datasets used in Section 5.3.2

In Section 5.3.2, different whitening methods are compared with the minimal-variance polynomial whitening method by applying them to the Iris dataset and the Wisconsin breast cancer dataset (the latter of which has been scaled to improve performance). The eigenvalues of these datasets are given below:

Eigenvalues of Iris: [4.2282, 0.2427, 0.0782, 0.0238]

Eigenvalues of Wisconsin Breast Cancer: [9.8005, 8.2868, 3.3664, 2.2588, 1.5496, 1.4151, 1.1688, 0.9771, 0.5900, 0.5073, 0.4427, 0.3733, 0.3303, 0.2486, 0.2024, 0.1211, 0.1064, 0.0798, 0.0737, 0.0519, 0.0452, 0.0369, 0.0302, 0.0250, 0.0226, 0.0186, 0.0144, 0.0125, 0.0058, 0.0026, 0.0010, 0.0004]

Producing correlated degenerate data in Section 5.6

In Snippet C.4, a method of finding a randomly generated degenerate and correlated covariance matrix with $d = 50$ is given. The last 20 dimensions are gradually multiplied by smaller values to force a gradual taper towards degeneracy, making the rank of the covariance matrix unclear. This method is used to find dataset 2 in Section 5.6.

```
import numpy as np

np.random.seed(0)
d = 50
sigma = np.random.rand(d, d)
for i in range(20):
    sigma[-(20-i):] *= 0.4**i
    sigma[:, -(20-i):] *= 0.4**i

sigma = sigma @ sigma.T
```

Snippet C.4: Code to produce the correlated degenerate covariance matrix used to generate dataset 2 in Section 5.6.

Appendix D

Clustering Metrics

D.1 Adjusted rand score

Let L_T be the vector of true labels, and let L_P be the labels assigned by the K-Means clustering. Define a as the number of pairs of points in the same set in L_T and in the same set in L_P, i.e. the number of points whose labels are the same in L_T and L_P. Define b as the number of pairs of points in different sets in L_T and in different sets in L_P, i.e. the number of points whose labels are different in L_T and L_P. The unadjusted rand score is given by

$$R = \frac{a+b}{\nu},$$

where ν is the total number of possible pairs in the dataset, without ordering. The unadjusted rand score does not account for the possibility that random label assignments can perform well, so the expected rand score $E[R]$ of random labelings is removed by defining the adjusted rand score as

$$AR = \frac{R - E[R]}{\max(R) - E[R]},$$

where the distribution of R is taken to be hypergeometric [113]. The adjusted rand score takes values in [-1, 1], where 1 indicates a perfect matching between L_T and L_P.

D.2 Purity score

The purity score is found as follows: let $T = \{t_1, t_2, \ldots, t_m\}$ be the set of 'true' clusters in the data, and let $P = \{p_1, p_2, \ldots, p_K\}$ be the set of predicted clusters. The purity score

measures the extent to which a predicted cluster p_i only contains points from a single 'true' cluster t_j:

$$\mathcal{P}(T, P) = \frac{1}{N} \sum_{i=1}^{K} \max_j |p_i \cap t_j|,$$

where N is the total number of points. That is, for each predicted cluster p_i, count the highest number of points from a single true cluster t_j predicted to be in p_i. These counts are summed and divided by the total number of observations. The purity score takes values in [0, 1], with 1 being a perfect clustering.

D.3 Silhouette score

The silhouette score is used to assess how well separated a set of clusters are. Let there be K clusters, denoted $C_1, C_2, \ldots, C_K$.

For a point $i \in C_I$, let $a(i)$ be the mean distance between i and all other points in the same cluster:

$$a(i) = \frac{1}{|C_I - 1|} \sum_{j \in C_I, i \neq j} d(i, j)$$

where d is the chosen distance measure (often chosen to be the Euclidean distance). The value $a(i)$ is a measure of how similar the point i is to all other points in its cluster (a small value indicates the point fits into the cluster well).

The mean dissimilarity of a point i to another cluster C_J, $J \neq I$, is defined as the mean distance from all points in i to all points in C_J. Define

$$b(i) = \min_{J \neq I} \sum_{j \in C_J} d(i, j)$$

to be the smallest mean distance from i to all points in any cluster other than C_J.

The silhouette value of one point i is then defined to be

$$s_i = \frac{b(i) - a(i)}{\max(a(i), b(i))}.$$

The silhouette score of the total clustering is then the mean of all silhouette values s_i. The silhouette score takes values between $[-1, 1]$, where a value of 1 indicates all values have been assigned to a cluster well. Values near 0 indicate overlapping clusters, and negative values indicate more points have been assigned to the incorrect cluster than the correct cluster.

Bibliography

[1] K. M. Abadir, W. Distaso, and F. Žikeš. Design-free estimation of variance matrices. *Journal of Econometrics*, 181(2):165–180, 2014.

[2] H. Abdi and L. J. Williams. Principal component analysis. *Wiley Interdisciplinary Reviews: computational statistics*, 2(4):433–459, 2010.

[3] H. A. Abu Alfeilat, A. B. Hassanat, O. Lasassmeh, A. S. Tarawneh, M. B. Alhasanat, H. S. Eyal Salman, and V. S. Prasath. Effects of distance measure choice on k-nearest neighbor classifier performance: a review. *Big Data*, 7(4):221–248, 2019.

[4] C. C. Aggarwal. An introduction to outlier analysis. In *Outlier Analysis*, pages 1–34. Springer, 2017.

[5] C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *International Conference on Database Theory*, pages 420–434. Springer, 2001.

[6] C. Agostinelli and L. Greco. Weighted likelihood estimation of multivariate location and scatter. *Test*, 28(3):756–784, 2019.

[7] M. Ahmed, R. Seraj, and S. M. S. Islam. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8):1295, 2020.

