Dimension reduction methods for high-dimensional datasets

Hayley Randall

2022

Submitted in partial fulfillment of
the requirements for the degree of
Master of Philosophy

School of Mathematics
Ysgol Mathemateg

Summary

In recent years computer power has increased massively which consequently has led to an increase in the size of data. The steep increase in size has led to a vast need for more modern ways of analysing this data. Classical methods for analysing data were intended for a low dimensional setting, hence an increasingly popular method of analysing large data is to perform a dimension reduction technique first to project the data into a lower dimension. A 'good' dimension reduction technique accurately predicts the correct dimension reduction subspace, without having a significant impact on the computational efficiency of the calculations. There are many dimension reduction methods already developed but few have successfully achieved a high level of accuracy without sacrificing the computation time. Our aim is to develop a method that rivals previous methods with high accuracy and those which are efficient computationally.

Another common drawback with classic methods is that not many are realistic options for data where the dimension size exceeds the sample size, many depend on calculating the inverse of the covariance matrix of the predictor variables which becomes singular as the dimension size surpasses the sample size. It has also been shown that many classic estimators of the central dimension reduction subspace do not remain consistent when the dimension size is larger than the sample size.

There are two main contributions from this work, we have developed a dimension reduction method using Distance-Weighted Discrimination (DWD) which has increased accuracy compared with classic methods and is computationally faster than more recent methods. We have also developed a dimension reduction method which can tackle larger datasets without being restricted by the dimension, and further improved the computational efficiency compared with classic methods in the form of a feature partitioning algorithm.

Declarations

This work has not been submitted in substance for any other degree or award at this or any other university or place of learning, nor is being submitted concurrently in candidature for any degree or other award.

Signed _____(candidate) Date _____

This thesis is being submitted in partial fulfillment of the requirements for the degree of Ph.D.

Signed _____(candidate) Date _____

This thesis is the result of my own independent work/investigation, except where otherwise stated, and the thesis has not been edited by a third party beyond what is permitted by Cardiff University's Policy on the Use of Third Party Editors by Research Degree Students. Other sources are acknowledged by explicit references. The views expressed are my own.

Signed _____(candidate) Date _____

I hereby give consent for my thesis, if accepted, to be available online in the University's Open Access repository and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed _____(candidate) Date _____

I hereby give consent for my thesis, if accepted, to be available online in the University's Open Access repository and for inter-library loans *after expiry of a bar on access previously approved by the Academic Standards & Quality Committee.*

Signed _____(candidate) Date _____

To my husband and best friend David. Without your support and constant encouragement this would not have been possible.

"Like what you do, and then you will do your best."

— Katherine Johnson

Acknowledgements

Firstly, words cannot describe my gratitude I feel for the consistent support I have received from my supervisor, Andreas Artemiou. You were always there to keep me on track and boost my confidence when I was having a bad day. A man I hope I can now call my friend and someone I truly respect. I would also like to thank my office colleague and best friend Asyl 'Asil' Hawa who could put a smile on my face even on the worst days. You gave me a person I could talk to when I really needed it and I hope I can return the favour.

A special mention goes to EPSRC and the School of Mathematics for their financial assistance, likewise to all the staff that provide us with constant support I would like to thank you for surrounding me with such a friendly and enjoyable working environment.

Finally, I would not be in the position I am without the encouragement I have received from my family. To my father who always believes in me even when I do not believe in myself and to my mother who patiently listens to me complain and for the helpful advice I have come to rely upon. Last but not least, to my father by choice rather than blood, the man who is always there when I call. Phil, in you I have found a person I can truly depend on and for that I will be forever grateful.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The computer capability available to the general population is increasing daily. The average hard-drive capacity and memory of a computer is constantly increasing due to technological growth and mass production causing a positive effect on the price of large hard-drive capacity and memory. The speed of computers, and thus the overall power, is also on the rise which has led to more and more data being collected every day. It is predicted that each day approximately 2.5 quintillion bytes of data is produced. We have seen an increase in not only the amount of data collected but also the size of the data being collected. Big data is here, and we need efficient ways to analyse it.

1.0.1 Large data in the real world

Large data causes a plethora of problems from a dissemination standpoint, this is commonly referred to as the curse of dimensionality. With a constantly changing world what we class as high dimensional data is relative and continuously increasing. Common occurrences of high dimensional data are medical records, web logs and DNA analysis. Some of the common obstacles faced when handling high dimensional data are explained below.

When the dimension is high using standard software and algorithms can be unrealistic in normal time. For example, most vector optimisation algorithms require optimising for each feature. For data with a dimension of 10 this can be costly but realistic. If instead you have data with a dimension of 100,000 then the number of possible vectors that need to be considered is extremely high.

If the dimension is higher than the sample size we refer to this data as high dimension low sample size (hdlss) data. When this occurs then often over-fitting of a model can happen. This refers to the model being to strongly fitted to the sample data and therefore unable to generalise well. If this is the case then the model is inaccurate for other samples of the data. The problem increases (inaccuracies

increase) as the dimension size increases or the sample size decreases. This can occur for smaller dimension data if the sample size is particularly small. For example, data with a sample size of 50 and a dimension of 100 is far more likely to suffer from over-fitting than data with a 100,000 samples and 20,000 features.

Another problem that occurs when the dimension is high in comparison to the sample size is sparsity in the data. This can cause a reduction in statistical significance as the data fills less of the data space. If you require all data points to be a maximum distance away from another data point then the number of sample required to achieve that increases significantly with each additional dimension. Alternatively if you are measuring the distance between points, an aim common to clustering analysis, then the distance between each point will increase as the dimension increases. This can be seen in the below plots.

Figure 1.1: Simulated normal 1-dimensional data with 50 samples.

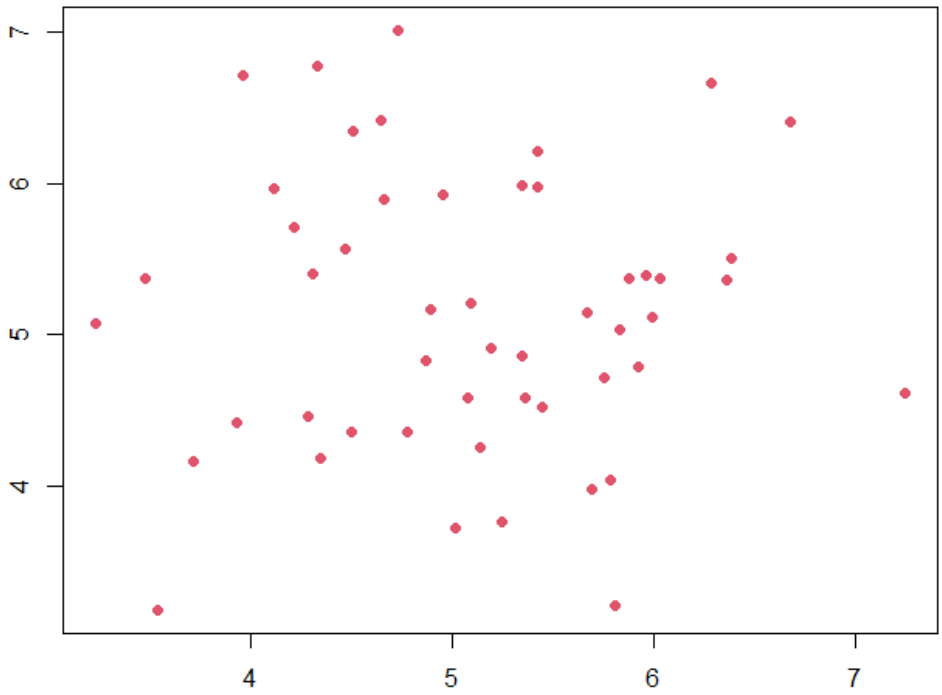

Figure 1.2: Simulated normal 2-dimensional data with 50 samples.

In the 1-dimensional case it appears as though much of the data sit close to one another. In the 2-dimensional case the distance between each point has increased considerable and once again the distance between the points in the 3-dimensional case is visibly higher than in the 1-dimensional and the 2-dimensional case.

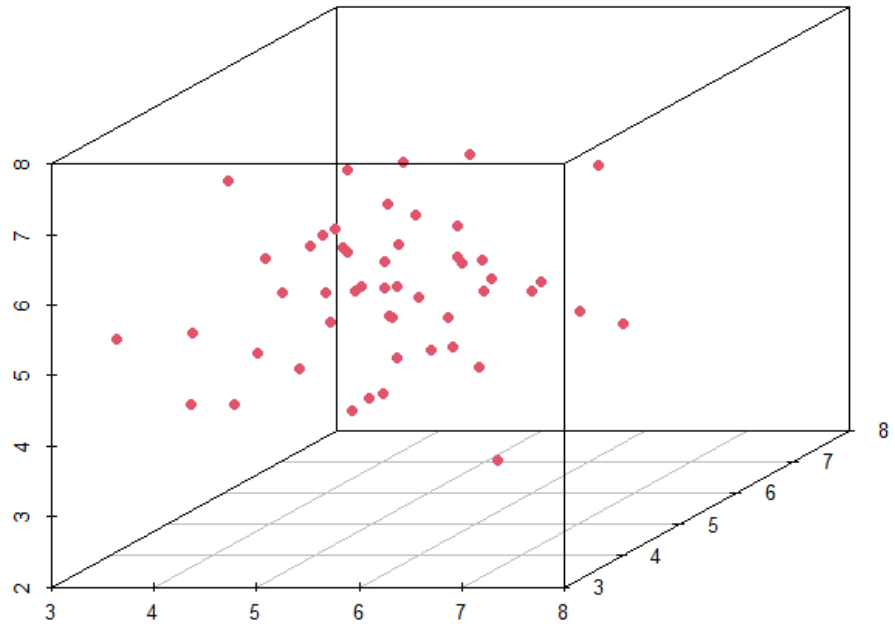

Figure 1.3: Simulated normal 3-dimensional data with 50 samples.

Finally, visualisation is not as simple when the dimension of the data exceeds 3. If the dimension of the data is larger than 3 it is often useful to attempt to visualise different slices of the data. Again, this is not realistic for larger dimensions. For example, the number of combinations of 3 dimensional views possible is equal to $\binom{\text{dimension size}}{3}$. For data with 10 features this is equal to 120. If the dimension is 10,000 then there are 166,616,670,000 different combinations of 3 features.

1.0.2 Established dimension reduction methods

A popular way to reduce the problems caused by high dimensional data is to perform a dimension reduction (DR) technique first. Reducing the data can lead to the dimension being a more manageable size for visualisation needs and classical analysing tools. Therefore, it is more important than ever that accurate and computationally efficient DR methods are developed.

Numerous classical methods already exist for DR, Li (1991), Cook and Weisberg (1991) and Li (1992). These methods are computationally efficient, however often produce less accurate results than many modern methods. The basic idea of dimension reduction is to find a linear combination of data which has a smaller dimension and retains all of the information. The more information that is retain the more accurate the method is. Producing a method which is computationally faster has never been more important but it is vital that we achieve this without loss of accuracy. A more recent method, developed by Li et al. (2011), proved effective at increasing the accuracy in comparison to other methods, however was significantly slower. We

aim to extend the work developed by others to produce a method for linear models which maintains the accuracy of newer methods and preserving the computational efficiency of the classical methods discussed.

The work produced by Li et al. (2011) combined linear DR methods and non-linear DR methods under a unified framework, which was the first method developed of this kind. Many data models are linear, however often a linear model is not sufficient for capturing the data. When this occurs non-linear DR is required to correctly reduce the dimension of the data. This motivates a need for a further extension of our method, to produce a non-linear method which mirrors the advantages of the linear method. Analogous to the work produced by Li et al. (2011), we will combine the linear and non-linear method under a unified framework.

In recent years, a considerable amount of focus has been put into adapting methods that separate the sample space or separating the feature space. The motivation for the separation of the sample space is to help improve the computational efficiency of results produced from big data. All the methods we have already considered have restrictions on the dimension size with respect to the sample size. By separating the feature space, the restriction that is often present in classical methods can be reduced, whilst ideally also improving or at least maintaining the computational efficiency. When separating the feature space in order to perform DR, one can proceed with a sequential based method or a parallel programming method. Previous work had been produced which performs DR directly, with few assumptions, by sub-setting the features and sequentially performing classical methods on the subsets.

We propose sub-setting the data by first decorrelating the variables which will allow us to use classical methods simultaneously on multiple machines and subsequently reducing the calculation time. Decorrelating the variables of a dataset will lead to restrictions on the dimension, similar to those found in classical methods. A method which is accurate, computationally efficient and has looser restriction on the dimension size is extremely desirable. For this reason, further investigation will be performed to evaluate the impact of replicating the previously defined method without the decorrelation step.

1.1 Research aims

The main aims of this work are as follows:

1. To develop a new linear method of DR that maintains the accuracy of modern methods while improving the computational efficiency.

2. Extend the method as a unified framework for linear and non-linear models.

3. Develop a new approach for DR through feature space partitioning by decorrelating the variables.

4. Investigate the effects of reproducing a method of DR through feature space partitioning, without decorrelating the variables.

1.2 Thesis structure

This work focuses solely on developing new DR techniques of different types. Each of the following chapters will take a similar form. Chapter 2 will concentrate on adapting a new linear DR method. We will look to extend the work produced by Li et al. (2011) in an attempt to improve the computational efficiency. The chapter will begin with a detailed literature review of previous work to help provide a clear background to the work which follows. The sections describing our new linear method will clearly outline the methodology with an estimation algorithm, an asymptotic analysis which will define any consistency restrictions, and an extensive simulation study to highlight the benefits of our method compared with similar methods.

Chapters 3 is a smaller chapter describing different types of order determination with particular attention paid to our chosen method. It will also include some simulation studies of our linear method using the chosen method for order determination.

The extension of our method into a unified framework for linear and non-linear models will be considered in chapter 4. Similar to chapter 2 we will begin with a small literature review of previous work before giving more detail describing our extension. The end of chapter 4 will contain simulation studies which will once again compare our method with similar methods.

Chapter 5 will once again begin with a literature review which will consider DR methods which separate the sample space and methods that separate the feature space. Chapter 5 will continue with the methodology for our proposed method which will begin with a method involving decorrelating the variables. We will then extend this adaption to include separation of both the feature space and the sample space, which will be concluded with a broad analysis of the method through synthetic and real data examples. Following from this we will investigate an extension to the method already developed, in which we skip the decorrelation step introduced. Included will be an estimation algorithm for separating only the feature space and whilst also separating the sample space. More simulation studies will be produced to further access the impact of not decorrelating the variables. The final analysis of this chapter will give a comparison between the methods with and without the decorrelation step.

The final chapter will consist of a conclusion and summary of all findings. A short explanation of future extensions will also be included.

1.3 Publications from this work

Randall et al. (2020):
Hayley Randall, Andreas Artemiou and Xingye Qiao (2020). Sufficient dimension reduction based on distance-weighted discrimination. Scandinavian Journal of Statistics.

1.4 Contributions

The contributions from this work are:

1. Much of this work focuses heavily on increasing computational efficiency which is beneficial for big data and real time data.

2. We have developed a simulation based dimension reduction method which separates the feature space without decorrelation the variables. This reduces the restriction on requiring the starting dimension size to be smaller than the sample size.

Chapter 2

Linear Sufficient Dimension Reduction

Let n denote the sample size and p denote the dimension of the data. In a regression setting, we have a p-dimensional predictor variable $\boldsymbol{X}$ and a response variable Y, where $\boldsymbol{X}$ is the data. Without loss of generality we only consider a univariate response variable. Our aim is to find a $p \times d$ matrix $\boldsymbol{\beta}$, where $d < p$, such that

$$Y \perp\!\!\!\perp \boldsymbol{X} | \boldsymbol{\beta}^{\mathsf{T}} \boldsymbol{X}. \tag{2.1}$$

In simple terms, we aim to find a set of linear combinations of the predictor variables, with dimension less than p, which can replace $\boldsymbol{X}$ without loss of information on the conditional distribution of $Y|\boldsymbol{X}$. In practice, we find the intersection of all the subspaces spanned by the columns of all the $\boldsymbol{\beta}$'s, frequently named the dimension reduction directions, that satisfy (2.1). We call this intersection the Central Dimension Reduction Subspace (CDRS) and is denoted by $S_{Y|\boldsymbol{X}}$. The CDRS is proven to exist and to be unique under mild conditions in Cook (1996) and Yin et al. (2008). We will assume the CDRS to exist for the remainder of this work. If $Y|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}$ has the same conditional distribution as $Y|\boldsymbol{X}$, then this is known as Sufficient Dimension Reduction (SDR).

Many dimension reduction methods rely on first standardising $\boldsymbol{X}$. This is possible since the central subspace transforms equivariantly under affine transformations of $\boldsymbol{X}$, proved in Cook (1998). The following theorem defines this condition more clearly. The proof is given in Li (2018) and is thus omitted.

Theorem 2.1 *If $A \in \mathbb{R}^{p\times p}$ is a nonsingular matrix and $b \in \mathbb{R}^p$, then*

$$S_{Y|\boldsymbol{X}} = A^{\mathsf{T}} S_{Y|A^{\mathsf{T}}\boldsymbol{X}+b}$$

All of the linear dimension reduction methods we discuss in this chapter have a common assumption described below.

Assumption 2.2 *(Linear Conditional Mean (LCM) assumption) For $\boldsymbol{\beta}$ defined in* (2.1)*, we assume* $\mathrm{E}[\boldsymbol{X}|\boldsymbol{\beta}^\mathsf{T}\boldsymbol{X}]$ *is a linear function of* $\boldsymbol{X}$*.*

This assumption has been proved to be equivalent to $\boldsymbol{X}$ having an elliptically contoured distribution and consequently allows $\mathrm{E}[\boldsymbol{X}|\boldsymbol{\beta}^\mathsf{T}\boldsymbol{X}]$ to be expressed as $\boldsymbol{P}_{\boldsymbol{\beta}}(\boldsymbol{\Sigma})\boldsymbol{X}$, where $\boldsymbol{P}_{\boldsymbol{\beta}}$ is the projection matrix $\boldsymbol{\beta}(\boldsymbol{\beta}^\mathsf{T}\boldsymbol{\Sigma}\boldsymbol{\beta})^{-1}\boldsymbol{\beta}^\mathsf{T}\boldsymbol{\Sigma}$. Another condition which is often, but not always, assumed in dimension reduction is described below.

Assumption 2.3 *(Constant Conditional Variance (CCV) assumption) For $\boldsymbol{\beta}$ defined in* (2.1)*, the conditional variance* $\mathrm{var}[\boldsymbol{X}|\boldsymbol{\beta}^\mathsf{T}\boldsymbol{X}]$ *is a non-random matrix.*

This assumption is satisfied if $\boldsymbol{X}$ has a Gaussian distribution with a nonsingular covariance matrix. This condition combined with the LCM assumption defined previously allows us to write $\mathrm{var}[\boldsymbol{X}|\boldsymbol{\beta}^\mathsf{T}\boldsymbol{X}]$ as $\boldsymbol{\Sigma}\boldsymbol{Q}_{\boldsymbol{\beta}}(\boldsymbol{\Sigma})$, where $\boldsymbol{Q}_{\boldsymbol{\beta}} = \boldsymbol{I} - \boldsymbol{P}_{\boldsymbol{\beta}}(\boldsymbol{\Sigma})$.