[8] J. Ahn. *High dimension, low sample size data analysis*. The University of North Carolina at Chapel Hill, 2006.

[9] J. Ahn, M. H. Lee, and J. A. Lee. Distance-based outlier detection for high dimension, low sample size data. *Journal of Applied Statistics*, 46(1):13–29, 2019.

[10] J. Akeret, A. Refregier, A. Amara, S. Seehars, and C. Hasner. Approximate

Bayesian computation for forward modeling in cosmology. *Journal of Cosmology and Astroparticle Physics*, 2015(08):043, 2015.

[11] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 96(12):6745–6750, 1999.

[12] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan. Machine learning in wireless sensor networks: Algorithms, strategies, and applications. *IEEE Communications Surveys & Tutorials*, 16(4):1996–2018, 2014.

[13] N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[14] K. Anaya-Izquierdo, F. Critchley, K. Vines, et al. Orthogonal simple component analysis: a new, exploratory approach. *The Annals of Applied Statistics*, 5(1):486–522, 2011.

[15] T. Anderson. *An introduction to multivariate statistics*. New York: Wiley, 2003.

[16] T. Ando and J. Bai. Clustering huge number of financial time series: A panel data approach with high-dimensional predictors and factor structures. *Journal of the American Statistical Association*, 112(519):1182–1198, 2017.

[17] F. Angiulli and C. Pizzuti. Fast outlier detection in high dimensional spaces. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 15–27. Springer, 2002.

[18] D. Anguita, A. Ghio, L. Oneto, et al. A public domain dataset for human activity recognition using smartphones. In *Esann*, volume 3, page 3, 2013.

[19] M. Aoshima and K. Yata. Two-sample tests for high-dimension, strongly spiked eigenvalue models. *Statistica Sinica*, pages 43–62, 2018.

[20] I. Assent. Clustering high dimensional data. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(4):340–350, 2012.

[21] H. Avron and S. Toledo. Randomized algorithms for estimating the trace of an

implicit symmetric positive semi-definite matrix. *Journal of the ACM (JACM)*, 58(2):1–34, 2011.

[22] J. Bai and S. Shi. Estimating high dimensional covariance matrices and its applications. *Annals of Economics and Finance*, 12(2):199–215, 2011.

[23] Z. Bai, J. Chen, and J. Yao. On estimation of the population spectral distribution from a high-dimensional sample covariance matrix. *Australian & New Zealand Journal of Statistics*, 52(4):423–437, 2010.

[24] E. Baktash, M. Karimi, and X. Wang. Covariance matrix estimation under degeneracy for complex elliptically symmetric distributions. *IEEE Transactions on Vehicular Technology*, 66(3):2474–2484, 2017.

[25] S. Banerjee and S. Ghosal. Posterior convergence rates for estimating large precision matrices using graphical models. *Electronic Journal of Statistics*, 8(2):2111–2137, 2014.

[26] S. Baraty, D. A. Simovici, and C. Zara. The impact of triangular inequality violations on medoid-based clustering. In *International Symposium on Methodologies for Intelligent Systems*, pages 280–289. Springer, 2011.

[27] C. Bartenhagen, H.-U. Klein, C. Ruckert, X. Jiang, and M. Dugas. Comparative study of unsupervised dimension reduction techniques for the visualization of microarray gene expression data. *BMC Bioinformatics*, 11(1):1–11, 2010.

[28] R. Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[29] A. Ben-Israel and T. N. Greville. *Generalized inverses: theory and applications*, volume 15. Springer Science & Business Media, 2003.

[30] Y. Bengio, O. Delalleau, and N. Le Roux. The curse of dimensionality for local kernel machines. *Technical Rep*, 1258:12, 2005.

[31] K. P. Bennett, U. Fayyad, and D. Geiger. Density-based indexing for approximate nearest-neighbor queries. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 233–243, 1999.

[32] T. Bernecker, M. E. Houle, H.-P. Kriegel, P. Kröger, M. Renz, E. Schubert, and

A. Zimek. Quality of similarity rankings in time series. In *International Symposium on Spatial and Temporal Databases*, pages 422–440. Springer, 2011.

[33] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful? In *International Conference on Database Theory*, pages 217–235. Springer, 1999.

[34] P. J. Bickel and E. Levina. Covariance regularization by thresholding. *The Annals of Statistics*, 36(6):2577–2604, 2008.

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

[36] E. Bingham and H. Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 245–250, 2001.

[37] G. Blom. Some properties of incomplete U-statistics. *Biometrika*, 63(3):573–580, 1976.

[38] A. Blum, J. Hopcroft, and R. Kannan. *Foundations of Data Science*. Cambridge University Press, 2014. doi:10.13140/2.1.5115.0726.

[39] C. Böhm, S. Berchtold, and D. A. Keim. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys (CSUR)*, 33(3):322–373, 2001.

[40] A.-R. Bologa, R. Bologa, A. Florea, et al. Big data and specific analysis methods for insurance fraud detection. *Database Systems Journal*, 4(4):30–39, 2013.

[41] R. Bott and R. Duffin. On the algebra of networks. *Transactions of the American Mathematical Society*, 74(1):99–109, 1953.

[42] R. G. Brereton and G. R. Lloyd. Re-evaluating the role of the Mahalanobis distance measure. *Journal of Chemometrics*, 30(4):134–143, 2016.

[43] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 93–104, 2000.

[44] P. Bühlmann and S. Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

[45] R. Bulirsch, J. Stoer, and J. Stoer. *Introduction to numerical analysis*, volume 3. Springer, 2002.

[46] T. Cai and W. Liu. Adaptive thresholding for sparse covariance matrix estimation. *Journal of the American Statistical Association*, 106(494):672–684, 2011.

[47] T. Cai, W. Liu, and X. Luo. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106(494):594–607, 2011.

[48] T. T. Cai, Z. Ren, and H. H. Zhou. Estimating structured high-dimensional covariance and precision matrices: Optimal rates and adaptive estimation. *Electronic Journal of Statistics*, 10(1):1–59, 2016.

[49] T. T. Cai and M. Yuan. Adaptive covariance matrix estimation through block thresholding. *The Annals of Statistics*, 40(4):2014–2042, 2012.

[50] T. T. Cai, C.-H. Zhang, and H. H. Zhou. Optimal rates of convergence for covariance matrix estimation. *The Annals of Statistics*, 38(4):2118–2144, 2010.

[51] G. Campos, A. Zimek, J. Sander, et al. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, 30(4):891–927, 7 2016.

[52] A. A. Cardenas, P. K. Manadhata, and S. P. Rajan. Big data analytics for security. *IEEE Security & Privacy*, 11(6):74–76, 2013.

[53] M. E. Celebi, H. A. Kingravi, and P. A. Vela. A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Systems with Applications*, 40(1):200–210, 2013.

[54] R.-B. Chen, M. Guo, W. K. Härdle, and S.-F. Huang. COPICA—independent component analysis via copula techniques. *Statistics and Computing*, 25(2):273–288, 2015.

[55] S. Cho, H. Hong, and B.-C. Ha. A hybrid approach based on the combination of variable selection using decision trees and case-based reasoning using the Maha-

lanobis distance: For bankruptcy prediction. *Expert Systems with Applications*, 37(4):3482–3488, 2010.

[56] A. Chokniwal and M. Singh. Faster Mahalanobis k-means clustering for Gaussian distributions. In *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 947–952, 2016.

[57] S. Chountasis, V. N. Katsikis, and D. Pappas. Applications of the Moore–Penrose inverse in digital image restoration. *Mathematical Problems in Engineering*, 2009, 2009.

[58] F. Cincotti, D. Mattia, C. Babiloni, F. Carducci, L. Bianchi, M. del RJ, J. Mourino, S. Salinari, M. Marciani, and F. Babiloni. Classification of EEG mental patterns by using two scalp electrodes and Mahalanobis distance-based classifiers. *Methods of Information in Medicine*, 41(04):337–341, 2002.

[59] R. Clarke, H. W. Ressom, A. Wang, J. Xuan, M. C. Liu, E. A. Gehan, and Y. Wang. The properties of high-dimensional data spaces: implications for exploring gene and protein expression data. *Nature Reviews Cancer*, 8(1):37–49, 2008.

[60] P. Courrieu. Fast computation of Moore–Penrose inverse matrices. *Neural Information Processing-Letters and Reviews*, 8(2), 2005.

[61] S. Dasgupta and A. Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

[62] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart. The Mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, 50(1):1–18, 2000.

[63] E. Debie and K. Shafi. Implications of the curse of dimensionality for supervised learning classifier systems: theoretical and empirical analyses. *Pattern Analysis and Applications*, 22(2):519–536, 2019.

[64] A. P. Dempster. Covariance selection. *Biometrics*, pages 157–175, 1972.

[65] B. Deng and G. Chen. A note on the generalized Bott–Duffin inverse. *Applied Mathematics Letters*, 20(7):746–750, 2007.

[66] M. Drazin. Pseudo-inverses in associative rings and semigroups. *The American Mathematical Monthly*, 65(7):506–514, 1958.

[67] C. C. Drovandi and A. N. Pettitt. Estimation of parameters for macroparasite population evolution using approximate Bayesian computation. *Biometrics*, 67(1):225–233, 2011.

[68] D. Dua and C. Graff. UCI machine learning repository, 2017. URL: http://archive.ics.uci.edu/ml.

[69] R. J. Durrant and A. Kabán. When is 'nearest neighbour' meaningful: A converse theorem and implications. *Journal of Complexity*, 25(4):385–397, 2009.

[70] N. El Karoui. Operator norm consistent estimation of large-dimensional sparse covariance matrices. *The Annals of Statistics*, 36(6):2717–2756, 2008.

[71] C. Elkan. Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 147–153, 2003.

[72] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[73] T. R. Etherington. Mahalanobis distances and ecological niche modelling: correcting a chi-squared probability error. *PeerJ*, 7:e6678, 2019.

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

[75] J. Fan, Y. Liao, and H. Liu. An overview of the estimation of large covariance and precision matrices. *The Econometrics Journal*, 19(1):C1–C32, 2016.

[76] M. Filannino. Dbworld e-mail classification using a very small corpus. *The University of Manchester*, 86, 2011.

[77] J. P. Fillmore. A note on rotation matrices. *IEEE Computer Graphics and Applications*, 4(2):30–33, 1984.

[78] P. Filzmoser, M. Gschwandtner, and V. Todorov. Review of sparse methods in regression and classification with application to chemometrics. *Journal of Chemometrics*, 26(3-4):42–51, 2012.

[79] R. A. Fisher et al. 138: The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

[80] E. Fix and J. L. Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review/Revue Internationale de Statistique*, 57(3):238–247, 1989.

[81] D. François, V. Wertz, and M. Verleysen. The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering*, 19(7):873–886, 2007.

[82] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[83] W. Gander. Algorithms for the QR decomposition. *Res. Rep*, 80(02):1251–1268, 1980.

[84] J. Gillard, E. O'Riordan, and A. Zhigljavsky. Polynomial whitening for high-dimensional data. *Computational Statistics*, 2022.

[85] J. Gillard, E. O'Riordan, and A. Zhigljavsky. Simplicial and minimal-variance distances in multivariate data analysis. *Journal of Statistical Theory and Practice*, 16(1):1–30, 2022.

[86] C. R. Givens and R. M. Shortt. A class of Wasserstein metrics for probability distributions. *Michigan Mathematical Journal*, 31(2):231–240, 1984.