One of the tricks many classic algorithms use for SDR is the idea of slicing the response, which in most regression settings is a continuous random variable (see for example Li (1991) and Li et al. (2011)). When the response is discrete this step is ignored as each discrete value is considered a slice. This is performed as follows, let Ω_Y be the support of Y. We then slice Ω_Y into h slices to give a disjoint subset A_i of Ω_Y, where $i = 1, \ldots, h$. The work produced by Li et al. (2011) uses classification and therefore the author defines A_1 and A_2 to be disjoint subsets of Ω_Y, to give

$$\tilde{Y} = I(Y \in A_1) - I(Y \in A_2). \tag{2.2}$$

The main difference is the requirement of two subsets at a time since the classification step depends on a response variable with at least two levels.

Throughout this section we assume the effective dimension d to be known. We will discuss the literature on estimating d in the next chapter where we will also run simulations to demonstrate how one of the methods to estimate d works without proposed algorithm.

2.1 Previous work

Some literature on linear SDR includes and is not limited to Sliced Inverse Regression (SIR) by Li (1991), Sliced Average Variance Estimation (SAVE) by Cook and Weisberg (1991), principal Hessian directions (pHd) by Li (1992), Contour Regression (CR) by Li et al. (2005), Slice Inverse Mean Difference by Artemiou and Tian (2015) and Slice Inverse Median Difference by Babos and Artemiou (2020), among others. A brief summary of some of the methods is given below.

2.1.1 Sliced inverse regression

In a seminal work by Li (1991), Sliced Inverse Regression (SIR) was proposed. SIR aims to find vectors of length p known as the sufficient dimension reduction directions, $\boldsymbol{\beta}_i$'s, using the inverse regression curve. The author defines the dimension reduction under the regression model:

$$Y = f(\boldsymbol{\beta}_1^{\mathsf{T}}\boldsymbol{X}, \boldsymbol{\beta}_2^{\mathsf{T}}\boldsymbol{X}, \ldots, \boldsymbol{\beta}_d^{\mathsf{T}}\boldsymbol{X}, \boldsymbol{\epsilon}) \tag{2.3}$$

Remark 2.4 *The work by Li (1991) was developed before the existence of the dimension reduction subspace was proved. This work instead defines an effective dimension reduction (EDR) space and the directions that span the space ($\boldsymbol{\beta}_i$'s) to be the EDR directions.*

Ordinarily in regression we regress Y against $\boldsymbol{X}$ (forward regression), whereas we can instead choose to regress $\boldsymbol{X}$ against Y. By switching the roles of $\boldsymbol{X}$ and Y we are then dealing with p one-dimensional regression problems rather than one p-dimensional regression problem. The inverse regression curve is given by $\mathrm{E}[\boldsymbol{X}|Y]$ as Y varies. The centre of the inverse regression curve is located at $\mathrm{E}[\boldsymbol{X}|Y] = \mathrm{E}[\boldsymbol{X}]$ and therefore the centred inverse regression curve is given by $\mathrm{E}[\boldsymbol{X}|Y] - \mathrm{E}[\boldsymbol{X}]$. The following theorem is the basis for the SIR methodology.

Theorem 2.5 *Under assumption 2.2 and model* (2.3)*, the centred inverse regression curve* $\mathrm{E}[\boldsymbol{X}|Y] - \mathrm{E}[\boldsymbol{X}]$ *is contained in the linear subspace spanned by* $\boldsymbol{\beta}_i^{\mathsf{T}}\boldsymbol{\Sigma}$*, for* $i = 1, \ldots, d$ *and* $\boldsymbol{\Sigma}$ *is the covariance of* $\boldsymbol{X}$*.*

This is also true for standardised $\boldsymbol{X}$ denoted by $\boldsymbol{Z}$.

Corollary 2.6 *Under assumption 2.2 and model* (2.3)*, the centred inverse regression curve* $\mathrm{E}[\boldsymbol{Z}|Y]$ *is contained in the linear subspace spanned by* η_i*, where* $i = 1, \ldots, d$ *and the* η_i*'s are the standardised EDR directions.*

Consequently, the covariance matrix $\mathrm{cov}[\mathrm{E}[\boldsymbol{Z}|Y]]$ is degenerate in any direction orthogonal to the $\boldsymbol{\eta}_i$'s. This is because an eigenvalue decomposition of a covariance matrix gives the variation of the space. There will be no variation in any directions orthogonal to $\boldsymbol{\eta}_i$'s since $\mathrm{E}[\boldsymbol{Z}|Y]$ is contained within the space spanned by the $\boldsymbol{\eta}_i$'s. Therefore, the d eigenvectors of $\mathrm{cov}[\mathrm{E}[\boldsymbol{Z}|Y]]$, corresponding to the d largest eigenvalues, are the standardised EDR directions.

To align the notation and terminology with later work, the above can be rewritten as follows.

Corollary 2.7 *Under assumption 2.2 and model* (2.3)*, the centred inverse regression curve* $\mathrm{E}[\boldsymbol{X}|Y] \in S_{Y|\boldsymbol{Z}}$*. Consequently, the column space of* $\mathrm{cov}[\mathrm{E}[\boldsymbol{Z}|X]] \in S_{Y|\boldsymbol{Z}}$*.*

The general method for the data $(Y_i, \boldsymbol{X}_i)$ $i = 1, \ldots, n$, as described in Li (1991)), is as follows:

1. Standardise $\boldsymbol{X}$ by an affine transformation to get

$$\boldsymbol{Z} = \hat{\boldsymbol{\Sigma}}^{-1/2}(\boldsymbol{X}_i - \bar{\boldsymbol{X}}) \quad i = 1, \ldots, n,$$

   where $\bar{\boldsymbol{X}}$ and $\hat{\boldsymbol{\Sigma}}$ are the sample mean and sample variance of $\boldsymbol{X}$, respectively.

2. Divide the range of Y into h slices, $A_1, \ldots, A_h$; let the proportion of the Y_i that falls in slice r be $\hat{p}_r$, where $r = 1, \ldots, h$; that is

$$\hat{p}_r = \frac{1}{n}\sum_{i=1}^{n} \delta_r(Y_i) \qquad \delta_r(Y_i) = \begin{cases} 1, & Y_i \in A_r \\ 0, & \text{otherwise} \end{cases}$$

3. Within each slice, compute the sample mean of the $\boldsymbol{Z}_i$'s denoted by $\hat{m}_r$, $r = 1, \ldots, h$, so that

$$\hat{m}_r = \frac{1}{n\hat{p}_r}\sum_{i=1}^{n} \boldsymbol{Z}_i \delta_r(Y_i).$$

4. Conduct a (weighted) principal component analysis for the data $\hat{m}_r$, $r = 1, \ldots, h$ in the following way: Form the weighted covariance matrix

$$\hat{V} = \sum_{r=1}^{h} \hat{p}_j \hat{m}_r \hat{m}_r^{\mathsf{T}}$$

   then find the eigenvalues and eigenvectors for $\hat{V}$

5. Let the d largest eigenvectors (row vectors) be the standardised directions, denoted by $\hat{\eta}_k$, $k = 1, \ldots, d$. Therefore the sufficient dimension reduction directions are

$$\hat{\boldsymbol{\beta}}_k = \hat{\eta}_k \hat{\boldsymbol{\Sigma}}^{-1/2}.$$

Even though SIR is the first method of its kind it still yields faster results when compared with many modern methods. However, it is not generally as accurate as newer methods, described below.

2.1.2 Sliced average variance estimate (SAVE)

In a comment following Li (1991), Cook and Weisberg (1991) considered the case where SIR fails when $\mathrm{E}[\boldsymbol{Z}|Y] = 0$ for all Y. Since $\mathrm{var}[\boldsymbol{Z}|Y]$ does differ from slice to slice, this led them to the conclusion that the dimension reduction directions can be determined by using the second or higher moments.

Remark 2.8 *This work was also developed before the existence of the dimension reduction subspace was proved. It has been shown that the SAVE method can also be used to estimate the CDRS. Therefore, instead of defining the below using the EDR space we will use the CDRS for consistency.*

Theorem 2.9 *Under assumption 2.2, 2.3 and model* (2.3)*, the column space of* $(I - \text{var}[\boldsymbol{Z}|Y])^2$ *is a subspace of* $S_{Y|\boldsymbol{Z}}$.

Therefore following a similar method as SIR, the eigenvalues corresponding to the d largest eigenvectors of

$$\sum_r (I - \text{var}[\boldsymbol{Z}|Y \in I_r])^2$$

span the CDRS.

This method has similar advantages and disadvantages as SIR when comparing speed and accuracy. The additional CCV assumption required for SAVE is more restrictive than the LCM alone. This makes SIR a more desirable choice when $\text{E}[\boldsymbol{Z}|Y] \neq 0$. This does not negate the value of the work but does highlight the limitations compared with other methods.

2.1.3 Principal Hessian directions (pHd)

Further study of SIR indicated that the inverse regression curve was always degenerate when g, as defined in (2.3), is symmetric about $\boldsymbol{X}$. In addition to SAVE, pHd offered an alternative remedy for this limitation.

The Hessian matrix $\boldsymbol{H}$ is a square matrix given by the second order partial derivatives of a function. We denote the average hessian matrix as $\text{E}[\boldsymbol{H}] = \hat{\boldsymbol{H}}$. It was determined that the Hessian matrix, and consequently the average hessian matrix, of $\text{E}[Y|\boldsymbol{X}]$ will be degenerate in any directions orthogonal to the CDRS. We define the eigenvectors of $\hat{\boldsymbol{H}}$ to be the principal hessian directions (pHd's).

Theorem 2.10 *Under assumption 2.2, 2.3 and model* (2.3)*, the rank of the average Hessian matrix,* $\boldsymbol{H}_{\boldsymbol{x}}$*, is at most* d*. Moreover, the pHd's corresponding to the non-zero eigenvalues span the CDRS.*

Corollary 2.11 *Using Stein's Lemma, when* $\boldsymbol{X}$ *is normal, the average Hessian matrix* $\bar{\boldsymbol{H}}_{\boldsymbol{x}}$ *is related to the covariance matrix*

$$\boldsymbol{\Sigma}_{y\boldsymbol{x}\boldsymbol{x}} = \text{E}[(Y - \mu_y)(\boldsymbol{X} - \mu_{\boldsymbol{x}})(\boldsymbol{X} - \mu_{\boldsymbol{x}})^{\mathsf{T}}]$$

through the identity

$$\bar{\boldsymbol{H}}_{\boldsymbol{x}} = \boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1}\boldsymbol{\Sigma}_{y\boldsymbol{x}\boldsymbol{x}}\boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1},$$

where $\boldsymbol{\Sigma_x}$ *denotes the covariance of* $\boldsymbol{X}$ *and* μ_y *and* $\mu_{\boldsymbol{x}}$ *are the means of* Y *and* $\boldsymbol{X}$ *respectively.*

Using this the author then found that when $\boldsymbol{X}$ is normal, the dimension reduction directions can be found by obtaining the eigenvectors for the eigenvalue decomposition of $\boldsymbol{\Sigma_{yxx}}$ with respect to $\boldsymbol{\Sigma_x}$. Let $\bar{\boldsymbol{X}}$ and $\hat{\boldsymbol{\Sigma}}$ denote the sample mean and sample variance of $\boldsymbol{X}$, respectively. The general method for the data $(Y_i, \boldsymbol{X}_i)$ $i = 1, \ldots, n$, given in Li (1992), is as follows:

1. Form the matrix

$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{yxx}} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \bar{Y})(\boldsymbol{X}_i - \bar{\boldsymbol{X}})(\boldsymbol{X}_i - \bar{\boldsymbol{X}})^{\mathsf{T}}.$$

2. Conduct an eigenvalue decomposition of $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{yxx}}$ with respect to $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}}$:

$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{yxx}}\hat{\boldsymbol{\beta}}_{y_j} = \hat{\lambda}_{y_j}\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}}\hat{\boldsymbol{\beta}}_{yj}, \quad j = 1, \ldots, p.$$

2.1.4 Contour regression

As discussed previously, earlier methods were found to be computationally inexpensive bit not always exhaustive of the central dimension reduction space. It was found that SIR failed when the regression curve was symmetric and SAVE and pHd are only exhaustive when normality is assumed which is a restrictive assumption. Instead contour regression was introduced in an aim to predict the central dimension reduction subspace without the additional normality assumption.

Contour regression, Li et al. (2005), aims to predict the central dimension reduction subspace by finding the contour directions (directions along which the response surface is flat) and these direction will span the orthogonal central subspace, $S^{\perp}_{Y|\boldsymbol{X}}$. Two approaches are proposed, SCR (Simple Contour Regression) and GCR (General Contour Regression) which will both be described below.

2.1.4.1 Simple contour regression (SCR)

To define the methodology for simple contour regression (SCR) we first need to define the below assumption.

Assumption 2.12 *For any choice of vectors* $v \in S_{Y|\boldsymbol{X}}$ *and* $w \in S^{\perp}_{Y|\boldsymbol{X}}$ *such that* $\|v\| = \|w\| = 1$, *and some constant* $c > 0$, *we have*

$$\text{var}\left[w^{\mathsf{T}}(\tilde{\boldsymbol{X}} - \boldsymbol{X})||\tilde{Y} - Y| \leq c\right] > \text{var}\left[v^{\mathsf{T}}(\tilde{\boldsymbol{X}} - \boldsymbol{X})||\tilde{Y} - Y| \leq c\right] \tag{2.4}$$

where $(\tilde{\boldsymbol{X}}, \tilde{Y})$ *is an independent copy of* $(\boldsymbol{X}, Y)$.

Now we define the matrix

$$K(c) = \mathrm{E}\left[(\tilde{\boldsymbol{Z}} - \boldsymbol{Z})(\tilde{\boldsymbol{Z}} - \boldsymbol{Z})|\tilde{Y} - Y| \leq c\right]$$

where $\boldsymbol{Z}$ and $\tilde{\boldsymbol{Z}}$ are the standardised versions of $\boldsymbol{X}$ and $\tilde{\boldsymbol{X}}$, respectively.

The following theorem is taken from Li et al. (2005).

Theorem 2.13 *Under assumptions 2.1 and 2.12, the eigenvectors of $K(c)$ corresponding to the d smallest eigenvalues span the central subspace $S_{Y|\boldsymbol{Z}}$.*

The estimation procedure for SCR, as described by Li et al. (2005), is as follows:

1. Compute the sample mean and sample variance matrix of the predictor $\boldsymbol{X}$, denoted $\bar{\boldsymbol{X}}$ and $\hat{\boldsymbol{\Sigma}}$, respectively.

2. Compute the matrix-valued U-statistic:

$$\hat{H}(c) = \frac{1}{\binom{n}{c}} \sum_{(i,j)\in N} (\boldsymbol{X}_j - \boldsymbol{X}_i)(\boldsymbol{X}_j - \boldsymbol{X}_i)^{\mathsf{T}} I(|Y_j - Y_i| \leq c),$$

   where N is the index set $\{(i,j) : i = 2,\ldots,n; j = 1,\ldots,i-1\}$.

3. Compute the spectral decomposition of $\hat{\boldsymbol{\Sigma}}^{-1/2}\hat{H}(c)\hat{\boldsymbol{\Sigma}}^{-1/2}$ and let $\hat{\boldsymbol{\beta}}_{p-d+1},\ldots,\hat{\boldsymbol{\beta}}_p$ be the eigenvectors corresponding to the smallest d eigenvalues.

4. The span of these eigenvectors estimates $S_{Y|\boldsymbol{Z}}$, where $\boldsymbol{Z}$ is the standardised version of $\boldsymbol{X}$. Thus, the estimate of the CDRS is

$$\hat{S}_{Y|\boldsymbol{X}} = \mathrm{span}(\hat{\boldsymbol{\Sigma}}^{-1/2}\boldsymbol{\beta}_{p-d+1},\ldots,\hat{\boldsymbol{\Sigma}}^{-1/2}\boldsymbol{\beta}_p).$$

2.1.4.2 General contour regression (GCR)

SCR uses the inequality $|\tilde{Y} - Y| \leq c$. If the regression function is nonmonotone then this method can be less accurate than other methods since this inequality also picks up other directions. These directions are averaged out which ensures that SCR remains $\sqrt{n}$-exhaustive however it can decrease the efficiency of the method. In an aim to reduce the inefficiencies introduced by using this inequality the author also introduced general contour regression (GCR). Similar to SCR, GCR depends on an additional assumption.

Assumption 2.14 *For any choice of vectors $v \in S_{Y|\boldsymbol{X}}$ and $w \in S^{\perp}_{Y|\boldsymbol{X}}$ such that $\|v\| = \|w\| = 1$, and some constant $c > 0$, we have*

$$\mathrm{var}\left[w^{\mathsf{T}}(\tilde{\boldsymbol{X}} - \boldsymbol{X})|V(\boldsymbol{X},\tilde{\boldsymbol{X}}) \leq c\right] > \mathrm{var}\left[v^{\mathsf{T}}(\tilde{\boldsymbol{X}} - \boldsymbol{X})|V(\boldsymbol{X},\tilde{\boldsymbol{X}}) \leq c\right] \tag{2.5}$$

where $(\tilde{\boldsymbol{X}},\tilde{Y})$ is an independent copy of $(\boldsymbol{X},Y)$.

Once again we define a matrix

$$G(c) = \mathrm{E}\left[(\tilde{\boldsymbol{Z}} - \boldsymbol{Z})(\tilde{\boldsymbol{Z}} - \boldsymbol{Z})|V(\boldsymbol{Z}, \tilde{\boldsymbol{Z}}) \leq c\right]$$

where $\boldsymbol{Z}$ and $\tilde{\boldsymbol{Z}}$ are the standardised versions of $\boldsymbol{X}$ and $\tilde{\boldsymbol{X}}$, respectively.

Theorem 2.15 *Under assumptions 2.1 and 2.14, the eigenvectors of $G(c)$ corresponding to the d smallest eigenvalues span the central subspace $S_{Y|\boldsymbol{Z}}$.*

2.2 Using classification for sufficient dimension reduction

It was shown in Li et al. (2011) that classification methods could be used as a tool for sufficient dimension reduction. We will give an overview of the classifications methods first and then we will discuss how these can be used for dimension reduction.