[87] R. Gnanadesikan, J. W. Harvey, and J. R. Kettenring. Mahalanobis metrics for cluster analysis. *Sankhyā: The Indian Journal of Statistics, Series A (1961-2002)*, 55(3):494–505, 1993.

[88] R. Gnanadesikan and J. R. Kettenring. Robust estimates, residuals, and outlier detection with multiresponse data. *Biometrics*, pages 81–124, 1972.

[89] G. H. Golub and Z. Strakoš. Estimates in quadratic formulas. *Numerical Algorithms*, 8(2):241–268, 1994.

[90] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[91] A. N. Gorban and I. Y. Tyukin. Blessing of dimensionality: mathematical foundations of the statistical physics of data. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2118):20170237, 2018.

[92] J. Gui and H. Li. Penalized Cox regression analysis in the high-dimensional and low-sample size settings, with applications to microarray gene expression data. *Bioinformatics*, 21(13):3001–3008, 2005.

[93] S.-M. Guo, L.-C. Chen, and J. S.-H. Tsai. A boundary method for outlier detection based on support vector domain description. *Pattern Recognition*, 42(1):77–83, 2009.

[94] F. K. Haghani and F. Soleymani. An improved Schulz-type iterative method for matrix inversion with application. *Transactions of the Institute of Measurement and Control*, 36(8):983–991, 2014.

[95] P. Hall, J. S. Marron, and A. Neeman. Geometric representation of high dimension, low sample size data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(3):427–444, 2005.

[96] W. Härdle and L. Simar. *Applied multivariate statistical analysis*, volume 22007. Springer, 2007.

[97] R. H. Hariri, E. M. Fredericks, and K. M. Bowers. Uncertainty in big data analytics: survey, opportunities, and challenges. *Journal of Big Data*, 6(1):1–16, 2019.

[98] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020. doi:10.1038/s41586-020-2649-2.

[99] M. Healy. Multiple regression with a singular matrix. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 17(2):110–117, 1968.

[100] N. Higham. What is a Vandermonde matrix? https://nhigham.com/2021/06/15/what-is-a-vandermonde-matrix/, 2021. Accessed: 22-05-2022.

[101] N. J. Higham. *Functions of matrices: theory and computation*. SIAM, 2008.

[102] N. J. Higham and N. Strabić. Anderson acceleration of the alternating projections method for computing the nearest correlation matrix. *Numerical Algorithms*, 72(4):1021–1042, 2016.

[103] A. Hinneburg, C. C. Aggarwal, and D. A. Keim. What is the nearest neighbor in high dimensional spaces? In *26th International Conference on Very Large Databases*, pages 506–515, 2000.

[104] H. S. Hoang and R. Baraille. A regularized estimator for linear regression model with possibly singular covariance. *IEEE Transactions on Automatic Control*, 58(1):236–241, 2012.

[105] Z.-Q. Hong and J.-Y. Yang. Optimal discriminant plane for a small number of samples and design method of classifier on the plane. *Pattern Recognition*, 24(4):317–324, 1991.

[106] Z. H. Hoo, J. Candlish, and D. Teare. What is an ROC curve? *Emergency Medicine Journal*, 34(6):357–359, 2017.

[107] M. Hossain. Whitening and coloring transformations for multivariate Gaussian data. *A slecture partly based on the ECE662 Spring*, 2014.

[108] M. E. Houle, H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek. Can shared-neighbor distances defeat the curse of dimensionality? In *International Conference on Scientific and Statistical Database Management*, pages 482–500. Springer, 2010.

[109] D. C. Hoyle. Accuracy of pseudo-inverse covariance learning—a random matrix theory analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, 33(7):1470–1481, 2010.

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

[111] L. Huang, D. Yang, B. Lang, and J. Deng. Decorrelated batch normalization. In

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 791–800, 2018.

[112] L. Huang, L. Zhao, Y. Zhou, F. Zhu, L. Liu, and L. Shao. An investigation into the stochasticity of batch whitening. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6439–6448, 2020.

[113] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

[114] M. F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

[115] A. Hyvärinen, J. Hurri, and P. O. Hoyer. *Natural image statistics: A probabilistic approach to early computational vision.*, volume 39. Springer Science & Business Media, 2009.

[116] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 13(4-5):411–430, 2000.

[117] S. Imori and D. Von Rosen. On the mean and dispersion of the Moore–Penrose generalized inverse of a Wishart matrix. *The Electronic Journal of Linear Algebra*, 36:124–133, 2020.

[118] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456. PMLR, 2015.

[119] J. Jacques and D. Fraix-Burnet. Linear regression in high dimension and/or for correlated inputs. *European Astronomical Society Publications Series*, 66:149–165, 2014.

[120] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.

[121] T. Jendoubi and K. Strimmer. A whitening approach to probabilistic canonical correlation analysis for omics data integration. *BMC Bioinformatics*, 20(1):1–13, 2019.

[122] I. M. Johnstone and A. Y. Lu. Sparse principal components analysis. *arXiv preprint arXiv:0901.4392*, 2009.

[123] I. M. Johnstone and D. Paul. PCA in high dimensions: An orientation. *Proceedings of the IEEE*, 106(8):1277–1292, 2018.

[124] I. M. Johnstone and D. M. Titterington. Statistical challenges of high-dimensional data. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1906):4237–4253, 2009.

[125] I. Jolliffe. *Principal Component Analysis*. Springer New York, NY, 1986.

[126] I. T. Jolliffe and J. Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[127] S. Kandanaarachchi, M. A. Muñoz, R. J. Hyndman, and K. Smith-Miles. On normalization and algorithm selection for unsupervised outlier detection. *Data Mining and Knowledge Discovery*, 34(2):309–354, 2020.

[128] N. Katayama and S. Satoh. Distinctiveness-sensitive nearest-neighbor search for efficient similarity retrieval of multimedia information. In *Proceedings 17th International Conference on Data Engineering*, pages 493–502. IEEE, 2001.

[129] V. N. Katsikis, D. Pappas, and A. Petralias. An improved method for the computation of the Moore–Penrose inverse matrix. *Applied Mathematics and Computation*, 217(23):9828–9834, 2011.

[130] A. Kessy, A. Lewin, and K. Strimmer. Optimal whitening and decorrelation. *The American Statistician*, 72(4):309–314, 2018.

[131] N. Kishore Kumar and J. Schneider. Literature survey on low rank approximation of matrices. *Linear Multilinear Algebra*, 65(11):2212–2244, 2017.

[132] E. M. Knorr, R. T. Ng, and V. Tucakov. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3):237–253, 2000.

[133] A. Koivunen and A. Kostinski. The feasibility of data whitening to improve performance of weather radar. *Journal of Applied Meteorology*, 38(6):741–749, 1999.

[134] S. G. Kong, J. Heo, B. R. Abidi, J. Paik, and M. A. Abidi. Recent advances in visual and infrared face recognition—a review. *Computer Vision and Image Understanding*, 97(1):103–135, 2005.

[135] G. Köthe. Topological vector spaces. In *Topological Vector Spaces I*, pages 123–201. Springer, 1983.

[136] H.-P. Kriegel, P. Kröger, and A. Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(1):1–58, 2009.

[137] M. Kritzman and Y. Li. Skulls, financial turbulence, and risk management. *Financial Analysts Journal*, 66(5):30–41, 2010.

[138] M. Kryszkiewicz and P. Lasek. TI-DBSCAN: Clustering with DBSCAN by means of the triangle inequality. In *International Conference on Rough Sets and Current Trends in Computing*, pages 60–69. Springer, 2010.

[139] S. Kudyba and S. Kudyba. *Big data, mining, and analytics*. Auerbach Publications Boca Raton, 2014.

[140] A. Lahav, R. Talmon, and Y. Kluger. Mahalanobis distance informed by clustering. *Information and Inference: A Journal of the IMA*, 8(2):377–406, 2019.

[141] C. Lam. High-dimensional covariance matrix estimation. *Wiley Interdisciplinary Reviews: Computational Statistics*, 12(2):e1485, 2020.

[142] C. Lam and J. Fan. Sparsistency and rates of convergence in large covariance matrix estimation. *Annals of Statistics*, 37(6B):4254, 2009.

[143] T. Lancewicki and M. Aladjem. Multi-target shrinkage estimation for covariance matrices. *IEEE Transactions on Signal Processing*, 62(24):6380–6390, 2014.

[144] Y. LeCun, C. Cortes, and C. Burges. MNIST handwritten digit database, 2010. URL: ATTLabs[Online].Available:http://yann.lecun.com/exdb/mnist.

[145] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade*, pages 9–48. Springer, 2012.

[146] O. Ledoit and S. Péché. Eigenvectors of some large sample covariance matrix ensembles. *Probability Theory and Related Fields*, 151(1):233–264, 2011.

[147] O. Ledoit and M. Wolf. Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. *Journal of Empirical Finance*, 10(5):603–621, 2003.

[148] O. Ledoit and M. Wolf. Honey, I shrunk the sample covariance matrix. *The Journal of Portfolio Management*, 30(4):110–119, 2004.

[149] O. Ledoit and M. Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411, 2004.

[150] O. Ledoit and M. Wolf. The power of (non-)linear shrinking: A review and guide to covariance matrix estimation. *Journal of Financial Econometrics*, 20(1):187–218, 2022.

[151] K. Lee and J. Lee. Estimating large precision matrices via modified cholesky decomposition. *Statistica Sinica*, 31(1):173–196, 2021.

[152] M. Lee and D. Kim. On the use of the Moore–Penrose generalized inverse in the portfolio optimization problem. *Finance Research Letters*, 22:259–267, 2017.

[153] S. Lee, D. Cook, N. da Silva, U. Laa, N. Spyrison, E. Wang, and H. S. Zhang. The state-of-the-art on tours for dynamic visualization of high-dimensional data. *Wiley Interdisciplinary Reviews: Computational Statistics*, 14(4):e1573, 2022.

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

[155] D. Li, C. Chen, Q. Lv, J. Yan, L. Shang, and S. Chu. Low-rank matrix approximation with stability. In *International Conference on Machine Learning*, pages 295–303. PMLR, 2016.

[156] G. Li and J. Zhang. Sphering and its properties. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 119–133, 1998.

[157] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007.

[158] B. G. Lindsay. On the determinants of moment matrices. *The Annals of Statistics*, pages 711–721, 1989.

[159] B. Liu, Y. Wei, Y. Zhang, and Q. Yang. Deep neural networks for high dimension, low sample size data. In *IJCAI*, pages 2287–2293, 2017.

[160] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[161] P. Luo. Learning deep architectures via generalized whitened neural networks. In *International Conference on Machine Learning*, pages 2238–2246. PMLR, 2017.

[162] J. R. Magnus et al. *The moments of products of quadratic forms in normal variables*. Univ., Instituut voor Actuariaat en Econometrie, 1978.

[163] P. C. Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936.

[164] P. Mahe, M. Arsac, S. Chatellier, V. Monnin, N. Perrot, S. Mailler, V. Girard, M. Ramjeet, J. Surre, B. Lacroix, et al. Automatic identification of mixed bacterial species fingerprints in a MALDI-TOF mass-spectrum. *Bioinformatics*, 30(9):1280–1286, 2014.

[165] A. Majumdar, J. S. Witte, and S. Ghosh. Semiparametric allelic tests for mapping multiple phenotypes: binomial regression and Mahalanobis distance. *Genetic epidemiology*, 39(8):635–650, 2015.