There are many successful methods with regards to classification, a common method is Vector Machines (SVM), Vapnik (1998). The basic idea behind classification, is to construct a set of hyperplanes to separate the observations, or classify, using the predictor variables. A 'good' separation is defined to be a hyperplane that is of maximum distance from the points closest to the hyperplane vectors of both classes. When creating a classification model, it is common practice to split your data into training data and testing data. The size of the split will depend on the number of samples you begin with, but the size of the training data is usually much larger than the testing data. The model is then built on the training data and tested for inaccuracies on the testing data.

2.2.1 Linear support vectors machines (SVM)

For SVM, the points on the boundary of each class (closest to the separating hyperplane) are called the support vectors. In the linear setting a hyperplane takes the form

$$\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{x} - t = 0,$$

where $\boldsymbol{\psi}$ is the normal to the hyperplane and the parameter t is proportional to the distance of the hyperplane from the origin. Depending on whether the training data is linearly separable or not, determines how the best hyperplane is constructed.

When the data is linearly separable, we use what is called the **hard-margin** approach. Assuming the y_i's take the values 1 and -1, we create two parallel hyperplanes, given by the equations $\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{x} - t = \pm 1$, which separate the data and are of maximum distance from one another. The margin between these hyperplanes is $2/\|\boldsymbol{\psi}\|$ and thus to maximise the distance we need to minimise $\|\boldsymbol{\psi}\|$. Finally, we

need to ensure that all data point remain outside of the margin. Thus the constraint $y_i(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{x_i} - t) \geq 1$ is introduced and our problem becomes

$$\min \boldsymbol{\psi}^\mathsf{T}\boldsymbol{\psi} \qquad \text{subject to } y_i(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{x_i} - t) \geq 1, \text{ for } i = 1, \ldots, n.$$

When the data is not linearly separable, we now have to use what is known as the **soft-margin** approach. In this case the optimisation problem takes the form

$$\begin{aligned} &\text{minimise } \boldsymbol{\psi}^\mathsf{T}\boldsymbol{\psi} + \frac{\lambda}{n}\sum_{i=1}^{n}\xi_i \;\; \text{among } (\boldsymbol{\psi}, t, \boldsymbol{\xi}) \in \mathbb{R}^p \times \mathbb{R} \times \mathbb{R}^n \\ &\text{subject to } Y_i[\boldsymbol{\psi}^\mathsf{T}(\boldsymbol{X}_i - \bar{\boldsymbol{X}}) - t] \geq 1 - \xi_i,\ \xi_i \geq 0,\ i = 1, \ldots, n, \end{aligned} \tag{2.6}$$

where $\lambda > 0$ is a cost parameter and $\boldsymbol{\xi}$ is the vector of ξ_i's. The penalisation vector, $\boldsymbol{\xi}$, takes values $\xi_i = 0$ for correctly classified points and $\xi_i > 0$ for misclassified points. The constraints are equivalent to

$$\xi_i \geq \max\{1 - Y_i\left[\boldsymbol{\psi}^\mathsf{T}(\boldsymbol{X}_i - \bar{\boldsymbol{X}})\right], 0\} = \left(1 - Y_i\left[\boldsymbol{\psi}^\mathsf{T}(\boldsymbol{X}_i - \bar{\boldsymbol{X}})\right]\right)^+. \tag{2.7}$$

Hence the optimisation problem can be rewritten as

$$\boldsymbol{\psi}^\mathsf{T}\boldsymbol{\psi} + \frac{\lambda}{n}\sum_{i=1}^{n}(1 - y_i(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{x_i} - t))^+. \tag{2.8}$$

There have been a number of extensions of the SVM classification technique, including least square SVM, Suykens et al. (2002), and LqSVM first proposed by Burgess and Crisp (1999).

2.2.2 Linear distance-weighted discrimination (DWD)

One of the most interesting variations of SVM was proposed by Marron et al. (2007) and is known as Distance-Weighted Discrimination (DWD). It was recognised that the generalisation performance of SVM in many high dimension low sample size (hdlss) cases was poor. This is due to the fact that SVM suffers from data piling when the dimension of the predictor space is large (see Figure 2.1).

It was predicted that if the dimension is much larger than the sample size then the SVM model is over-fitted to the training data. Therefore, it will be extremely useful for describing the training data but not general enough to be used with other data. We have previously discussed how SVM works by maximising the distance between the support vectors which sit on two orthogonal boundary planes. When the data are projected onto the normal vectors, which are orthogonal to the boundary planes, the support vectors will be projected to one of two common points. With hdlss data, the number of support vectors can be quite large and therefore a large amount of data is projected to the two common points.

To try and prevent this from occurring Marron et al. (2007) chose to estimate the hyperplane that was of maximum distance from each point by optimising the sum of the inverse distances. Using this method, it is clear that the points closest to the hyperplane will influence the direction, more than the points further away, while at the same time the points further away still have some in influence, unlike in SVM.

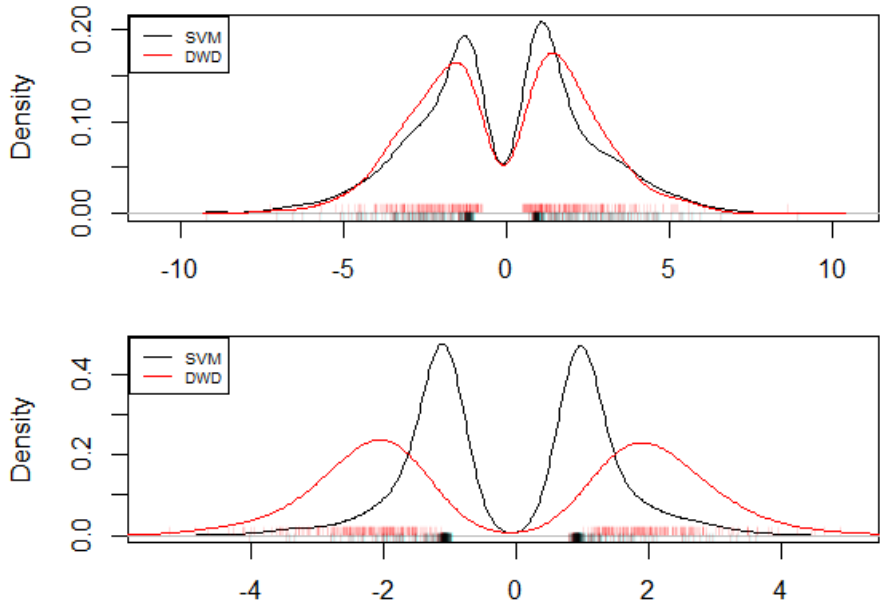

Figure 2.1: Density of projections for $n = 1000$. Top panel: $p = 500$; bottom panel: $p = 1000$. The datasets consists of two classes of points taken from the model $Y = X_1 + \epsilon$.

2.2.2.1 Flexible high-dimensional classification machines and their asymptotic properties (FLAME)

We have already described that a strength of DWD compared with SVM is the lack of data piling. However, DWD is more sensitive to imbalanced data than SVM which was highlighted by Qiao and Zhang (2015). Imbalanced data, in a classification setting, is data that contains more samples in one class than another. The work by Qiao and Zhang (2015) aimed to take advantage of this by constructing a composite function which included the strengths of both DWD and SVM. By first rewriting the DWD loss function as

$$V(u) = \begin{cases} 2\sqrt{\lambda} - \lambda u, & \text{if } u \leq \frac{1}{\sqrt{\lambda}} \\ 1/u, & \text{otherwise} \end{cases} \tag{2.9}$$

and producing a modified hinge loss function

$$H^*(u) = \begin{cases} \sqrt{\lambda} - \lambda u, & \text{if } u \leq \frac{1}{\sqrt{\lambda}} \\ 0, & \text{otherwise} \end{cases}. \tag{2.10}$$

The composite function takes the form

$$L(u) = \left(V(u) - \theta\sqrt{\lambda}\right)^+ = \begin{cases} (2-\theta)\sqrt{\lambda} - \lambda u, & \text{if } u \leq \frac{1}{\sqrt{\lambda}} \\ 1/u - \theta\sqrt{\lambda}, & \text{if } \frac{1}{\sqrt{\lambda}} \leq u < \frac{1}{\theta\sqrt{\lambda}} \\ 0, & \text{otherwise} \end{cases}, \tag{2.11}$$

where $0 \leq \theta \leq 1$.

2.2.2.2 Review of DWD problem

Let $(\boldsymbol{X}_i, Y_i), i = 1, \ldots, n$ be an i.i.d sample of $(\boldsymbol{X}, Y)$. Denote $\bar{\boldsymbol{X}} = n^{-1}\sum_{i=1}^n \boldsymbol{X}_i$ and $\boldsymbol{\Sigma} = \text{var}(\boldsymbol{X})$. Now suppose Y is a binary random variable, which takes values ± 1. DWD is defined by the following optimisation problem:

$$\begin{aligned} &\text{minimise } \sum_{i=1}^n \frac{1}{r_i} + \frac{\lambda}{n}\sum_{i=1}^n \xi_i \;\; \text{among } (\boldsymbol{r}, \boldsymbol{\psi}, t, \boldsymbol{\xi}) \in \mathbb{R}^p \times \mathbb{R} \times \mathbb{R}^n \\ &\text{subject to } r_i = Y_i[\boldsymbol{\psi}^\mathsf{T}(\boldsymbol{X}_i - \bar{\boldsymbol{X}}) - t] + \xi_i \geq 0,\ \xi_i \geq 0,\ i = 1, \ldots, n,\ \|\boldsymbol{\psi}\| \leq 1, \end{aligned} \tag{2.12}$$

where $\boldsymbol{r}$ is a vector of all r_i's and $\boldsymbol{\xi}$ is the vector of all ξ_i's. Here $\lambda > 0$ is a tuning parameter also called the cost (or misclassification penalty) and $\boldsymbol{\xi}$ is a penalisation vector where $\xi_i = 0$ for correctly classified points and $\xi_i > 0$ for misclassified points.

The above optimisation problem can be written slightly differently using the following vector form (for details, see Qiao and Zhang (2015):

$$\begin{aligned} \boldsymbol{\psi}^\mathsf{T}\boldsymbol{\psi} + \sum_{i=1}^n \Bigg[\Bigg[Y_i[\boldsymbol{\psi}^\mathsf{T}(\boldsymbol{X}_i - \bar{\boldsymbol{X}}) - t] + \left(\tfrac{1}{\sqrt{\lambda}} - Y_i[\boldsymbol{\psi}^\mathsf{T}(\boldsymbol{X}_i - \bar{\boldsymbol{X}}) - t]\right)^+ \Bigg]^{-1} \\ + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - Y_i[\boldsymbol{\psi}^\mathsf{T}(\boldsymbol{X}_i - \bar{\boldsymbol{X}}) - t]\right)^+ \Bigg], \end{aligned} \tag{2.13}$$

where the first term comes from the constraint $\|\boldsymbol{\psi}\| \leq 1$ and the rest from replacing ξ_i with the hinge loss $\left(\frac{1}{\sqrt{\lambda}} - Y_i[\boldsymbol{\psi}^\mathsf{T}(\boldsymbol{X}_i - \bar{\boldsymbol{X}}) - t]\right)^+$.

2.2.2.3 Another look at DWD - A novel algorithm

The main aim of the work produced by Wang and Zou (2015) was to produce an alternative algorithm for DWD that would be faster than the second-order-cone programming (SOCP) problem that was proposed by Marron et al. (2007) and develop more work in relation to non-linear DWD. Wang et al. (2016) utilise the majorisation-minimisation (MM) principal whilst developing their computationally superior algorithm, which calculates the solution to the generalised DWD loss function. Before we consider the general DWD loss function we will rewrite the DWD function that we have already considered as:

Definition 2.16 *The loss function for the generalised DWD classifier is written as*

$$\min_{\boldsymbol{\psi},t} \boldsymbol{C}(\boldsymbol{\psi},t) = \min_{\boldsymbol{\psi},t} \left[\frac{1}{n} \sum_{i=1}^{n} V_q(y_i(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{x}_i + t)) + \lambda \boldsymbol{\psi}^\mathsf{T}\boldsymbol{\psi} \right] \tag{2.14}$$

for some λ*, where*

$$V_q(u) = \begin{cases} 1-u & \text{if } u \leq \frac{q}{q+1} \\ \frac{1}{u^q}\frac{q^q}{(q+1)^{q+1}} & \text{if } u > \frac{q}{q+1} \end{cases} \tag{2.15}$$

Now since this function is differentiable everywhere, Wang and Zou (2015) completed all the steps of the MM algorithm to produce the computationally faster DWD algorithm shown below.

1. Initialise $(\tilde{t}, \tilde{\boldsymbol{\psi}})$, the common choice is $\mathbf{0}_{p+1}$.
2. Compute $\boldsymbol{P}^{-1}(\lambda)$:
$$\boldsymbol{P}^{-1}(\lambda) = \begin{pmatrix} n & \mathbf{1}^\mathsf{T}\boldsymbol{X} \\ \boldsymbol{X}^\mathsf{T}\mathbf{1} & \boldsymbol{X}^\mathsf{T}\boldsymbol{X} + \frac{2n\lambda}{M}\boldsymbol{I}_p \end{pmatrix}^{-1}.$$
3. Compute $\boldsymbol{z} = (z_1, \ldots, z_n) : z_i = y_i V_q'(y_i(\tilde{\boldsymbol{\psi}}^\mathsf{T}\boldsymbol{x}_i + \tilde{t}))/n$.
4. Compute:
$$\begin{pmatrix} t \\ \boldsymbol{\psi} \end{pmatrix} = \begin{pmatrix} \tilde{t} \\ \tilde{\boldsymbol{\psi}} \end{pmatrix} - \frac{nq}{(q+1)^2}\boldsymbol{P}^{-1}(\lambda)\begin{pmatrix} \mathbf{1}^\mathsf{T} \\ \boldsymbol{X}^\mathsf{T}\boldsymbol{z} + 2\lambda\tilde{\boldsymbol{\psi}} \end{pmatrix}.$$
5. Set $(\tilde{t}, \tilde{\boldsymbol{\psi}}) = (t, \boldsymbol{\psi})$.
6. Repeat steps 2-5 until convergence condition is met. A commonly used convergence condition is to continue until $\|(\tilde{t}, \tilde{\boldsymbol{\psi}}) - (t, \boldsymbol{\psi})\|_2$ is less than a given tolerance.

Each test that Wang and Zou (2015) performed showed that this algorithm was considerable quicker than the algorithm proposed by Marron et al. (2007).

2.2.3 Linear principal support vector machines (PSVM)

Principal Support Vector Machines (PSVM), Li et al. (2011), proposes a new approach to SDR using SVM as a tool. It was established by Li et al. (2005) that the contours of the regression function span $S_{Y|\boldsymbol{X}}^{\perp}$, the space orthogonal to the central space $S_{Y|\boldsymbol{X}}$. The authors of PSVM proposed that classification methods, more specifically SVM, can be used to estimate the contours of the regression function and therefore estimate the central space. An example showing the relationship between the hyperplanes produced using SVM and the contours of the regression function is given later.

The basic idea behind PSVM is to split $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ into h slices according to the values of the response variable. SVM is then used to to find the optimal hyperplanes to split these slices. Next Principal Component Analysis (PCA) is used on the normal vectors of the hyperplanes. It can be shown that the principal components are an unbiased estimator of the central SDR subspace. More detail of the method is described below.

Consider the regression model

$$Y = f(2X_1 + X_2) + \epsilon. \tag{2.16}$$

Here the central subspace is spanned by $(2, 1)^\mathsf{T}$ and thus the contours for this regression function are defined as the set $\{(x_1, x_2) : 2x_1 + x_2 = c\}$. Our aim is to estimate the contours using the hyperplanes generated from performing SVM or DWD on multiple slices of $\boldsymbol{X}$, corresponding to the values of Y. As described in Li et al. (2011), the normals of these hyperplanes are approximately aligned with the directions that form the central subspace. Therefore, the the principal components of these normals can be used to estimate the central subspace.

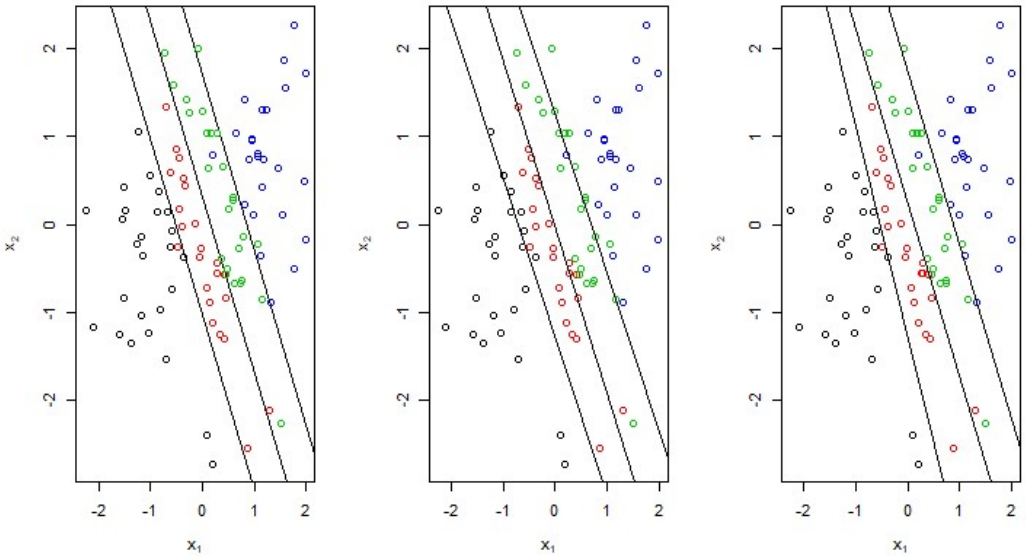

Figure 2.2: Linear contours for model $Y = 2X_1 + X_2 + \epsilon$. Left panel: true contours; centre panel: contours from SVM; right panel: contours from DWD.

To show this, using model (2.16) with f as the identity mapping, we generate 100 replicates. $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{100}$ is then split corresponding to the 25th, 50th and 75th sample quantiles of Y, which are shown by differently coloured dots in Figure 2.2. The centre panel and right panel show the hyperplanes formulated from these slices using SVM and DWD respectively and the left panel shows the true contours. We can see that the contours estimated using both SVM and DWD closely resemble the contours derived directly from the model. Therefore we can determine that the

normals of these hyperplanes gives relatively accurate approximations of the central subspace.

The PSVM objective function is given as:

$$L(\boldsymbol{\psi}, t) = \boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{\psi} + \lambda \mathrm{E}[1 - \tilde{Y}[\tilde{\psi}^{\mathsf{T}}(\boldsymbol{X} - \mathrm{E}[\boldsymbol{X}]) - t]]^{+} \tag{2.17}$$

where $a^{+} = \max\{a, 0\}$, $\boldsymbol{\Sigma} = \mathrm{var}[\boldsymbol{X}]$ and $\tilde{Y}$ is the sliced Y described in (2.2).