[166] G. Malsiner-Walli, S. Frühwirth-Schnatter, and B. Grün. Model-based clustering based on sparse finite Gaussian mixtures. *Statistics and Computing*, 26(1-2):303–324, 2016.

[167] C. D. Manning, H. Schütze, and P. Raghavan. *Introduction to information retrieval*. Cambridge University Press, 2008.

[168] K. Mardia and J. Kent. Multivariate analysis. *New York: AcademicPress*, 1979.

[169] H. Martens, M. Høy, B. M. Wise, R. Bro, and P. B. Brockhoff. Pre-whitening of data by covariance-weighted pre-processing. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 17(3):153–165, 2003.

[170] N. Matamala, M. T. Vargas, R. Gonzalez-Campora, R. Minambres, J. I. Arias, P. Menendez, E. Andres-Leon, G. Gomez-Lopez, K. Yanowsky, J. Calvete-Candenas, et al. Tumor microrna expression profiling identifies circulating micrornas for early breast cancer detection. *Clinical Chemistry*, 61(8):1098–1106, 2015.

[171] A. M. Mathai and S. B. Provost. *Quadratic forms in random variables: theory and applications*. Dekker, 1992.

[172] D. S. Matteson and R. S. Tsay. Independent component analysis via distance covariance. *Journal of the American Statistical Association*, 112(518):623–637, 2017.

[173] N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, 34(3):1436–1462, 2006.

[174] I. Melnykov and V. Melnykov. On K-means algorithm with the use of Mahalanobis distances. *Statistics & Probability Letters*, 84:88 – 95, 2014.

[175] E. M. Mirkes, J. Allohibi, and A. Gorban. Fractional norms and quasinorms do not help to overcome the curse of dimensionality. *Entropy*, 22(10):1105, 2020.

[176] E. H. Moore. On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society*, 26:394–395, 1920.

[177] M. Mudrova and A. Procházka. Principal component analysis in image processing. In *Proceedings of the MATLAB Technical Computing Conference, Prague*, 2005.

[178] N. Najat and A. M. Abdulazeez. Gene clustering with partition around mediods algorithm based on weighted and normalized Mahalanobis distance. In *2017 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, pages 140–145. IEEE, 2017.

[179] H. Oja. Descriptive statistics for multivariate distributions. *Statistics & Probability Letters*, 1(6):327–332, 1983.

[180] Y. Pan, Z. Pan, Y. Wang, and W. Wang. A new fast search algorithm for exact k-nearest neighbors based on optimal triangle-inequality-based check strategy. *Knowledge-Based Systems*, 189:105088, 2020.

[181] C. Park and H. Park. A fast dimension reduction algorithm with applications on face recognition and text classification. *Retrieved from the University of Minnesota Digital Conservancy*, 2003.

[182] L. Parsons, E. Haque, and H. Liu. Subspace clustering for high dimensional data: a review. *ACM SIGKDD Explorations Newsletter*, 6(1):90–105, 2004.

[183] B. K. Patra. Using the triangle inequality to accelerate density based outlier detection method. *Procedia Technology*, 6:469–474, 2012.

[184] D. Paul. Asymptotics of sample eigenstructure for a large dimensional spiked covariance model. *Statistica Sinica*, pages 1617–1642, 2007.

[185] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

[186] R. Penrose. A generalized inverse for matrices. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 51, pages 406–413. Cambridge University Press, 1955.

[187] V. Perlibakas. Distance measures for PCA-based face recognition. *Pattern Recognition Letters*, 25(6):711–724, 2004.

[188] M. D. Petković and P. S. Stanimirović. Iterative method for computing the Moore–Penrose inverse based on Penrose equations. *Journal of Computational and Applied Mathematics*, 235(6):1604–1613, 2011.

[189] M. Pourahmadi. *High-dimensional covariance estimation: with high-dimensional data*, volume 882. John Wiley & Sons, 2013.

[190] W. B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

[191] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge University Press, 2007.

[192] L. Pronzato, H. Wynn, and A. Zhigljavsky. Simplicial variances, potentials and Mahalanobis distances. *Journal of Multivariate Analysis*, pages 276–289, 2018.

[193] S. Prykhodko, N. Prykhodko, L. Makarova, and A. Pukhalevych. Application of the squared Mahalanobis distance for detecting outliers in multivariate non-Gaussian data. In *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, pages 962–965. IEEE, 2018.

[194] H. Qi and D. Sun. An augmented Lagrangian dual approach for the H-weighted nearest correlation matrix problem. *IMA Journal of Numerical Analysis*, 31(2):491–511, 2011.

[195] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 427–438, 2000.

[196] S. Raudys and R. P. Duin. Expected classification error of the Fisher linear classifier with pseudo-inverse covariance matrix. *Pattern Recognition Letters*, 19(5-6):385–392, 1998.

[197] P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu. High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935–980, 2011.

[198] S. Rayana. ODDS library, 2016. URL: http://odds.cs.stonybrook.edu.

[199] A. C. Rencher and G. B. Schaalje. *Linear Models in Statistics (2nd ed.)*. Wiley-Intersci., 2008.

[200] M. Roser and H. Ritchie. Technological change. *Our World in Data*, 2013. URL: https://ourworldindata.org/technological-change.

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

[202] A. J. Rothman, E. Levina, and J. Zhu. Generalized thresholding of large covariance matrices. *Journal of the American Statistical Association*, 104(485):177–186, 2009.

[203] P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of

cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[204] S. Sarkar and A. K. Ghosh. On perfect clustering of high dimension, low sample size data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(9):2257–2272, 2019.

[205] J. Schäfer and K. Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4(1), 2005.

[206] A. Schuler, V. Liu, J. Wan, A. Callahan, M. Udell, D. E. Stark, and N. H. Shah. Discovering patient phenotypes using generalized low rank models. In *Biocomputing 2016: Proceedings of the Pacific Symposium*, pages 144–155. World Scientific, 2016.

[207] G. A. Seber and A. J. Lee. *Linear regression analysis*, volume 329. John Wiley & Sons, 2012.

[208] L. Shen, M. J. Er, and Q. Yin. Classification for high-dimension low-sample size data. *Pattern Recognition*, page 108828, 2022.

[209] X. Shi, Z. Guo, F. Nie, L. Yang, J. You, and D. Tao. Two-dimensional whitening reconstruction for enhancing robustness of principal component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2130–2136, 2015.

[210] O. M. Shir and A. Yehudayoff. On the covariance-hessian relation in evolution strategies. *Theoretical Computer Science*, 801:157–174, 2020.

[211] K. Sin and L. Muthu. Application of big data in education data mining and learning analytics–a literature review. *ICTACT Journal on Soft Computing*, 5(4), 2015.

[212] A. Singh, A. Yadav, and A. Rana. K-means with three different distance metrics. *International Journal of Computer Applications*, 67(10), 2013.

[213] K. Singh, M. Jardak, A. Sandu, K. Bowman, M. Lee, and D. Jones. Construction of non-diagonal background error covariance matrices for global chemical data assimilation. *Geoscientific Model Development*, 4(2):299–316, 2011.

[214] A. Smiti. A critical overview of outlier detection methods. *Computer Science Review*, 38:100306, 2020.

[215] A. Smoktunowicz and I. Wróbel. Numerical aspects of computing the Moore–Penrose inverse of full column rank matrices. *BIT Numerical Mathematics*, 52(2):503–524, 2012.

[216] D. K. Sodickson and C. A. McKenzie. A generalized approach to parallel magnetic resonance imaging. *Medical Physics*, 28(8):1629–1643, 2001.

[217] M. S. Srivastava and M. Du. A test for the mean vector with fewer observations than the dimension. *Journal of Multivariate Analysis*, 99(3):386–402, 2008.

[218] N. Srivastava and S. Rao. Learning-based text classifiers using the Mahalanobis distance for correlated datasets. *International Journal of Big Data Intelligence*, 3(1):18–27, 2016.

[219] C. Stein. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, pages 197–206, 1956.

[220] D. Steinley. Properties of the Hubert-Arable adjusted rand index. *Psychological Methods*, 9(3):386, 2004.

[221] D. Steinley. K-means clustering: a half-century synthesis. *British Journal of Mathematical and Statistical Psychology*, 59(1):1–34, 2006.

[222] G. V. Stevens. On the inverse of the covariance matrix in portfolio analysis. *The Journal of Finance*, 53(5):1821–1827, 1998.

[223] P. F. Stifanelli, T. M. Creanza, R. Anglani, V. C. Liuzzi, S. Mukherjee, and N. Ancona. A comparative study of Gaussian graphical model approaches for genomic data. *arXiv preprint arXiv:1107.0261*, 2011.

[224] S. Stöckl and M. Hanke. Financial applications of the Mahalanobis distance. *Applied Economics and Finance*, 1(2):78–84, 2014.

[225] C.-T. Su and T.-S. Li. A Mahalanobis distance based classifier for diagnosis of diseases. *Journal of the Chinese Institute of Industrial Engineers*, 19(5):41–47, 2002.

[226] H. Suzuki, M. Sota, C. J. Brown, and E. M. Top. Using Mahalanobis distance to compare genomic signatures between bacterial plasmids and chromosomes. *Nucleic Acids Research*, 36(22):e147–e147, 2008.

[227] L. Tan. *A Generalized Framework of Linear Multivariable Control*. Butterworth-Heinemann, 2017.

[228] K. Taunk, S. De, S. Verma, and A. Swetapadma. A brief review of nearest neighbor algorithm for learning and classification. In *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pages 1255–1260. IEEE, 2019.

[229] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22(3):281–285, 1999.

[230] M. H. Tekieh and B. Raahemi. Importance of data mining in healthcare: a survey. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 1057–1062, 2015.

[231] M. Thameri, A. Kammoun, K. Abed-Meraim, and A. Belouchrani. Fast principal component analysis and data whitening algorithms. In *International Workshop on Systems, Signal Processing and their Applications, WOSSPA*, pages 139–142. IEEE, 2011.

[232] A. Torokhti and S. Friedland. Towards theory of generic principal component analysis. *Journal of Multivariate Analysis*, 100(4):661–669, 2009.

[233] S. Ubaru, J. Chen, and Y. Saad. Fast estimation of tr(f(a)) via stochastic Lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1075–1099, 2017.

[234] S. Ubaru and Y. Saad. Applications of trace estimation techniques. In *International Conference on High Performance Computing in Science and Engineering*, pages 19–33. Springer, 2017.

[235] M. Udell and A. Townsend. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 1(1):144–160, 2019.

[236] C. F. Van Loan and G. Golub. *Matrix computations (Johns Hopkins studies in mathematical sciences)*. The Johns Hopkins University Press, 1996.

[237] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo. OpenML: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013.

[238] M. Verleysen and D. François. The curse of dimensionality in data mining and time series prediction. In *International Work-Conference on Artificial Neural Networks*, pages 758–770. Springer, 2005.

[239] M. Verleysen, D. Francois, G. Simon, and V. Wertz. On the effects of dimensionality on data analysis with neural networks. In *International Work-Conference on Artificial Neural Networks*, pages 105–112. Springer, 2003.

[240] D. Ververidis and C. Kotropoulos. Gaussian mixture modeling by exploiting the Mahalanobis distance. *IEEE Transactions on Signal Processing*, 56(7):2797–2811, 2008.

[241] R. Vidal and P. Favaro. Low rank subspace clustering (LRSC). *Pattern Recognition Letters*, 43:47–61, 2014.

[242] D. Wang, D. S. Yeung, and E. C. Tsang. Weighted Mahalanobis distance kernels for support vector machines. *IEEE Transactions on Neural Networks*, 18(5):1453–1462, 2007.