Theorem 2.17 *Let $(\boldsymbol{\psi}^*, t^*)$ be the minimisers of (2.17) for all $(\boldsymbol{\psi}, t) \in \mathbb{R}^p \times \mathbb{R}$, then under assumption 2.1, $\boldsymbol{\psi}^* \in S_{Y|\boldsymbol{X}}$.*

In linear PSVM there were two ways proposed to choose the slices; left versus right (LVR) and one versus another (OVA). Generally, one would choose LVR if the response is continuous and OVA if the response is categorical.

2.2.3.1 Estimation algorithm

The estimation algorithm, given by Li et al. (2011), is as follows:

1. Compute the sample mean $\bar{\boldsymbol{X}}$ and sample variance matrix $\hat{\boldsymbol{\Sigma}}$.

2. (LVR) Let q_r, $r = 1, \ldots h-1$, be $h-1$ points that will be used to slice the data and let

   $$\tilde{Y}_i^r = I(Y_i > q_r) - I(Y_i \leq q_r)$$

   for $i = 1, \ldots, n$. Let $(\hat{\boldsymbol{\psi}}_r, \hat{t}_r)$ be the minimisers of the SVM objective function

   $$\boldsymbol{\psi}^{\mathsf{T}}\hat{\boldsymbol{\Sigma}}\boldsymbol{\psi} + \lambda \mathrm{E}_n \left[1 - \left(\tilde{Y}^r(\boldsymbol{\psi}^{\mathsf{T}}(\boldsymbol{X} - \bar{\boldsymbol{X}}) - t)\right)^{+}\right].$$

   (OVA) Apply SVM to each pair of slices from the h slices. More specifically, let $q_0 = \min\{Y_1, \ldots, Y_h\}$ and $q_h = \max\{Y_1, \ldots, Y_h\}$. Then for each (r, s) such that $1 \leq r < s \leq h$, let

   $$\tilde{Y}_i^{rs} = I(q_{s-1} < Y_i \leq q_s) - I(q_{r-1} < Y_i \leq q_r).$$

   Let $(\hat{\boldsymbol{\psi}}_{rs}, \hat{t}_{rs})$ be the minimisers of the SVM objective function

   $$\boldsymbol{\psi}^{\mathsf{T}}\hat{\boldsymbol{\Sigma}}\boldsymbol{\psi} + \lambda \mathrm{E}_n \left[1 - \left(\tilde{Y}^{rs}(\boldsymbol{\psi}^{\mathsf{T}}(\boldsymbol{X} - \bar{\boldsymbol{X}}) - t)\right)^{+}\right].$$

3. Let $\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d$ be the d leading eigenvectors of one of the matrices

   $$\hat{\boldsymbol{M}}_n = \sum_{r=1}^{h-1} \hat{\boldsymbol{\psi}}_r \hat{\boldsymbol{\psi}}_r^{\mathsf{T}} \quad \text{or} \quad \hat{\boldsymbol{M}}_n = \sum_{r=1}^{h-1} \sum_{s=r+1}^{h} \hat{\boldsymbol{\psi}}_{rs} \hat{\boldsymbol{\psi}}_{rs}^{\mathsf{T}}.$$

   We can now estimate $S_{Y|\boldsymbol{X}}$ using the subspace spanned by $\hat{\boldsymbol{v}} = (\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d)$.

Similar to SVM in a classification setting, there are many extensions of PSVM using the extensions of SVM. Principal L_qSVM (PL_qSVM) was introduced by Artemiou and Dong (2016), which investigates for $q > 1$. When $q > 1$ the objective function for L_qSVM becomes strictly convex and ensures the uniqueness of the solution which was highlighted by the author. Another extension, developed by Artemiou et al. (2020), named principal least square SVM (PLSSVM) substitutes the classic SVM objective function with the least square SVM function. This leads to an explicit function rather than an optimisation problem. Others include Zhou and Zhu (2016) who used a minimax variation for sparse SDR, Shin and Artemiou (2017) replaced the hinge loss with a logistic loss to achieve the desired result, Shin et al. (2017) used weighted SVM approach for binary responses and both Artemiou and Shu (2014) and Smallman and Artemiou (2017) focused on removing the bias due to imbalance.

2.3 Linear principal distance-weighted discrimination(PDWD)

The results in this section also appeared in Randall et al. (2020). Our aim is to investigate whether DWD has similar advantages over SVM in the SDR framework, as the ones it has in the classification framework. We will create a similar method as the one in Li et al. (2011) with the difference that the objective function of DWD will replace the objective function of SVM. We call our method Principal DWD (PDWD) following a similar pattern to Li et al. (2011) calling their method Principal SVM. Interestingly, results show that actually DWD works better than SVM for low-dimensional problems and as the dimension increases two methods converge. Thus, data piling seems to help the dimension reduction framework in the regression setting. This observation may be explained due to the fact that in the regression setting we are more interested in a hyperplane alignment than reducing misclassification error. Therefore, data piling may help "stabilise" the alignment of the hyperplane on the correct direction for PSVM.

2.3.1 Linear sufficient dimension reduction using distance-weighted discrimination (DWD)

In the dimension reduction framework, we are interested to work with the population version of the DWD objective function. The version of the DWD population function,

first introduced in Qiao and Zhang (2015), is as follows:

$$\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\psi} + \mathrm{E}\Bigg[\Big[Y[\boldsymbol{\psi}^{\mathsf{T}}(\boldsymbol{X} - \mathrm{E}[\boldsymbol{X}]) - t] + \Big(\tfrac{1}{\sqrt{\lambda}} - Y[\boldsymbol{\psi}^{\mathsf{T}}(\boldsymbol{X} - \mathrm{E}[\boldsymbol{X}]) - t]\Big)^{+}\Big]^{-1} + \lambda\Big(\tfrac{1}{\sqrt{\lambda}} - Y[\boldsymbol{\psi}^{\mathsf{T}}(\boldsymbol{X} - \mathrm{E}[\boldsymbol{X}]) - t]\Big)^{+}\Bigg]. \tag{2.18}$$

There were a number of extensions of the DWD algorithm. Some include the weighted DWD approach by Qiao et al. (2010) and the sparse DWD approach by Wang and Zou (2016). Marron et al. (2007) as well as the extensions discussed above used cone programming to solve the optimisation problem in (2.12) (or the respective one for each extension). More recently, Wang and Zou (2015) proposed the generalised DWD algorithm which allows for faster computational calculations. In this work, we utilise their idea and thus our estimation algorithm is much faster than previous methodology in the SVM-based SDR framework.

Analogous to PSVM there are two ways one can choose the slices; left versus right (LVR) and one versus another (OVA). Replacing Y in the population objective function of DWD with $\tilde{Y}$ defined in (2.2), we get the following objective function in the SDR framework:

$$L(\boldsymbol{\psi}, t) = \mathrm{E}\Bigg[\Big[\tilde{Y}[\boldsymbol{\psi}^{\mathsf{T}}(\boldsymbol{X} - \mathrm{E}[\boldsymbol{X}]) - t] + \Big(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}[\boldsymbol{\psi}^{\mathsf{T}}(\boldsymbol{X} - \mathrm{E}[\boldsymbol{X}]) - t]\Big)^{+}\Big]^{-1} + \lambda\Big(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}[\boldsymbol{\psi}^{\mathsf{T}}(\boldsymbol{X} - \mathrm{E}[\boldsymbol{X}]) - t]\Big)^{+}\Bigg] + \boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{\psi}. \tag{2.19}$$

Following Li et al. (2011) we note that we have also inserted $\boldsymbol{\Sigma}$ into the first term to ensure the resulting DWD estimate is unbiased and to provide the unified framework for non-linear SDR. Assuming $\mathrm{E}[\boldsymbol{X}] = 0$ without loss of generality, and by setting $u = \tilde{Y}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X} - t]$ we can simplify the above objective function to:

$$\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{\psi} + \mathrm{E}\left[\Big[u + \Big(\tfrac{1}{\sqrt{\lambda}} - u\Big)^{+}\Big]^{-1} + \lambda\Big(\tfrac{1}{\sqrt{\lambda}} - u\Big)^{+}\right]. \tag{2.20}$$

The following Lemma is used to prove the convexity of the objective function. This Lemma will be crucial in proving the theorem which shows that the normal vector $\boldsymbol{\psi}$ of the optimal hyperplane, developed by the PDWD, is indeed in the CS.

Lemma 2.18 *If* $f(u) = \Big[u + \Big(\tfrac{1}{\sqrt{\lambda}} - u\Big)^{+}\Big]^{-1} + \lambda\Big(\tfrac{1}{\sqrt{\lambda}} - u\Big)^{+}$, *then* f *is convex for all* $\lambda > 0$.

Proof. To prove convexity, we need to show

$$f(\alpha u_1 + (1-\alpha)u_2) \leq \alpha f(u_1) + (1-\alpha) f(u_2)$$

for all $u \in \mathbb{R}$ and $\alpha \in [0,1]$. Firstly, we can rewrite f as

$$f(u) = \begin{cases} \frac{1}{u} & u \geq \frac{1}{\sqrt{\lambda}} \\ 2\sqrt{\lambda} - \lambda u & u < \frac{1}{\sqrt{\lambda}} \end{cases}.$$

For $u \geq \frac{1}{\sqrt{\lambda}}$ we have $2\sqrt{\lambda} - \lambda u \leq \frac{1}{u}$ and for $u_1 \leq u_2$ we have $f(u_1) \geq f(u_2)$ since f is a decreasing function. We need to consider three cases:

(i) When $u_1 < \frac{1}{\sqrt{\lambda}}$ and $u_2 < \frac{1}{\sqrt{\lambda}}$ we have $\alpha u_1 + (1-\alpha)u_2 < \frac{1}{\sqrt{\lambda}}$, hence

$$\begin{aligned} f(\alpha u_1 + (1-\alpha)u_2) &= 2\sqrt{\lambda} - \lambda(\alpha u_1 + (1-\alpha)u_2) \\ &= \alpha(2\sqrt{\lambda} - \lambda u_1) + (1-\alpha)(2\sqrt{\lambda} - \lambda u_2) \\ &= \alpha f(u_1) + (1-\alpha) f(u_2) \end{aligned}$$

(ii) Since the gradient of f is equal when approaching from the left and right of $\frac{1}{\sqrt{\lambda}}$ when $u_1 < \frac{1}{\sqrt{\lambda}}$ and $u_2 \geq \frac{1}{\sqrt{\lambda}}$ we can assume without loss of generality that $\alpha u_1 + (1-\alpha)u_2 < \frac{1}{\sqrt{\lambda}}$ and so

$$\begin{aligned} f(\alpha u_1 + (1-\alpha)u_2) &= 2\sqrt{\lambda} - \lambda(\alpha u_1 + (1-\alpha)u_2) \\ &= \alpha(2\sqrt{\lambda} - \lambda u_1) + (1-\alpha)(2\sqrt{\lambda} - \lambda u_2) \\ &\leq \alpha(2\sqrt{\lambda} - \lambda u_1) + \frac{(1-\alpha)}{u_2} \\ &= \alpha f(u_1) + (1-\alpha) f(u_2) \end{aligned}$$

(iii) When $u_1 \geq \frac{1}{\sqrt{\lambda}}$ and $u_2 \geq \frac{1}{\sqrt{\lambda}}$ we have $\alpha u_1 + (1-\alpha)u_2 \geq \frac{1}{\sqrt{\lambda}}$. In this case we can simply prove that the second derivative of $f(u) = \frac{1}{u}$ only gives positive values as follows

$$f''(u) = \frac{2}{u^3} > 0 \ \text{ since } \lambda > 0.$$

Hence we have

$$f(\alpha u_1 + (1-\alpha)u_2) \leq \alpha f(u_1) + (1-\alpha) f(u_2)$$

for all $u \in \mathbb{R}$, and therefore f is convex. □

Having verified the convexity of the objective function then one can prove the following theorem which demonstrates that the normal vector of the hyperplane is in $\mathcal{S}_{Y|\boldsymbol{X}}$. This follows directly from the proof in Li et al. (2011) due to the fact that the hinge loss in SVM is replaced with another convex function and as Li et al. (2011) claim their proof holds for every convex function.

Theorem 2.19 *If* $\mathrm{E}(\boldsymbol{X}|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X})$ *is a linear function of* $\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}$*, where* $\boldsymbol{\beta}$ *is defined as in* (2.1) *and if* $(\boldsymbol{\psi}^*, t^*)$ *minimises the objective function* (2.19) *among all* $(\boldsymbol{\psi}, t) \in \mathbb{R}^p \times \mathbb{R}$*, then* $\boldsymbol{\psi}^* \in \mathcal{S}_{Y|\boldsymbol{X}}$.

Proof. It is important to note that under the conditions of the theorem we can write the conditional expectation

$$\mathrm{E}[\boldsymbol{X}|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}] = \boldsymbol{P}_{\boldsymbol{\beta}}^{\mathsf{T}}(\boldsymbol{\Sigma})\boldsymbol{X},$$

where $\boldsymbol{P}_{\boldsymbol{\beta}}(\boldsymbol{\Sigma})$ is the projection matrix $\boldsymbol{\beta}(\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{\beta})^{-1}\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{\Sigma}$.
Our objective function then takes the form

$$L(\boldsymbol{\psi}, t) = \boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{\psi} + \mathrm{E}\left[\left[u + \left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+}\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+}\right].$$

Beginning with the first term we have

$$\begin{aligned}
\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{\psi} &= \mathrm{var}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X}] \\
&= \mathrm{var}[\mathrm{E}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X}|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}]] + \mathrm{E}[\mathrm{var}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X}|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}]] \\
&\geq \mathrm{var}[\mathrm{E}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X}|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}]] \\
&= \mathrm{var}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{P}_{\boldsymbol{\beta}}^{\mathsf{T}}\boldsymbol{X}] \\
&= (\boldsymbol{P}_{\boldsymbol{\beta}}(\boldsymbol{\Sigma})\boldsymbol{\psi})^{\mathsf{T}}\boldsymbol{\Sigma}(\boldsymbol{P}_{\boldsymbol{\beta}}(\boldsymbol{\Sigma})\boldsymbol{\psi}).
\end{aligned}$$

Hence

$$\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{\psi} \geq (\boldsymbol{P}_{\boldsymbol{\beta}}\boldsymbol{\psi})^{\mathsf{T}}\boldsymbol{\Sigma}(\boldsymbol{P}_{\boldsymbol{\beta}}\boldsymbol{\psi}). \tag{2.21}$$

Now let us look at the second term. Again, we can write

$$\begin{aligned}
&\mathrm{E}\left[\left[u + \left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+}\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+}\right] \\
&= \mathrm{E}\left[\mathrm{E}\left[\left[u + \left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+}\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+} \middle| \tilde{Y}, \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}\right]\right].
\end{aligned}$$

If we define the function f such that $f(a) = \left[a + \left(\tfrac{1}{\sqrt{\lambda}} - a\right)^{+}\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - a\right)^{+}$ then this gives

$$\mathrm{E}\left[\left[u + \left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+}\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+} \middle| \tilde{Y}, \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}\right] = \mathrm{E}[f(u)|\tilde{Y}, \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}].$$

Since f is a convex function, we can use Jenson's inequality as follows:

$$\begin{aligned}
\mathrm{E}[f(u)|\tilde{Y}, \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}] &\geq \left[\mathrm{E}[u|\tilde{Y}, \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}] + \left(\tfrac{1}{\sqrt{\lambda}} - \mathrm{E}[u|\tilde{Y}, \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}]\right)^{+}\right]^{-1} \\
&\qquad + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \mathrm{E}[u|\tilde{Y}, \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}]\right)^{+} \\
&= \left[\tilde{Y}(\mathrm{E}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X}|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}] - t) + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\mathrm{E}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X}|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}] - t)\right)^{+}\right]^{-1} \\
&\qquad + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\mathrm{E}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X}|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}] - t)\right)^{+} \\
&= \left[\tilde{Y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{P}_{\boldsymbol{\beta}}^{\mathsf{T}}(\boldsymbol{\Sigma})\boldsymbol{X} - t) + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{P}_{\boldsymbol{\beta}}^{\mathsf{T}}(\boldsymbol{\Sigma})\boldsymbol{X} - t)\right)^{+}\right]^{-1} \\
&\qquad + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{P}_{\boldsymbol{\beta}}^{\mathsf{T}}(\boldsymbol{\Sigma})\boldsymbol{X} - t)\right)^{+}.
\end{aligned}$$

Thus combing this with (2.21) we get

$$L(\boldsymbol{\psi}, t) \geq L(\boldsymbol{P}_{\boldsymbol{\beta}}(\boldsymbol{\Sigma})\boldsymbol{\psi}, t). \tag{2.22}$$

If $\boldsymbol{\psi}$ does not belong to $\mathcal{S}_{Y|\boldsymbol{X}}$, then $\text{var}[\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X}|\boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{X}] > 0$ and the inequality in (2.21) becomes strict. Hence the inequality in (2.22) is strict. Therefore, such $\boldsymbol{\psi}$ cannot be the minimiser of $L(\boldsymbol{\psi}, t)$. □

2.3.2 Sample estimation algorithm

Having established the theoretical properties of the minimiser of the objective function in PDWD we now investigate the sample estimation algorithm of our method. Before giving the algorithm though we look at available packages in solving the optimisation problem of DWD. As the available packages solve the objective function of DWD which does not include $\boldsymbol{\Sigma}$ in the first term, we demonstrate below that by standardising the data the objective function of PDWD becomes equivalent to the objective function of DWD and therefore available packages can be used.

As was mentioned above the objective function of DWD

$$\begin{aligned}\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\psi} + \mathrm{E}\Bigg[\Bigg[\tilde{Y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X} - t) + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X} - t)\right)^{+}\Bigg]^{-1} \\ + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X} - t)\right)^{+}\Bigg]\end{aligned} \tag{2.23}$$

and the one for PDWD is

$$\begin{aligned}\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\Sigma}^{\mathsf{T}}\boldsymbol{\psi} + \mathrm{E}\Bigg[\Bigg[\tilde{Y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X} - t) + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X} - t)\right)^{+}\Bigg]^{-1} \\ + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{X} - t)\right)^{+}\Bigg].\end{aligned} \tag{2.24}$$

Now if we let $\boldsymbol{\zeta} = \boldsymbol{\Sigma}^{1/2}\boldsymbol{\psi}$ and $\boldsymbol{Z} = \boldsymbol{\Sigma}^{-1/2}\left(\boldsymbol{X} - \bar{\boldsymbol{X}}\right)$, and substitute these into (2.24) we have

$$\begin{aligned}\boldsymbol{\zeta}^{\mathsf{T}}\boldsymbol{\zeta} + \mathrm{E}\Bigg[\Bigg[\tilde{Y}(\boldsymbol{\zeta}^{\mathsf{T}}\boldsymbol{Z} - t) + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\boldsymbol{\zeta}^{\mathsf{T}}\boldsymbol{Z} - t)\right)^{+}\Bigg]^{-1} \\ + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\boldsymbol{\zeta}^{\mathsf{T}}\boldsymbol{Z} - t)\right)^{+}\Bigg],\end{aligned} \tag{2.25}$$

which we can see is of the same form as (2.23). Hence, as we stated above, we can see that standardising $\boldsymbol{X}$ modifies the PDWD in a desired way. We emphasise here that this fact allows us to use existing algorithms for DWD in the literature to estimate the PDWD solution. Hence, in our algorithm below we require the standardisation of the data.

The estimation procedure is as follows:

1. Compute the sample mean $\bar{\boldsymbol{X}}$ and sample variance matrix $\hat{\boldsymbol{\Sigma}}$.

2. We find the minimiser using the algorithm in Wang and Zou (2015). In more detail:
   (LVR) Let q_r, $r = 1, \dots h-1$, be $h-1$ dividing points and let

$$\tilde{Y}_i^r = I(Y_i > q_r) - I(Y_i \leq q_r)$$

   for $i = 1, \dots, n$. Then using DWD, let $(\hat{\boldsymbol{\psi}}_r, \hat{t}_r)$ be the minimisers of

$$\boldsymbol{\psi}^\mathsf{T}\hat{\boldsymbol{\Sigma}}\boldsymbol{\psi} + \mathrm{E}_n\left[\left[\tilde{Y}^r(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{X} - t) + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}^r(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{X} - t)\right)^+\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}^r(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{X} - t)\right)^+\right].$$

   (OVA) Apply DWD to each pair of slices from the h slices. More specifically, let $q_0 = \min\{Y_1, \dots, Y_h\}$ and $q_h = \max\{Y_1, \dots, Y_h\}$. Then for each (r, s) such that $1 \leq r < s \leq h$, let

$$\tilde{Y}_i^{rs} = I(q_{s-1} < Y_i \leq q_s) - I(q_{r-1} < Y_i \leq q_r).$$

   Let $(\hat{\boldsymbol{\psi}}_{rs}, \hat{t}_{rs})$ be the minimisers of

$$\boldsymbol{\psi}^\mathsf{T}\hat{\boldsymbol{\Sigma}}\boldsymbol{\psi} + \mathrm{E}_n\left[\left[\tilde{Y}^{rs}(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{X} - t) + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}^{rs}(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{X} - t)\right)^+\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}^{rs}(\boldsymbol{\psi}^\mathsf{T}\boldsymbol{X} - t)\right)^+\right].$$

3. Let $\hat{\boldsymbol{v}}_1, \dots, \hat{\boldsymbol{v}}_d$ be the d leading eigenvectors of one of the matrices

$$\hat{\boldsymbol{M}}_n = \sum_{r=1}^{h-1} \hat{\boldsymbol{\psi}}_r \hat{\boldsymbol{\psi}}_r^\mathsf{T} \quad \text{or} \quad \hat{\boldsymbol{M}}_n = \sum_{r=1}^{h-1} \sum_{s=r+1}^{h} \hat{\boldsymbol{\psi}}_{rs} \hat{\boldsymbol{\psi}}_r s^\mathsf{T}. \tag{2.26}$$

   We can now estimate $S_{Y|\boldsymbol{X}}$ using the subspace spanned by $\hat{\boldsymbol{v}} = (\hat{\boldsymbol{v}}_1, \dots, \hat{\boldsymbol{v}}_d)$.

2.3.3 Asymptotic analysis of linear principal distance weighted discrimination (PDWD)

In this section we discuss the asymptotic properties of PDWD. We find the Hessian matrix and the influence function before proving consistency. We demonstrate the consistency when p is fixed, as well as when p is not fixed and tends to infinity, although we still require it to be less than n. To make the proofs easier to read we use the following notation. Let $\boldsymbol{\theta} = (\boldsymbol{\psi}^\mathsf{T}, t)^\mathsf{T}$, $\boldsymbol{Z} = (\boldsymbol{X}^\mathsf{T}, \tilde{Y})^\mathsf{T}$, $\boldsymbol{X}^* = (\boldsymbol{X}^\mathsf{T}, -1)^\mathsf{T}$ and $\boldsymbol{\Sigma}^* = \mathrm{diag}(\boldsymbol{\Sigma}, 0)$, then $u = \boldsymbol{\theta}^\mathsf{T}\boldsymbol{X}^*\tilde{Y}$ and thus

$$\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{\psi} + \left[u + \left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+}\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - u\right)^{+}$$
$$= \boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{\Sigma}^{*}\boldsymbol{\theta} + \left[\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{X}^{*}\tilde{Y} + \left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{X}^{*}\tilde{Y}\right)^{+}\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{X}^{*}\tilde{Y}\right)^{+}.$$

We denote this function by $m(\boldsymbol{\theta}, \boldsymbol{Z})$. Let $\Omega_{\boldsymbol{Z}}$ be the support of $\boldsymbol{Z}$ and let $\boldsymbol{h} : \Theta \times \Omega_{\boldsymbol{Z}} \to \mathbb{R}^{+}$ be a function of $(\boldsymbol{\theta}, \boldsymbol{Z})$. Let $D_{\boldsymbol{\theta}}$ denote the $(p+1)$-dimensional column vector of differential operators $(\partial/\partial\theta_1, \ldots, \partial/\partial\theta_{p+1})^{\mathsf{T}}$.

Before we consider the gradient of the DWD objective function, we prove that the function f is differentiable at all points.

Lemma 2.20 *The function f, as defined in Lemma 2.18, is differentiable at all points.*

Proof. We need to prove that the gradient of f as we approach $\frac{1}{\sqrt{\lambda}}$ from below is equal to the gradient as we approach from the above. We have

$$f'(a) = -\left[a + \left(\tfrac{1}{\sqrt{\lambda}} - a\right)^{+}\right]^{-2}$$
$$= \begin{cases} -a^{-2} & a \geq \frac{1}{\sqrt{\lambda}} \\ -\lambda & a < \frac{1}{\sqrt{\lambda}} \end{cases}.$$

Hence $\lim_{a\downarrow\frac{1}{\sqrt{\lambda}}} f'(a) = -\lambda = \lim_{a\uparrow\frac{1}{\sqrt{\lambda}}} f'(a)$. Therefore f is differentiable everywhere. □

The next theorem gives the gradient of the DWD objective function $\mathrm{E}[m(\boldsymbol{\theta}, \boldsymbol{Z})]$. The proof follows straight from Lemma 2.20 and is therefore omitted. Let $D^2_{\boldsymbol{\theta}}$ denote the operator $D_{\boldsymbol{\theta}}D^{\mathsf{T}}_{\boldsymbol{\theta}}$. Thus, $D^2_{\boldsymbol{\theta}}m(\boldsymbol{\theta}, \boldsymbol{Z})$ is the $(p+1)\times(p+1)$ matrix whose (i,j)th entry is $\partial^2 m/\partial\boldsymbol{\theta}_i\partial\boldsymbol{\theta}_j$.

Theorem 2.21 *The gradient of $m(\boldsymbol{\theta}, \boldsymbol{z})$ takes the form*

$$D_{\boldsymbol{\theta}}\mathrm{E}[m(\boldsymbol{\theta}, \boldsymbol{z})] = 2\boldsymbol{\Sigma}^{*}\boldsymbol{\theta} - \mathrm{E}\left[\boldsymbol{X}^{*}\tilde{Y}\left[\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{X}^{*}\tilde{Y} + \left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{X}^{*}\tilde{Y}\right)^{+}\right]^{-2}\right]. \tag{2.27}$$

The next step is to find the Hessian matrix. Before doing so, we state some helpful results. First we use the following notation. Let $\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{z}) = D_{\boldsymbol{\theta}}m(\boldsymbol{\theta}, \boldsymbol{z})$ and for each $\boldsymbol{\theta} \in \Theta$, let $N_{\boldsymbol{\theta}}(\boldsymbol{n})$ be the set of $\boldsymbol{X}$ for which a function $\boldsymbol{n}(\boldsymbol{z}, \cdot)$ is not differentiable at $\boldsymbol{\theta}$. That is,

$$N_{\boldsymbol{\theta}}(\boldsymbol{n}) = \{\boldsymbol{z} : D_{\boldsymbol{\theta}}\boldsymbol{n}(\cdot, \boldsymbol{z}) \text{ is not differentiable at } \boldsymbol{\theta}\}.$$

Lemma 2.22 *Suppose that $\boldsymbol{n} : \Theta \times \Omega_{\boldsymbol{Z}} \to \mathbb{R}$ satisfies the following conditions*

1. *(almost surely differentiable) for each* $\boldsymbol{\theta} \in \Theta$, $\mathrm{P}[\boldsymbol{Z} \in N_{\boldsymbol{\theta}}(\boldsymbol{n})] = 0$.

2. *(Lipschitz condition) there is an integrable function* $c(\boldsymbol{z})$, *independent of* $\boldsymbol{\theta}$, *such that for any* $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \Theta$,

$$|\boldsymbol{n}(\boldsymbol{\theta}_2, \boldsymbol{z}) - \boldsymbol{n}(\boldsymbol{\theta}_1, \boldsymbol{z})| \leq c(\boldsymbol{z})\|\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1\|.$$

Then $D_{\boldsymbol{\theta}}[\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{Z})]$ *is integrable,* $\mathrm{E}[D_{\boldsymbol{\theta}}\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{Z})]$ *is differentiable and*

$$D_{\boldsymbol{\theta}}\mathrm{E}[\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{Z})] = \mathrm{E}[D_{\boldsymbol{\theta}}\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{Z})]. \tag{2.28}$$

Lemma 2.23 *For* $c > 0$ *we have the following inequality*

$$\left|\frac{(a + (c-a)^+)^2 - (b + (c-b)^+)^2}{(a + (c-a)^+)^2(b + (c-b)^+)^2}\right| \leq \frac{2}{c^3}|b - a|.$$

Proof. It is clear that the result holds if $a > b$ or if $b > a$ so we will assume $a > b$. To proof this inequality hold we need to consider three case.

When $c \leq a$ and $c \leq b$, we have $a + (c-a)^+ = a$ and $b + (c-b)^+ = b$. Therefore

$$\begin{aligned}\left|\frac{(a + (c-a)^+)^2 - (b + (c-b)^+)^2}{(a + (c-a)^+)^2(b + (c-b)^+)^2}\right| &= \left|\frac{a^2 - b^2}{a^2b^2}\right| < \left|\frac{(a-b)(a+b)}{a^2b^2}\right| \\ &< \frac{|a-b|}{c^2}\left|\frac{a+b}{ab}\right| = \frac{|a-b|}{c^2}\left|\frac{1}{b} + \frac{1}{a}\right| \\ &< \frac{|a-b|}{c^2}\left|\frac{1}{c} + \frac{1}{c}\right| = \frac{2}{c^3}|a-b|.\end{aligned}$$

When $c \leq a$ and $c > b$, we have $a + (c-a)^+ = a$ and $b + (c-b)^+ = c$. Therefore

$$\begin{aligned}\left|\frac{(a + (c-a)^+)^2 - (b + (c-b)^+)^2}{(a + (c-a)^+)^2(b + (c-b)^+)^2}\right| &= \left|\frac{a^2 - c^2}{a^2c^2}\right| < \left|\frac{(a-c)(a+c)}{a^2c^2}\right| \\ &< \frac{|a-c|}{c^2}\left|\frac{a+c}{ac}\right| = \frac{|a-c|}{c^2}\left|\frac{1}{c} + \frac{1}{a}\right| \\ &< \frac{|a-c|}{c^2}\left|\frac{1}{c} + \frac{1}{c}\right| = \frac{2}{c^3}|a-c| \\ &< \frac{2}{c^3}|a-b|.\end{aligned}$$

When $c > a$ and $c > b$, we have $a + (c-a)^+ = c$ and $b + (c-b)^+ = c$. Therefore

$$\left|\frac{(a + (c-a)^+)^2 - (b + (c-b)^+)^2}{(a + (c-a)^+)^2(b + (c-b)^+)^2}\right| = \left|\frac{c^2 - c^2}{c^2c^2}\right| = 0 < \frac{2}{c^3}|a-b|.$$

Since this holds for the three cases we can assume the result is true for and a and b and for any $c > 0$. □

Now we have the necessary results which will be helpful in finding the Hessian matrix as the following theorem states.

Theorem 2.24 *Suppose, for each $\tilde{y} = -1, 1$, the distribution of $\boldsymbol{X}|\tilde{Y} = \tilde{y}$ is dominated by the Lebesgue measure and $\mathrm{E}[\|\boldsymbol{X}\|^2] < \infty$. Then*

$$D_{\boldsymbol{\theta}}\mathrm{E}[\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{Z})] = 2\boldsymbol{\Sigma}^* - \mathrm{E}\left[\boldsymbol{X}^*\boldsymbol{X}^{*\mathsf{T}} I\left(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{X}^*\tilde{Y} < \tfrac{1}{\sqrt{\lambda}}\right)\left[\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{X}^*\tilde{Y}\right]^{-3}\right]. \tag{2.29}$$

Proof. Let $H(\boldsymbol{\psi}, a)$ denote the hyperplane $\{\boldsymbol{x} : \boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{x} = a\}$. We first need to verify the two assumptions in Lemma 2.22. In our case,

$$\mathrm{P}[(\boldsymbol{X}, \tilde{Y}) \in N_{\boldsymbol{\theta}}(n)] = \sum_{\tilde{y}\in\{-1,1\}} \mathrm{P}(\tilde{Y} = \tilde{y})\mathrm{P}\left[\boldsymbol{X} \in H\left(\boldsymbol{\psi}, t + \frac{\tilde{y}}{\sqrt{\lambda}}\right)\middle|\tilde{Y} = \tilde{y}\right].$$

Since the Lebesgue measure of $H\left(\boldsymbol{\psi}, t + \frac{\tilde{y}}{\sqrt{\lambda}}\right)$ is 0 for $\tilde{y} \in \{-1, 1\}$, by assumption 1 of the theorem, the above probability is 0. Thus condition 1 of Lemma 2.22 is satisfied.

Let $\boldsymbol{n}_1(\boldsymbol{\theta}, \boldsymbol{z}) = \boldsymbol{\Sigma}^*\boldsymbol{\theta}$ and $\boldsymbol{n}_2(\boldsymbol{\theta}, \boldsymbol{z}) = \boldsymbol{x}^*\tilde{y}\left[\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}^*\tilde{y} + \left(\frac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-2}$. Then $\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{z}) = 2\boldsymbol{n}_1(\boldsymbol{\theta}, \boldsymbol{z}) + \boldsymbol{n}_2(\boldsymbol{\theta}, \boldsymbol{z})$. Since $\boldsymbol{n}_1$ is non-random and differentiable, it obviously satisfies $\mathrm{E}[D_{\boldsymbol{\theta}}\boldsymbol{n}_1(\boldsymbol{\theta}, \boldsymbol{z})] = D_{\boldsymbol{\theta}}\mathrm{E}[\boldsymbol{n}_1(\boldsymbol{\theta}, \boldsymbol{z})]$. To verify that $\boldsymbol{n}_2$ is Lipschitz, let $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^{p+1}$. Then

$$\begin{aligned}
&\|\boldsymbol{n}_2(\boldsymbol{\theta}_2, \boldsymbol{z}) - \boldsymbol{n}_2(\boldsymbol{\theta}_1, \boldsymbol{z})\| \\
&\quad= \left\|\boldsymbol{x}^*\tilde{y}\left[\boldsymbol{\theta}_2^{\mathsf{T}}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_2^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-2} - \boldsymbol{x}^*\tilde{y}\left[\boldsymbol{\theta}_1^{\mathsf{T}}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_1^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-2}\right\| \\
&\quad\le \|\boldsymbol{x}^*\|\left\|\frac{\left(\boldsymbol{\theta}_1^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}+\left(\frac{1}{\sqrt{\lambda}}-\boldsymbol{\theta}_1^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}\right)^+\right)^2 - \left(\boldsymbol{\theta}_2^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}+\left(\frac{1}{\sqrt{\lambda}}-\boldsymbol{\theta}_2^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}\right)^+\right)^2}{\left(\boldsymbol{\theta}_1^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}+\left(\frac{1}{\sqrt{\lambda}}-\boldsymbol{\theta}_1^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}\right)^+\right)^2\left(\boldsymbol{\theta}_2^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}+\left(\frac{1}{\sqrt{\lambda}}-\boldsymbol{\theta}_2^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}\right)^+\right)^2}\right\|.
\end{aligned}$$

From Lemma 4 we get:

$$\begin{aligned}
\|\boldsymbol{n}_2(\boldsymbol{\theta}_2, \boldsymbol{z}) - \boldsymbol{n}_2(\boldsymbol{\theta}_1, \boldsymbol{z})\| &\le 2\lambda^{3/2}\|\boldsymbol{x}^*\|\|\boldsymbol{\theta}_1^{\mathsf{T}}\boldsymbol{x}^* - \boldsymbol{\theta}_2^{\mathsf{T}}\boldsymbol{x}^*\| \\
&\le 2\lambda^{3/2}(1 + \|\boldsymbol{x}\|^2)\|\boldsymbol{\theta}_1^{\mathsf{T}} - \boldsymbol{\theta}_2^{\mathsf{T}}\|.
\end{aligned}$$

Since $\mathrm{E}[\|\boldsymbol{X}\|^2] < \infty$,

$$\mathrm{E}\left[1 + \|\boldsymbol{X}\|^2\right] = 1 + \mathrm{E}[\|\boldsymbol{X}\|^2] < \infty.$$

This verifies condition 2 of Lemma 3. Finally, by direct calculations we find that, for $\boldsymbol{z} \notin N_{\boldsymbol{\theta}}(\boldsymbol{n})$,

$$\begin{aligned}
D_{\boldsymbol{\psi}}[\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{z})] &= 2\boldsymbol{\Sigma} + 2\boldsymbol{x}^*\boldsymbol{x}^{\mathsf{T}}\mathrm{I}\left(\tilde{y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{x} - t) \ge \tfrac{1}{\sqrt{\lambda}}\right)\left[\tilde{y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{x} - t)\right]^{-3}, \\
D_t[\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{z})] &= -2\boldsymbol{x}^*\mathrm{I}\left(\tilde{y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{x} - t) \ge \tfrac{1}{\sqrt{\lambda}}\right)\left[\tilde{y}(\boldsymbol{\psi}^{\mathsf{T}}\boldsymbol{x} - t)\right]^{-3}.
\end{aligned}$$

Hence

$$D_{\boldsymbol{\theta}}[\boldsymbol{n}(\boldsymbol{\theta}, \boldsymbol{z})] = 2\boldsymbol{\Sigma}^* - 2\boldsymbol{x}^*\boldsymbol{x}^{*\mathsf{T}}\mathrm{I}\left(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}^*\tilde{y} \ge \tfrac{1}{\sqrt{\lambda}}\right)\left[\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}^*\tilde{y}\right]^{-3}.$$

The theorem follows now from Lemma 3. □

The following theorem gives the influence function of PDWD. A similar result in the SVM literature can be found in Jiang et al. (2008).