[243] G. Wang, Y. Wei, and S. Qiao. *Generalized inverses: theory and computations*, volume 53. Springer, 2018.

[244] W. Wang and J. Fan. Asymptotics of empirical eigenstructure for high dimensional spiked covariance. *Annals of Statistics*, 45(3):1342, 2017.

[245] Z. Wang and D. W. Scott. Nonparametric density estimation for high-dimensional data—algorithms and applications. *Wiley Interdisciplinary Reviews: Computational Statistics*, 11(4):e1461, 2019.

[246] D. I. Warton. Penalized normal likelihood and ridge regularization of correlation and covariance matrices. *Journal of the American Statistical Association*, 103(481):340–349, 2008.

[247] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study

for similarity-search methods in high-dimensional spaces. In *VLDB*, volume 98, pages 194–205, 1998.

[248] X.-K. Wei, G.-B. Huang, and Y.-H. Li. Mahalanobis ellipsoidal learning machine for one class classification. In *2007 International Conference on Machine Learning and Cybernetics*, volume 6, pages 3528–3533. IEEE, 2007.

[249] S. Wiesler and H. Ney. A convergence analysis of log-linear training. *Advances in Neural Information Processing Systems*, 24, 2011.

[250] S. S. Wilks. Multidimensional statistical scatter. *Contributions to Probability and Statistics (Essays in Honor of Harold Hotelling)(Olkin, Ingram et al., eds.)*, pages 486–503, 1960.

[251] W. A. Wilson. On semi-metric spaces. *American Journal of Mathematics*, 53(2):361–373, 1931.

[252] A. Wismüller. The exploration machine: a novel method for analyzing high-dimensional data in computer-aided diagnosis. In *Medical Imaging 2009: Computer-Aided Diagnosis*, volume 7260, pages 143–149. SPIE, 2009.

[253] W. H. Wolberg, W. N. Street, and O. L. Mangasarian. Breast cancer Wisconsin (diagnostic) data set. *UCI Machine Learning Repository*, 1992.

[254] J. H. Won and S.-J. Kim. Maximum likelihood covariance estimation with a condition number constraint. In *2006 Fortieth Asilomar Conference on Signals, Systems and Computers*, pages 1445–1449. IEEE, 2006.

[255] D. Wu, D. Wang, M. Q. Zhang, and J. Gu. Fast dimension reduction and integrative clustering of multi-omics data using low-rank approximation: application to cancer molecular classification. *BMC Genomics*, 16(1):1022, 2015.

[256] T.-J. Wu, J. P. Burke, and D. B. Davison. A measure of DNA sequence dissimilarity based on Mahalanobis distance between frequencies of words. *Biometrics*, pages 1431–1439, 1997.

[257] W. B. Wu and M. Pourahmadi. Nonparametric estimation of large covariance matrices of longitudinal data. *Biometrika*, 90(4):831–844, 2003.

[258] W. B. Wu and M. Pourahmadi. Banding sample autocovariance matrices of stationary processes. *Statistica Sinica*, pages 1755–1768, 2009.

[259] R. Xiang, K. Khare, and M. Ghosh. High dimensional posterior convergence rates for decomposable graphical models. *Electronic Journal of Statistics*, 9(2):2828–2854, 2015.

[260] S. Xiang, F. Nie, and C. Zhang. Learning a Mahalanobis distance metric for data clustering and classification. *Pattern Recognition*, 41(12):3600–3612, 2008.

[261] Z. Xiao. Efficient GMM estimation with singular system of moment conditions. *Statistical Theory and Related Fields*, 4(2):172–178, 2020.

[262] L. Yang and R. Jin. Distance metric learning: A comprehensive survey. *Michigan State University*, 2(2):4, 2006.

[263] K. Yata and M. Aoshima. PCA consistency for the power spiked model in high-dimensional settings. *Journal of Multivariate Analysis*, 122:334–354, 2013.

[264] J. Ye and T. Wang. Regularized discriminant analysis for high dimensional, low sample size data. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 454–463, 2006.

[265] J. Ye and T. Xiong. Null space versus orthogonal linear discriminant analysis. In *Proceedings of the 23rd international conference on Machine learning*, pages 1073–1080, 2006.

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

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

[268] S. Zafeiriou and N. Laskaris. On the improvement of support vector techniques for clustering by means of whitening transform. *IEEE Signal Processing Letters*, 15:198–201, 2008.

[269] Y. Zhang, B. Du, L. Zhang, and S. Wang. A low-rank and sparse matrix decomposition-based Mahalanobis distance method for hyperspectral anomaly de-

tection. *IEEE Transactions on Geoscience and Remote Sensing*, 54(3):1376–1389, 2015.

[270] X. Zhao, Y. Li, and Q. Zhao. Mahalanobis distance based on fuzzy clustering algorithm for image segmentation. *Digital Signal Processing*, 43:8–16, 2015.

[271] Y. Zhao, Z. Nasrullah, and Z. Li. PyOD: A Python toolbox for scalable outlier detection. *Journal of Machine Learning Research*, 20(96):1–7, 2019.

[272] B. Zheng and R. Bapat. Generalized inverse A(2)T,S and a rank equation. *Applied Mathematics and Computation*, 155(2):407–415, 2004.

[273] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan. Large-scale parallel collaborative filtering for the Netflix prize. In *International Conference on Algorithmic Applications in Management*, pages 337–348. Springer, 2008.

[274] A. Zimek, E. Schubert, and H.-P. Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.

[275] V. Zuber and K. Strimmer. Gene ranking and biomarker discovery under correlation. *Bioinformatics*, 25(20):2700–2707, 2009.

[276] Y. Zuo. Multidimensional medians and uniqueness. *Computational Statistics & Data Analysis*, 66:82–88, 2013.