Theorem 2.25 *Let $\boldsymbol{\theta}_0 = (\boldsymbol{\psi}_0^\mathsf{T}, t_0)^\mathsf{T}$ be the minimiser of $E[m(\boldsymbol{\theta}, \boldsymbol{Z})]$. Suppose, for each $\tilde{y} = -1, 1$, the distribution of $\boldsymbol{X}|\tilde{Y} = \tilde{y}$ is dominated by the Lebesgue measure and $E[\|\boldsymbol{X}\|^2] < \infty$. Then*

$$\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0 = -n^{-1}\boldsymbol{H}^{-1}\sum_{i=1}^{n} \mathrm{B}_i(\boldsymbol{z}) + o_p(n^{-1/2}), \tag{2.30}$$

where $\mathrm{B}_i(\boldsymbol{z}) = 2\boldsymbol{\Sigma}^*\boldsymbol{\theta}_0 - \boldsymbol{x}_i^*\tilde{y}_i\left[\boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}_i^*\tilde{y}_i + \left(\frac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}_i^*\tilde{y}_i\right)^*\right]^{-2}$ *and H is the Hessian matrix defined previously.*

Proof. Let $\boldsymbol{a} = (\boldsymbol{\psi}_{\boldsymbol{a}}{}^\mathsf{T}, t_{\boldsymbol{a}})^\mathsf{T})$ and now we write

$$\begin{aligned}
&m(\boldsymbol{z}, \boldsymbol{\theta}_0 + \boldsymbol{a}) - m(\boldsymbol{z}, \boldsymbol{\theta}_0)\\
&= (\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{\Sigma}^*(\boldsymbol{\theta}_0 + \boldsymbol{a}) - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{\Sigma}^*\boldsymbol{\theta}\\
&+ \left[(\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - (\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-1} - \left[\boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-1}\\
&+\lambda\left(\tfrac{1}{\sqrt{\lambda}} - (\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+ - \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\\
&= \boldsymbol{a}^\mathsf{T}\boldsymbol{\Sigma}^*\boldsymbol{a} + 2\boldsymbol{a}^\mathsf{T}\boldsymbol{\Sigma}^*\boldsymbol{\theta}\\
&+ \left[(\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - (\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-1} - \left[\boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-1}\\
&+\lambda\left(\tfrac{1}{\sqrt{\lambda}} - (\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+ - \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\\
&= \boldsymbol{a}^\mathsf{T}D_{\boldsymbol{\theta}_0}m(\boldsymbol{z}, \boldsymbol{\theta}_0) + R(\boldsymbol{z}, \boldsymbol{a}),
\end{aligned}$$

where

$$\begin{aligned}
R(\boldsymbol{z}, \boldsymbol{a}) = \boldsymbol{a}^\mathsf{T}\boldsymbol{\Sigma}^*\boldsymbol{a} &+ \left[(\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - (\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-1}\\
&- \left[\boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - (\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\\
&- \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+ - \boldsymbol{a}^\mathsf{T}\boldsymbol{x}^*\tilde{y}\left[\boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-2},
\end{aligned}$$

$$\begin{aligned}
D_{\boldsymbol{a}}R(\boldsymbol{z}, \boldsymbol{a}) = 2\boldsymbol{\Sigma}^*\boldsymbol{a} &- \boldsymbol{x}^*\tilde{y}\left[(\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - (\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-2}\\
&+ \boldsymbol{x}^*\tilde{y}\left[\boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y} + \left(\tfrac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right)^+\right]^{-2},
\end{aligned}$$

$$D_{\boldsymbol{a}}[D_{\boldsymbol{a}}R(\boldsymbol{z}, \boldsymbol{a})] = 2\boldsymbol{\Sigma}^* + 2\boldsymbol{x}\boldsymbol{x}^\mathsf{T}I\left((\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y} \geq \tfrac{1}{\sqrt{\lambda}}\right)\left[(\boldsymbol{\theta}_0 + \boldsymbol{a})^\mathsf{T}\boldsymbol{x}^*\tilde{y}\right]^{-3}.$$

This gives, $R(\boldsymbol{z}, \boldsymbol{0}) = 0$, $D_{\boldsymbol{a}}R(\boldsymbol{z}, \boldsymbol{0}) = \boldsymbol{0}$ and $\mathrm{E}\left[D_{\boldsymbol{a}}[D_{\boldsymbol{a}}R(\boldsymbol{z}, \boldsymbol{0})]\right] = \boldsymbol{H}$. By definition we also have $\mathrm{E}\left[D_{\boldsymbol{\theta}_0}m(\boldsymbol{z}, \boldsymbol{\theta}_0)\right] = \boldsymbol{0}$. Hence

$$\mathrm{E}[m(\boldsymbol{z}, \boldsymbol{\theta}_0 + \boldsymbol{a}) - m(\boldsymbol{z}, \boldsymbol{\theta}_0)] = \mathrm{E}[R(\boldsymbol{z}, \boldsymbol{a})] = \frac{\boldsymbol{a}^\mathsf{T}\boldsymbol{H}\boldsymbol{a}}{2} + o(\|\boldsymbol{a}\|^2) \tag{2.31}$$

and since $\boldsymbol{H}$ is the Hessian of a strictly convex function we can establish that it is symmetric and positive definite. Now let $\boldsymbol{s} = (\boldsymbol{\psi_s}^\mathsf{T}, t_{\boldsymbol{s}})^\mathsf{T}$ and

$$A_n(\boldsymbol{s}) = \sum_{i=1}^{n} \left\{ m(\boldsymbol{z}_i, \boldsymbol{\theta}_0 + n^{-1/2}\boldsymbol{s}) - m(\boldsymbol{z}_i, \boldsymbol{\theta}_0) \right\}.$$

We can see that $A_n(\boldsymbol{s})$ is convex with respect to $\boldsymbol{s}$ and is therefore minimised by $\sqrt{n}(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0)$. Now we can write

$$\begin{aligned} A_n(\boldsymbol{s}) &= \sum_{i=1}^{n} \left\{ n^{-1/2}\boldsymbol{s}^\mathsf{T}\mathrm{B}(\boldsymbol{z}_i) + \mathrm{R}(\boldsymbol{z}_i, n^{-1/2}\boldsymbol{s}) - \mathrm{E}[\mathrm{R}(\boldsymbol{z}_i, n^{-1/2}\boldsymbol{s})] \right\} + n\mathrm{E}[\mathrm{R}(\boldsymbol{z}, n^{-1/2}\boldsymbol{s})] \\ &= n^{-1/2} \sum_{i=1}^{n} \boldsymbol{s}^\mathsf{T}\mathrm{B}(\boldsymbol{z}_i) + \frac{1}{2}\boldsymbol{s}^\mathsf{T}\boldsymbol{H}\boldsymbol{s} + r_{n,0}(\boldsymbol{s}) + r_{n,1}(\boldsymbol{s}), \end{aligned}$$

where $r_{n,0}(\boldsymbol{s}) = o(\|\boldsymbol{s}\|^2) \to 0$ for fixed $\boldsymbol{s}$ and $r_{n,1}(\boldsymbol{s}) = \sum_{i=1}^{n} \mathrm{R}(\boldsymbol{z}_i, n^{-1/2}\boldsymbol{s}) - \mathrm{E}[\mathrm{R}(\boldsymbol{z}_i, n^{-1/2}\boldsymbol{s})] \to 0$ in probability since it has mean zero and variance $o(\|\boldsymbol{s}\|^2)$. Since H is positive definite, and the covariance matrix $\mathrm{var}[\boldsymbol{X}]$ is finite, it follows from the basic corollary of Hjort and Pollard (1993) that (2.30) holds. □

Let $\boldsymbol{\theta}_{0r} = (\boldsymbol{\psi}_{0r}^\mathsf{T}, t_{0r})^\mathsf{T}$ be the minimiser of $\mathrm{E}[m(\boldsymbol{\theta}, \boldsymbol{Z}^r]$ and $\hat{\boldsymbol{\theta}}_{\boldsymbol{r}} = (\hat{\boldsymbol{\psi}}_r^\mathsf{T}, t_r)^\mathsf{T}$ be the minimiser of $\mathrm{E}_n[m(\boldsymbol{\theta}, \boldsymbol{Z}^r)]$. Let $\boldsymbol{H}_r$ be the Hessian matrix of $\mathrm{E}[m(\boldsymbol{\theta}, \boldsymbol{Z}^r)]$ and let $\boldsymbol{F}_r$ be the first p rows of $\boldsymbol{H}_r^{-1}$. By the last theorem we have

$$\hat{\boldsymbol{\psi}}_r = \boldsymbol{\psi}_{0r} - n^{-1}\boldsymbol{F}_r \sum_{i=1}^{n} \tilde{\mathrm{B}}_i(\boldsymbol{z}) + o_p(n^{-1/2}), \tag{2.32}$$

where $\tilde{\mathrm{B}}_i(\boldsymbol{z}) = 2\boldsymbol{\Sigma}\boldsymbol{\psi}_0 - \boldsymbol{x}_i\tilde{y}_i \left[\boldsymbol{\theta}_{0r}^\mathsf{T}\boldsymbol{x^*}_i\tilde{y}_i + \left(\frac{1}{\sqrt{\lambda}} - \boldsymbol{\theta}_{0r}^\mathsf{T}\boldsymbol{x^*}_i\tilde{y}_i \right)^+ \right]^{-2}$. Now let

$$\hat{\boldsymbol{M}}_n = \sum_{r=1}^{h-1} \boldsymbol{\psi}_r\boldsymbol{\psi}_r^\mathsf{T} \qquad \boldsymbol{M}_0 = \sum_{r=1}^{h-1} \boldsymbol{\psi}_{0r}\boldsymbol{\psi}_{0r}^\mathsf{T}. \tag{2.33}$$

Then it can be shown that

$$\hat{\boldsymbol{M}}_n = \boldsymbol{M}_0 + \sum_{r=1}^{h-1} \left\{ \boldsymbol{\psi}_{0r}^\mathsf{T}\mathrm{D}(\boldsymbol{\theta}_{0r}, \boldsymbol{z}) + \mathrm{D}^\mathsf{T}(\boldsymbol{\theta}_{0r}, \boldsymbol{z})\boldsymbol{\psi}_{0r} + \mathrm{D}(\boldsymbol{\theta}_{0r}, \boldsymbol{z})\mathrm{D}^\mathsf{T}(\boldsymbol{\theta}_{0r}, \boldsymbol{z}) \right\}, \tag{2.34}$$

where $\mathrm{D}(\boldsymbol{\theta}_{0r}, \boldsymbol{z}) = -n^{-1}\boldsymbol{F}_r \sum_{i=1}^{n} \tilde{\mathrm{B}}_i(\boldsymbol{z}) + o_p(n^{-1/2})$.

Having the influence function we can now demonstrate the consistency when p is fixed.

Theorem 2.26 *Let $\boldsymbol{\theta}_0 = (\boldsymbol{\psi}_0^\mathsf{T}, t_0)^\mathsf{T}$ be the minimiser of $\mathring{E}[(\boldsymbol{\theta}, \boldsymbol{Z})]$. Suppose for each $\tilde{y} = -1, 1$, the distribution of $\boldsymbol{X}|\tilde{Y} = \tilde{y}$ is dominated by the Lebesgue measure and $\mathring{E}[\|\boldsymbol{X}\|^2] < \infty$. Then $\hat{\boldsymbol{\theta}}$ is a consistent estimate of $\boldsymbol{\theta}_0$ as long as p and n tend to infinity.*

Proof. To begin, we first state the following identity:

$$\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0\|_2 \leq \sqrt{p} \max_i |[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0]_i|. \tag{2.35}$$

Using this and (2.30) we can write

$$\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0\|_2 \leq \sqrt{p} \max_i \left| n^{-1} \boldsymbol{H}_i^{-1} \sum_{j=1}^{n} B_j(\boldsymbol{z}) \right| + o_p(n^{-1/2}). \tag{2.36}$$

We know the first term on the right tends to 0 as $n \to \infty$, by the consistency of sample mean. Therefore, $\hat{\boldsymbol{\theta}}$ is a consistent estimator of $\boldsymbol{\theta}_0$ if $o_p(n^{-1/2}) \to 0$ as $n \to \infty$. □

2.3.4 Numerical studies

In this section we demonstrate the advantages of PDWD over PSVM through a simulation study and through a real data experiment.

2.3.4.1 Simulation studies

We use the following three synthetic models:

$$\text{Model I:} \quad Y = X_1 + X_2 + 0.2\epsilon$$

$$\text{Model II:} \quad Y = \frac{X_1}{0.5 + (X_2 + 1)^2} + 0.2\epsilon$$

$$\text{Model III:} \quad Y = X_1(X_1 + X_2 + 1) + 0.2\epsilon$$

where $X \sim \mathrm{N}(\mathbf{0}, I_p)$ and $\epsilon \sim \mathrm{N}(0, 1)$. We choose $n = 100$, $p = 20, 30, 50, 100$ and $h = 20$ unless stated otherwise.

We will use the distance method defined in Li et al. (2005) to estimate the performance of the algorithms. Let $\boldsymbol{\beta} \in \mathbb{R}^{p \times d}$ denote the basis of the central space and let $\hat{\boldsymbol{\beta}}$ be its estimator. Then we estimate the performance of $\hat{\boldsymbol{\beta}}$ as with the following distance measure

$$\mathrm{dist}(\boldsymbol{\beta}, \hat{\boldsymbol{\beta}}) = \|\boldsymbol{P}_{\boldsymbol{\beta}} - \boldsymbol{P}_{\hat{\boldsymbol{\beta}}}\|, \tag{2.37}$$

where $\boldsymbol{P}_{\boldsymbol{A}} = \boldsymbol{A}(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathsf{T}}$, that is the projection matrix, and $\|\cdot\|$ is the Frobenius norm.

We compare our method with PSVM and the results are shown in Table 2.1. The results show that PDWD and PSVM have similar performance for values of p close to n or close to 0 but for values in between PDWD has a clear advantage. In the classification literature (see Marron et al. (2007)) it was shown that DWD

clearly outperforms SVM for larger p due to the SVM suffering from data piling. The fact that here the two methods are equivalent as p tends to n we believe is due to the different nature of the problem. We emphasise that we are not interested in classification where the performance of the classifier is measured on the percentage of correctly classified points, and which will be hindered by data piling. Instead, we are interested in dimension reduction through hyperplane alignment. It seems that on the various iterations of the algorithm data piling actually "hinders" the performance of both PSVM and PDWD by causing them to overfit the data and that's why the performance of the two algorithms is becoming equivalent as p gets closer to n

Model	p	PSVM	PDWD
I	20	0.20 (0.044)	0.17 (0.038)
	30	0.27 (0.051)	0.24 (0.052)
	50	0.45 (0.072)	0.39 (0.069)
	100	1.30 (0.101)	1.28 (0.114)
II	20	1.01 (0.182)	1.01 (0.156)
	30	1.33 (0.128)	1.15 (0.135)
	50	1.51 (0.115)	1.42 (0.106)
	100	1.95 (0.038)	1.95 (0.039)
III	20	1.46 (0.235)	1.31 (0.202)
	30	1.70 (0.120)	1.51 (0.164)
	50	1.88 (0.061)	1.74 (0.121)
	100	1.97 (0.021)	1.97 (0.021)

Table 2.1: Comparison of estimation performance between PDWD and PSVM. The table reports the mean performance of 100 iterations (standard errors in parenthesis) for the two methods.

2.3.4.2 Computational time

As was mentioned earlier using a newly developed algorithm for DWD by Wang and Zou (2015) there is a computational advantage as the computation of Principal DWD is much less than the one for Principal SVM. We emphasise here that when Li et al. (2011) proposed Principal SVM they identified that the fact that PSVM needs quadratic programming leads to higher computational cost and that was probably the only disadvantage of PSVM over earlier methods which were based on inverse moments. As Figure 2.3 indicates there is a huge difference in time as n increases (and p is constant) while the difference stays relatively the same as p increases (and n is constant).

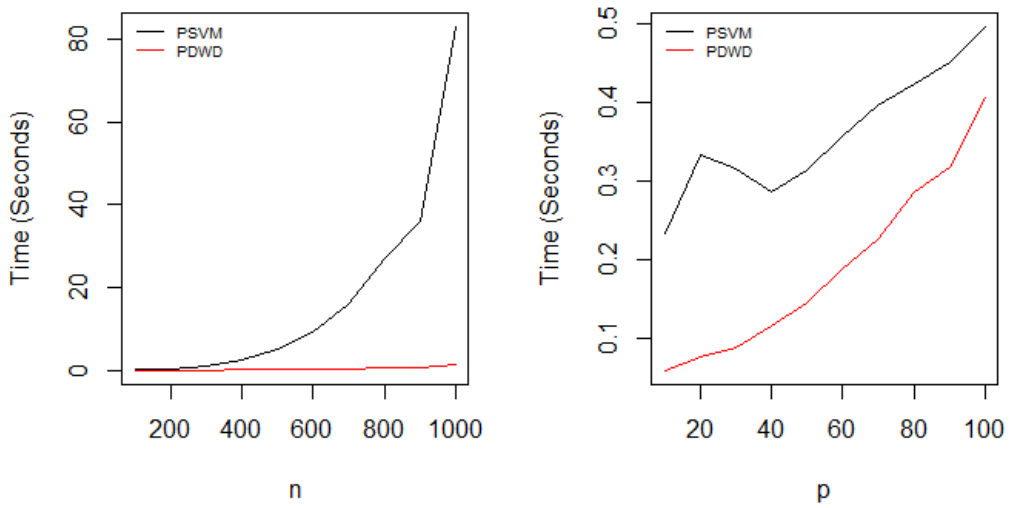

Figure 2.3: Left panel: time of two algorithms as n increases; right panel: time of two algorithms as p increases.

2.3.5 Real data analysis

$\tilde{p}$	20	40	60	80	100
PSVM	0.70 (0.244)	0.64 (0.227)	0.64 (0.206)	0.61 (0.170)	0.56 (0.080)
PDWD	0.03 (0.024)	0.09 (0.196)	0.15 (0.266)	0.13 (0.185)	0.13 (0.236)

Table 2.2: Distances as extra predictors are added in the dataset, for 100 simulations. Each column adds a different value of data, and we report the mean distance, standard deviation in parenthesis, of the estimated CS from the "oracle" CS, that is, the one when only the original predictors are used.

We now turn our attention to real data analysis. Our aim is to assess the effect of introducing random variables to the data. This will help us understand how robust our estimator is against unrelated data. Consider the Concrete slump data analysed in Yeh (1998). We have evaluated the response variable Compressive Strength. The data consists of 103 samples and 7 predictor variables called cement, slag, fly ash, water, superplasticizer (SP), coarse aggregate, and fine aggregate. Let $\tilde{p} = p+$ number of added predictors and we fix $\lambda = 0.1$ and $h = 20$. We first run the two methods and we calculate

$$\hat{\boldsymbol{\beta}}^{\mathsf{T}}_{\text{PDWD}} = (0.01, -0.001, 0.009, -0.024, 0.048, -0.005, -0.003)$$
$$\hat{\boldsymbol{\beta}}^{\mathsf{T}}_{\text{PSVM}} = (0.013, 0.002, 0.01, -0.02, 0.033, -0.003, -0.001)$$

which span the CS estimated by each method. Then we add extra predictors in the dataset, which are randomly distributed from a standard Normal distribution,

and calculate the new $\boldsymbol{\beta}$'s that span the Central Space using the two methods. We calculate the distance of the new vector from the original one, that is the one that was calculated based on the original predictors. Table 2.2 shows the distances between the estimator and the original estimator for each of the two methods, PDWD and PSVM, and for different number of added predictors $\tilde{p} = 20, 40, 60, 80, 100$. We can see that the estimator of the PDWD moves a lot less than the PSVM predictor. This implies that as unrelated features are added our estimates shows little change. This is what we would hope to see as the random features that have been added will give no information about the response variable and should therefore not affect the estimator. Therefore, we deduce that the PDWD estimator is more robust against random predictor variables.

Chapter 3

Order determination

Our theory so far has treated the dimension of the central subspace d as known, however in practice this is extremely unlikely. Developing an effective method for determining the dimension is vital when developing methods for SDR and plays an important role in the performance of such methods. This is ordinarily achieved through further analysis of the eigenvalues and eigenvectors of the candidate matrix produced.

3.1 Literature review

Many methods for order determination have been produced, Li (1991) proposed a sequential based method of order determination which relies on using the eigenvalues of the candidate matrix. Zhu et al. (2006) proposed the BIC type criteria for order determination, which once again takes advantage of the eigenvalues of the candidate matrix to find the optimum number of dimensions. There are many variations of the BIC criteria, see Wang and Yin (2008), Li et al. (2010) and Guo et al. (2015).

3.1.1 Sequential test

The sequential test, first discussed in Li (1991), was one of the first order determination estimators developed for dimension reduction and was further unified by Bura and Yang (2011). We define a candidate matrix to be a matrix whose columns span the CDRS. Let $\boldsymbol{M}$ to be a candidate matrix produced through a sufficient dimension reduction technique, then for any number $r = 0, \ldots, p-1$, the sequential test statistic takes the form

$$B_r = n \sum_{i=r}^{p} \lambda_i, \tag{3.1}$$

where the λ_i's are the eigenvalues of $\boldsymbol{M}$. When r is equal to the rank of $\boldsymbol{M}$ it is clear that B_r will be negligible, therefore the asymptotic distribution under the hypothesis

that r is the rank of $\boldsymbol{M}$ would be much larger than B_r. We then perform a sequence of hypothesis tests

$$H_0^r : \text{rank}(\boldsymbol{M}) = r, \quad r = 0, \ldots, p-1, \tag{3.2}$$

with the alternative hypothesis being $\text{rank}(\boldsymbol{M}) > r$. We can estimate the rank of $\boldsymbol{M}$ to be the first r for which the hypothesis is not rejected. If the hypothesis is not accepted for all $r = 0, \ldots, p-1$, then the rank is assumed to be p. Therefore, we estimate the dimension to be

$$\hat{d} = \min\{r : H_0^r \text{ is accepted}\} \quad \text{for } r = 0, \ldots, p-1, \tag{3.3}$$

where $\min_r\{\emptyset\} = p$. The generality of this method means that is can be adapted to be a compatible order determination estimator for most dimension reduction techniques.

3.1.2 BIC criteria

A form of BIC criteria was proposed in Li et al. (2011) which was an extension of a criterion introduced in Wang and Yin (2008). Define r to be the rank of the candidate matrix found using an SDR method. Using the form developed in Li et al. (2011) and the BIC criteria introduced in Zhou and Zhu (2016), Li (2018) developed a general form of BIC criteria given by

$$B_n(k) = \rho_k(\lambda_1, \ldots, \lambda_p) + c_1(n)c_2(k), \quad k = 0, \ldots, p, \tag{3.4}$$

where $c_1(n)$ is a sequence of positive numbers, $c_2(k)$ is an increasing function of k with $c_2(0) = 0$ and $\rho(\lambda_1, \ldots, \lambda_p)$ are differentiable functions that satisfy

$$\rho_0(\lambda_1, \ldots, \lambda_p) < \cdots < \rho_r(\lambda_1, \ldots, \lambda_p) = \rho_{r+1}(\lambda_1, \ldots, \lambda_p) = \cdots = \rho_p(\lambda_1, \ldots, \lambda_p). \tag{3.5}$$

This then gives the estimate of d to be

$$\hat{d} = \max\{k : B_n(k)\} \quad \text{for } k = 0, \ldots, p.$$

When $c_1(n)$ is a decreasing function $B_n(k)$ will increase with a peak at $k = r$ and $B_n(k)$ will be decreasing for $p > r$.

3.1.3 Ladle plot

The Ladle estimator, developed by Luo and Li (2016), is a combinations of the scree plot method and the Ye-Weiss plot developed by Ye and Weiss (2003). Since $\hat{\boldsymbol{M}}$ is a consistent estimator of $\boldsymbol{M}$ and $\boldsymbol{M}$ has rank d we can know that $\hat{\lambda}_{d+1}$ will be much smaller than $\hat{\lambda}_d$, where $\boldsymbol{M}$ is a candidate matrix and the λ_i's are the eigenvalues of $\boldsymbol{M}$. Using this the following function is defined

$$\phi_n : \{0, \ldots, p-1\} \to \mathbb{R}, \quad \phi_n(k) = \frac{\hat{\lambda}_{k+1}}{1 + \sum_{i=0}^{p-1} \hat{\lambda}_{i+1}}. \tag{3.6}$$

The eigenvalues have been shifted so that ϕ_n takes small values at $k = d$ rather than at $k = d + 1$.

Next, we turn our attention to the Ye-Weiss plot. Let F be the distribution of $(\boldsymbol{X}, Y)$ and let F_n be the empirical distribution based on $S = (\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)$. Conditioning on S, let $(\boldsymbol{X}^*_{1,n}, Y^*_{1,n}), \ldots, (\boldsymbol{X}^*_{n,n}, Y^*_{n,n})$ be an i.i.d bootstrap sample from F_n. Now define $\{\hat{\lambda}_1, \ldots, \hat{\lambda}_p, \hat{v}_1, \ldots, \hat{v}_p\}$ and $\{\lambda^*_1, \ldots, \lambda^*_p, v^*_1, \ldots, v^*_p\}$ be the eigenvalues and eigenvectors of $\hat{\boldsymbol{M}}$ and $\boldsymbol{M}^*$ respectively. For each $k < p$, let

$$\hat{\boldsymbol{B}}_k = (\hat{v}_1, \ldots, \hat{v}_k) \quad \boldsymbol{B}^*_k = (v^*_1, \ldots, v^*_k)$$

and define the function

$$f^0_n : \{0, \ldots, p-1\} \to \mathbb{R}, \quad f^0_n(k) = \begin{cases} 0 & k = 0 \\ n^{-1} \sum_{i=1}^n 1 - |\det(\boldsymbol{B}_k^{\mathsf{T}} \boldsymbol{B}^*_{k,i})| & k = 1, \ldots, p-1 \end{cases} \tag{3.7}$$

where $\boldsymbol{B}^*_{k,i}$ denotes the ith bootstrap sample. From Ye and Weiss (2003), it can be established that the function $f^0_n(k)$ gives a measure of the variability of the bootstrap estimates around the full sample estimate $\hat{\boldsymbol{B}}_k$. The range of f^0_n is $[0, 1]$, where 0 indicates each $\boldsymbol{B}^*_{k,i}$ spans the same column space as $\hat{\boldsymbol{B}}_k$ and 1 occurs when $\boldsymbol{B}^*_{k,i}$ spans a space orthogonal to $\hat{\boldsymbol{B}}_k$. So if we define the function

$$f_n : \{0, \ldots, p-1\} \to \mathbb{R} \quad f_n(k) = \frac{f^0_n(k)}{1 + \sum_{i=0}^{p-1} f^0_n(i)}. \tag{3.8}$$

Ye and Weiss (2003) determined that f_n is small for $k = d$ and larger for $k > d$.

Lastly, the ladle estimator of the rank d is defined to be

$$\hat{d} = \arg\min_k \{g_n(k) : k \in \mathcal{D}(g_n)\}, \tag{3.9}$$

where $g_n(k) = \phi_n(k) + f_n(k)$.

3.2 Numerical studies

Consider the synthetic regression models

$$\text{Model I:} \quad Y = X_1 + X_2 + 0.2\epsilon$$

$$\text{Model II:} \quad Y = \frac{X_1}{0.5 + (X_2 + 1)^2} + 0.2\epsilon$$

$$\text{Model III:} \quad Y = X_1(X_1 + X_2 + 1) + 0.2\epsilon$$

Choosing $n = 100$ and $p = 10$, Figure 3.1 shows the ladle plot for model II. As we can see, the ladle plot correctly estimates d to be 2. We consider the models

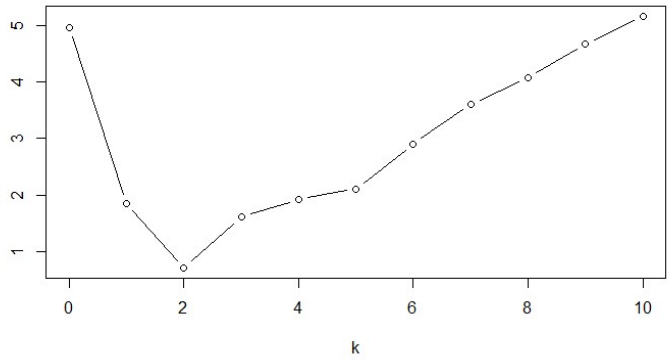

Figure 3.1: Ladle plot of model II with $n = 100$ and $p = 10$.

defined above where model I has effective dimension 1 and models II and III have effective dimension 2. We run 1000 simulation experiments with $n = 100$, $\sigma = 0.2$ and $h = 20$. Table 3.1 shows the percentage of correct estimates as p increases. This is a very promising result as it demonstrates that the performance of the algorithm does not suffer a lot when the dimension is increased, instead we can see that as p increase the number of correct estimates for Models II and III decreases slightly but remains high.

Model	p		
	10	30	50
I	100	100	100
II	99	98	97
III	99	98	97

Table 3.1: Percentage of correct estimations of d in 1000 simulations using the ladle estimator for the three models.

Chapter 4

Non-linear SDR

In recent years there is an interest in non-linear SDR, where we extract linear or non-linear functions of the predictors. Extracting non-linear functions of the predictors will enable us to reduce the dimension of data further. Consider the model $Y = X_1X_2 + X_3X_4 + \epsilon$. We can see that this model has four linear directions which are X_1, X_2, X_3 and X_4. Alternatively, this model has one non-linear direction which is $X_1X_2 + X_3X_4$.

To perform linear dimension reduction, we work under the non-linear conditional independence model:

$$Y \perp\!\!\!\perp \boldsymbol{X} | \boldsymbol{f}(\boldsymbol{X}), \tag{4.1}$$

where $\boldsymbol{f} : \mathbb{R}^p \to \mathbb{R}^d$ denotes linear or non-linear functions of the predictors. Some examples include the work by Wu (2008) and Yeh et al. (2009) which introduced Kernel SIR and the work by Fukumizu et al. (2009) which used kernel regression.

4.0.1 Reproducing kernel Hilbert space

The Reproducing Kernel Hilbert Space (RKHS) is defined repeatedly (see Aronszajn (1950), Berlinet and Thomas-Agnan (2004), Wu (2008) and Li (2018)) in non-linear literature and will play a vital role in the extension of our method to a unified linear and non-linear setting. To begin we will give a clear definition of a RKHS.

To understand a reproducing kernel Hilbert space we will first define a reproducing kernel (Berlinet and Thomas-Agnan (2004)).

Definition 4.1 *Let $\mathcal{H}$ be a Hilbert space of $\mathbb{R}$-valued functions defined on a non-empty set $\mathcal{X}$. A function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a reproducing kernel of H if it satisfies:*

$$\forall x \in \mathcal{X}, \kappa(\cdot, x) \in \mathcal{H}$$
$$\forall x \in \mathcal{X}, \forall g \in \mathcal{H}, \langle g, \kappa(\cdot, x)\rangle_{\mathcal{H}} = g(x).$$

The second condition is called the reproducing property.

A Hilbert space which possesses a reproducing kernel is called a reproducing kernel Hilbert space.

4.1 Previous methods

4.1.1 Kernel sliced inverse regression

The work developed by Wu (2008) adapts linear SIR into the non-linear setting using what is known as the kernel trick and named kernel SIR. It was proposed that in the non-linear setting, that with the introduction of kernel data, SIR can be reformulated as solving the eigen-problem

$$\boldsymbol{E}_h \boldsymbol{K} \boldsymbol{a} = \lambda \boldsymbol{K} \boldsymbol{a}, \tag{4.2}$$

where $\boldsymbol{K} := \{\kappa_{ij} = \langle \boldsymbol{\phi}(\boldsymbol{X}_i), \boldsymbol{\phi}(\boldsymbol{X}_j) \rangle_{\mathcal{H}}\}$, $\boldsymbol{E}_h = \sum_{j=1}^{h} n_j^{-1} \mathbf{1}_j \mathbf{1}_j^{\mathsf{T}}$, $\mathbf{1}_j = [\delta_j(y_1) \dots \delta_j(y_n)]^{\mathsf{T}}$ and $\boldsymbol{a}$ is an n-vector whose ith element is the coefficient a_i.

Therefore, Kernel SIR performs a spectrum decomposition of the weighted kernel matrix $\boldsymbol{E}_h \boldsymbol{K}$ with respect to the kernel matrix $\boldsymbol{K}$. Let $\lambda_1, \dots, \lambda_n$ denote the eigenvalues, and $\boldsymbol{a}^1, \dots, \boldsymbol{a}^n$ be the corresponding complete set of eigenvectors. Through further investigation, Wu (2008) proposed that the projections of $\boldsymbol{\phi}(\boldsymbol{X})$ along the eigenvectors $\boldsymbol{a}^k$, $k = 1, \dots, n$ are given by

$$\langle \boldsymbol{\beta}_k, \boldsymbol{\phi}(\boldsymbol{X}) \rangle_{\mathcal{H}} = \sum_{i=1}^{n} a_i^k \langle \boldsymbol{\phi}(\boldsymbol{X}_i), \boldsymbol{\phi}(\boldsymbol{X}) \rangle_{\mathcal{H}} = \sum_{i=1}^{n} a_i^k \kappa(\boldsymbol{X}_i, \boldsymbol{X}). \tag{4.3}$$

Similar to the linear case, the following assumption is present in many dimension reduction methods.

Assumption 4.2 *For any $\boldsymbol{v} \in \mathcal{H}$, we have that $\mathrm{E}[\boldsymbol{v}^{\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{X}) | \boldsymbol{\beta}^{\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{X})]$ is linear in $\boldsymbol{\beta}^{\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{X})$.*

This is once again equivalent to assuming $\boldsymbol{\phi}(\boldsymbol{X})$ is elliptically symmetric and is named the linear design condition.

The following theorem forms the basis for the kernel SIR methodology.

Theorem 4.3 *Under assumption 4.2, $\mathrm{E}[\boldsymbol{\phi}(\boldsymbol{X})|y] - \mathrm{E}[\boldsymbol{\phi}(\boldsymbol{X})]$ falls into the linear subspace spanned by $\boldsymbol{\beta}^{\mathsf{T}} \boldsymbol{\Sigma}_{xx}$, where $\boldsymbol{\Sigma}_{xx}$ is the covariance matrix of $\boldsymbol{X}$.*

The estimation algorithm for kernel SIR is almost the same as the linear estimation algorithm, with the addition of a first step. The estimation procedure, given in Wu (2008), is as follows:

1. Prepare the data in kernel form $\tilde{\boldsymbol{K}}$, where $\tilde{\boldsymbol{K}}$ is centered and possibly reduced.

2. Partition a range of Y into h slices to get the discretised $\tilde{Y}$.

3. Calculate within-slice means for each slice and the between-slice covariance matrix $\hat{V}$ using $\tilde{\boldsymbol{K}}$ in place of $\boldsymbol{X}$. Also calculate the covariance matrix for the kernel data, denoted by $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{K}}}$.

4. Extract the leading eigenvalues and eigenvectors of $\hat{V}$ with respect to $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{K}}}$. This is equivalent to solving the eigen-problem in equation (4.2).

5. Normalise the eigenvectors and get the projection directions.

4.1.2 Non-linear principal support vector machines

Li et al. (2011) extended linear PSVM to the non-linear problem defined in (4.1) via the reproducing kernel Hilbert space (RKHS). Under a unified framework the methodology of the non-linear method follows a very similar layout to the linear methodology. Let $\mathcal{H}$ be defined as a RKHS, then the sample version of the PSVM objective function can be written as

$$\hat{\Lambda}(\boldsymbol{c}) = n^{-1}\boldsymbol{c}^{\mathsf{T}}\boldsymbol{\Psi}^{\mathsf{T}}\boldsymbol{\Psi}\boldsymbol{c} + \lambda n^{-1}\sum_{i=1}^{n}\Big(1 - \tilde{Y}_i(\boldsymbol{\Psi}_i^{\mathsf{T}}\boldsymbol{c} - t)\Big)^{+}, \tag{4.4}$$

where $\boldsymbol{\Psi}_i^{\mathsf{T}} = (\psi_1(\boldsymbol{X}_i), \ldots, \psi_k(\boldsymbol{X}_i))$, $\boldsymbol{c} \in \mathbb{R}^k$ and $\Lambda : \mathcal{H} \times \mathbb{R} \to \mathbb{R}^+$. The quadratic programming problem that solves (4.4) is defined in the following theorem, where the symbol $\odot$ is the Hadamard product and $\boldsymbol{P}_{\boldsymbol{A}}$ is the projection matrix $\boldsymbol{A}(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathsf{T}}$ for a matrix $\boldsymbol{A}$ of full rank.

The next theorem outlines the non-linear PSVM problem as a quadratic programming problem, taken from Li et al. (2011).

Theorem 4.4 *If $\boldsymbol{c}^*$ minimises $\hat{\Lambda}(\boldsymbol{c})$ over $\mathbb{R}^k$, then $\boldsymbol{c}^* = \frac{1}{2}(\boldsymbol{\Psi}^{\mathsf{T}}\boldsymbol{\Psi})^{-1}\boldsymbol{\Psi}^{\mathsf{T}}(\tilde{\boldsymbol{y}} \odot \boldsymbol{\alpha}^*)$, where $\boldsymbol{\alpha}^*$ is the solution to the quadratic programming problem:*

$$\begin{aligned} &\textit{maximise } \mathbf{1}^{\mathsf{T}}\boldsymbol{\alpha} - \frac{1}{4}(\boldsymbol{\alpha} \odot \tilde{\boldsymbol{y}})^{\mathsf{T}}\boldsymbol{P}_{\boldsymbol{\Psi}}(\boldsymbol{\alpha} \odot \tilde{\boldsymbol{y}}) \\ &\textit{subject to } 0 \leq \alpha \leq \lambda,\ \boldsymbol{\alpha}^{\mathsf{T}}\tilde{\boldsymbol{y}} = 0. \end{aligned} \tag{4.5}$$

Note the projection matrix $\boldsymbol{P}_{\boldsymbol{\Psi}}$ is replaced by the kernel matrix $\boldsymbol{K}_n = \{\kappa(i,j) : i,j = 1,\ldots,n\}$ for some positive bivariate mapping $\kappa : \Omega_{\boldsymbol{X}} \times \Omega_{\boldsymbol{X}} \to \mathbb{R}$. Let $\boldsymbol{Q}_n = \boldsymbol{I}_n - \boldsymbol{J}_n/n$, where $\boldsymbol{I}_n$ is the $n \times n$ identity matrix and $\boldsymbol{J}_n$ is an $n \times n$ matrix of 1's. The following proposition is taken from Li et al. (2011).

Proposition 1 *Let $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_n)$, $\psi_{\boldsymbol{\omega}} = \sum_{i=1}^{n}\omega_i\left[\kappa(\boldsymbol{X}, \boldsymbol{X}_i) - \mathrm{E}_n\kappa(\boldsymbol{X}, \boldsymbol{X})\right]$. The following statements are equivalent:*

1. *$\boldsymbol{\omega}$ is an eigenvector of the matrix $\boldsymbol{Q}_n\boldsymbol{K}_n\boldsymbol{Q}_n$ with eigenvalues λ.*

2. *$\psi_{\boldsymbol{\omega}}$ is an eigenfunction of the operator Σ_n with eigenvalue λ/n.*

If $\lambda \neq 0$*, then either statement implies* $(\psi_{\boldsymbol{\omega}}(\boldsymbol{X}_1), \ldots, \psi_{\boldsymbol{\omega}}(\boldsymbol{X}_n)) = \lambda \boldsymbol{\omega}^\mathsf{T}$

The above proposition can be used to find the eigenfunctions of Σ_n. The estimation algorithm, discussed in Li et al. (2011), takes the form:

1. (Optional) Marginally standardise $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$. This step can be omitted if the components of $\boldsymbol{X}_i$ have similar variances.

2. Choose a kernel κ and the number of basis functions k (say $k = n/2$). Compute $\boldsymbol{\Psi} = (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_k)$ and $\boldsymbol{P}_{\boldsymbol{\Psi}}$ from $\boldsymbol{Q}_n \boldsymbol{K}_n \boldsymbol{Q}_n$.

3. Divide the sample according to LVR or OVA, each yielding a set of slices. For each pair of slices, solve the quadratic programming problem in Theorem 4.4. This gives coefficients $\boldsymbol{c^*}_1, \ldots, \boldsymbol{c^*}_{\tilde{h}} \in \mathbb{R}^k$, where $\tilde{h} = h - 1$ for LVR and $\binom{h}{r}$ for OVA.

4. Compute the first d eigenvectors $\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d$ of the matrix $\sum_{i=1}^{\tilde{h}} \boldsymbol{c}_i \boldsymbol{c}_i^\mathsf{T}$. Denote the rth component of $\boldsymbol{v}_s$ as v_{sr}.

5. The sth sufficient predictor evaluated at $\boldsymbol{x}$ is $v_{s1}\psi_1(x) + \cdots + v_{sk}\psi_k(\boldsymbol{x})$, where $\psi_r(\boldsymbol{x}) = \lambda_r^{-1} \sum_{i=1}^n \omega_{ri}[\kappa(\boldsymbol{x}, \boldsymbol{X}_i) - \mathrm{E}_n \kappa(\boldsymbol{x}, \boldsymbol{X})]$. If step 1 is used, then $\boldsymbol{x}$ should be marginally standardised.

4.2 Non-linear principal distance-weighted discrimination

In this section we turn our attention to the extension of this method to the non-linear case. Analogous to the work developed by Li et al. (2011) we will be expanding linear PDWD for non-linear PDWD under a unified framework. We have specified that the PSVM problem is a quadratic programming problem. Similar to the linear problem, due to the convexity and differentiability of the DWD problem, the non-linear DWD problem can be solved by finding the zeroes of the first derivative. This usually yields faster results that quadratic programming.

4.2.1 Non-linear sufficient dimension reduction using distance-weighted discrimination

Let $\mathcal{H}$ be a reproducing kernel Hilbert space of functions of $\boldsymbol{X}$ with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$. Similar to the linear case, the objective function, $\Lambda(\psi, t) : \mathcal{H} \times \mathbb{R} \mapsto \mathbb{R}^+$,

takes the form

$$\Lambda(\psi,t) = \mathrm{var}(\psi(\boldsymbol{X})) + \mathrm{E}\Bigg[\lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})] - t)\right)^{+} + \left[\tilde{Y}(\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})] - t) + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})] - t)\right)^{+}\right]^{-1}\Bigg], \tag{4.6}$$

where $\tilde{Y}$ is defined as in (2.2). Now define $\langle \phi_1, \Sigma\phi_2 \rangle_{\mathcal{H}} = \mathrm{cov}[\phi_1(\boldsymbol{X}), \phi_2(\boldsymbol{X})]$, for any $\phi_1, \phi_2 \in \mathcal{H}$, where $\Sigma : \mathcal{H} \mapsto \mathcal{H}$ is the covariance operator. Therefore (4.6) can be rewritten as

$$\Lambda(\psi,t) = \langle \psi, \Sigma\psi \rangle_{\mathcal{H}} + \mathrm{E}\Bigg[\lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})] - t)\right)^{+} + \left[\tilde{Y}(\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})] - t) + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})] - t)\right)^{+}\right]^{-1}\Bigg]. \tag{4.7}$$

Finally, for $\hat{u} = \tilde{Y}(\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})] - t)$ we can write

$$\Lambda(\psi,t) = \langle \psi, \Sigma\psi \rangle_{\mathcal{H}} + \mathrm{E}\left[\lambda\left(\tfrac{1}{\sqrt{\lambda}} - \hat{u}\right)^{+} + \left[\hat{u} + \left(\tfrac{1}{\sqrt{\lambda}} - \hat{u}\right)^{+}\right]^{-1}\right]. \tag{4.8}$$

Lemma 4.5 *Suppose the mapping $\mathcal{H} \to L_2(P_{\boldsymbol{X}})$, $f \mapsto f$ is continuous. Then for each fixed t in $\mathbb{R}$, the function $\psi \mapsto \Lambda(\psi, t)$ is continuous with respect to the $L_2(P_{\boldsymbol{X}})$-norm.*

Proof. Let ψ_1 and ψ_2 be two members of $L_2(P_{\boldsymbol{X}})$, where $\hat{u}_1 = \tilde{Y}(\psi_1(\boldsymbol{X}) - \mathrm{E}[\psi_1(\boldsymbol{X})] - t)$ and $\hat{u}_2 = \tilde{Y}(\psi_2(\boldsymbol{X}) - \mathrm{E}[\psi_2(\boldsymbol{X})] - t)$. Then

$$|\Lambda(\psi_2,t) - \Lambda(\psi_1,t)| \le |\mathrm{var}[\psi_2(\boldsymbol{X})] - \mathrm{var}[\psi_1(\boldsymbol{X})]| + \mathrm{E}\Bigg|\lambda\left(\tfrac{1}{\sqrt{\lambda}} - \hat{u}_2\right)^{+} - \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \hat{u}_1\right)^{+} + \left[\hat{u}_2 + \left(\tfrac{1}{\sqrt{\lambda}} - \hat{u}_2\right)^{+}\right]^{-1} - \left[\hat{u}_1 + \left(\tfrac{1}{\sqrt{\lambda}} - \hat{u}_1\right)^{+}\right]^{-1}\Bigg|.$$

We start by considering the first term on the right-hand side. This gives

$$\begin{aligned}
&|\mathrm{var}[\psi_2(\boldsymbol{X})] - \mathrm{var}[\psi_1(\boldsymbol{X})]| \\
&\quad = |\mathrm{var}[\psi_2(\boldsymbol{X}) - \psi_1(\boldsymbol{X}) + \psi_1(\boldsymbol{X})] - \mathrm{var}[\psi_1(\boldsymbol{X})]| \\
&\quad = |\mathrm{var}[\psi_2(\boldsymbol{X}) - \psi_1(\boldsymbol{X})] + 2\mathrm{cov}[\psi_2(\boldsymbol{X}) - \psi_1(\boldsymbol{X}), \psi_1(\boldsymbol{X})]| \\
&\quad \le |\mathrm{var}[\psi_2(\boldsymbol{X}) - \psi_1(\boldsymbol{X})]| + 2|\mathrm{var}[\psi_2(\boldsymbol{X}) - \psi_1(\boldsymbol{X})]\mathrm{var}[\psi_1(\boldsymbol{X})]|^{1/2} \\
&\quad \le \|\psi_2 - \psi_1\|^2_{L_2(P_{\boldsymbol{X}})} + 2\|\psi_2 - \psi_1\|_{L_2(P_{\boldsymbol{X}})}\|\psi_1\|_{L_2(P_{\boldsymbol{X}})}.
\end{aligned}$$

Before we consider the remaining terms of the above equation, we first note, for $a, b \in \mathbb{R}$ and $c > 0$ we have

$$|[b + (c-b)^{+}]^{-1} - [a + (c-a)^{+}]^{-1} + c^{-2}(c-b)^{+} - c^{-2}(c-a)^{+}| \le c^{-2}|a-b|.$$

Therefore, the remaining terms can be rewritten as

$$\begin{aligned}
&\mathrm{E}\left|\lambda\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}_2\right)^+-\lambda\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}_1\right)^+ +\left[\hat{u}_2+\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}_2\right)^+\right]^{-1}-\left[\hat{u}_1+\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}_1\right)^+\right]^{-1}\right| \\
&\qquad\leq \lambda\mathrm{E}|\hat{u}_2-\hat{u}_1| \\
&\qquad= \lambda\mathrm{E}|\tilde{Y}(\psi_2(\boldsymbol{X})-t)-\tilde{Y}(\psi_1(\boldsymbol{X})-t)| \\
&\qquad= \lambda\mathrm{E}|\psi_2(\boldsymbol{X})-\psi_1(\boldsymbol{X})| \\
&\qquad= \lambda\|\psi_2-\psi_1\|_{L_2(P_{\boldsymbol{X}})}.
\end{aligned}$$

Combining this, we find

$$\begin{aligned}
|\Lambda(\psi_2,t)-\Lambda(\psi_1,t)| &\leq \|\psi_2-\psi_1\|^2_{L_2(P_{\boldsymbol{X}})}+2\|\psi_2-\psi_1\|_{L_2(P_{\boldsymbol{X}})}\|\psi_1\|_{L_2(P_{\boldsymbol{X}})}+\lambda\|\psi_2-\psi_1\|_{L_2(P_{\boldsymbol{X}})} \\
&= \|\psi_2-\psi_1\|_{L_2(P_{\boldsymbol{X}})}\left(\|\psi_2-\psi_1\|_{L_2(P_{\boldsymbol{X}})}+2\|\psi_1\|_{L_2(P_{\boldsymbol{X}})}+\lambda\right).
\end{aligned}$$

Therefore $|\Lambda(\psi_2,t)-\Lambda(\psi_1,t)|\to 0$ as $\|\psi_2-\psi_1\|\to 0$. □

Following the definition 1 in Li et al. (2011), we say that a function $\psi\in\mathcal{H}$ is unbiased for non-linear sufficient dimension reduction if it has a version that is measurable with respect to $\sigma\{\boldsymbol{f}(\boldsymbol{X})\}$. Using this then we prove the following theorem which proves that the minimiser of the objective function (4.7) estimates the CS.

Theorem 4.6 *Suppose the mapping $\mathcal{H}\to L_2(P_{\boldsymbol{X}})$, $f\mapsto f$ is continuous and*

1. *$\mathcal{H}$ is a dense subset of $L_2(P_{\boldsymbol{X}})$*
2. *$Y\perp\!\!\!\perp\boldsymbol{X}|\boldsymbol{f}(\boldsymbol{X})$*

If (ψ^*,t^*) minimises $\Lambda(\psi,t)$ among all $(\psi,t)\in\mathcal{H}\times\mathbb{R}$, then $\psi^*(\boldsymbol{X})$ is unbiased.

Proof. Beginning with the first term we have

$$\mathrm{var}[\psi(\boldsymbol{X})]=\mathrm{var}[\mathrm{E}[\psi(\boldsymbol{X})|\boldsymbol{f}(\boldsymbol{X})]]+\mathrm{E}[\mathrm{var}[\psi(\boldsymbol{X})|\boldsymbol{f}(\boldsymbol{X})]]\geq\mathrm{var}[\mathrm{E}[\psi(\boldsymbol{X})|\boldsymbol{f}(\boldsymbol{X})]]. \tag{4.9}$$

Now let us look at the second term. Again, we can write

$$\begin{aligned}
&\mathrm{E}\left[\left[\hat{u}+\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}\right)^+\right]^{-1}+\lambda\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}\right)^+\right] \\
&=\mathrm{E}\left[\mathrm{E}\left[\left[\hat{u}+\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}\right)^+\right]^{-1}+\lambda\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}\right)^+\middle|\tilde{Y},\boldsymbol{f}(\boldsymbol{X})\right]\right].
\end{aligned}$$

If we define the function g such that $g(a)=\left[a+\left(\tfrac{1}{\sqrt{\lambda}}-a\right)^+\right]^{-1}+\lambda\left(\tfrac{1}{\sqrt{\lambda}}-a\right)^+$ then this gives

$$\mathrm{E}\left[\left[\hat{u}+\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}\right)^+\right]^{-1}+\lambda\left(\tfrac{1}{\sqrt{\lambda}}-\hat{u}\right)^+\middle|\tilde{Y},\boldsymbol{f}(\boldsymbol{X})\right]=\mathrm{E}[g(\hat{u})|\tilde{Y},\boldsymbol{f}(\boldsymbol{X})].$$

Since g is a convex function, we can use Jenson's inequality as follows:

$$\begin{aligned}
\mathrm{E}[f(\hat{u})|\tilde{Y}, \boldsymbol{f}(\boldsymbol{X})] &\geq \left[\mathrm{E}[\hat{u}|\tilde{Y}, \boldsymbol{f}(\boldsymbol{X})] + \left(\tfrac{1}{\sqrt{\lambda}} - \mathrm{E}[\hat{u}|\tilde{Y}, \boldsymbol{f}(\boldsymbol{X})]\right)^+\right]^{-1} + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \mathrm{E}[\hat{u}|\tilde{Y}, \boldsymbol{f}(\boldsymbol{X})]\right)^+ \\
&\geq \Bigg[\tilde{Y}(\mathrm{E}[\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})]|\boldsymbol{f}(\boldsymbol{X})] - t) \\
&\qquad + \left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\mathrm{E}[\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})]|\boldsymbol{f}(\boldsymbol{X})] - t)\right)^+\Bigg]^{-1} \\
&\qquad + \lambda\left(\tfrac{1}{\sqrt{\lambda}} - \tilde{Y}(\mathrm{E}[\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})]|\boldsymbol{f}(\boldsymbol{X})] - t)\right)^+
\end{aligned}$$

Thus combing this with (4.9) we get

$$L(\boldsymbol{\psi}, t) \geq \Lambda(\mathcal{L}(\psi), t), \tag{4.10}$$

where $\mathcal{L}(\psi)$ denotes the function $\mathrm{E}[\psi(\boldsymbol{X}) - \mathrm{E}[\psi(\boldsymbol{X})]|\boldsymbol{f}(\boldsymbol{X})]$. Equation (4.9) becomes strict if $\mathrm{E}[\mathrm{var}[\psi(\boldsymbol{X})|\boldsymbol{f}(\boldsymbol{X})]] > 0$. The equality $\mathrm{E}[\mathrm{var}[\psi(\boldsymbol{X})|\boldsymbol{f}(\boldsymbol{X})]] = 0$ means that $\psi(\boldsymbol{X})$ is constant given $\boldsymbol{f}(\boldsymbol{X})$ and is therefore equivalent to there being a version of ψ that is measurable with respect to $\sigma\{\boldsymbol{f}(\boldsymbol{X})\}$. Hence if there is no version of ψ that is measurable with respect to $\sigma\{\boldsymbol{f}(\boldsymbol{X})\}$, then

$$\Lambda(\psi, t) > \Lambda(\mathcal{L}(\psi), t).$$

Since $\mathcal{H} \subset L_2(P_{\boldsymbol{X}})$, ψ belongs to $L_2(P_{\boldsymbol{X}})$, for any $\epsilon > 0$, there is a $\psi_1 \in \mathcal{H}$ such that

$$\|\psi_1 - \mathcal{L}(\psi)\|_{L_2(P_{\boldsymbol{X}})} < \epsilon.$$

By Lemma 4.5, we can choose ϵ to be sufficiently small so that $\Lambda(\psi, t) > \Lambda(\psi_1, t)$, which means ψ cannot be ψ^*. □

4.2.2 Sample estimation algorithm

Let $\mathcal{H}$ be a linear space of functions from $\Omega_{\boldsymbol{X}}$ to $\mathbb{R}$ spanned by $\mathcal{F}_n = \{\psi_1, \ldots, \psi_n\}$. These functions are chosen, such that, $\mathrm{E}_n[f_i(\boldsymbol{X})] = 0$. Let

$$\boldsymbol{\Psi} = \begin{pmatrix} \psi_1(\boldsymbol{X}_1) & \cdots & \psi_1(\boldsymbol{X}_n) \\ \vdots & \ddots & \vdots \\ \psi_n(\boldsymbol{X}_1) & \cdots & \psi_n(\boldsymbol{X}_n) \end{pmatrix}.$$

Hence, the sample version of (4.7) becomes

$$\begin{aligned}
\hat{\Lambda}(\boldsymbol{c}) = \boldsymbol{c}^\mathsf{T}\boldsymbol{\Psi}^\mathsf{T}\boldsymbol{\Psi}\boldsymbol{c} + \frac{1}{n}\sum_{i=1}^{n}\Bigg[&\lambda\left(\frac{1}{\sqrt{\lambda}} - \tilde{Y}_i(\boldsymbol{\Psi}_i^\mathsf{T}\boldsymbol{c} - t)\right)^+ \\
&+ \left[\tilde{Y}_i(\boldsymbol{\Psi}_i^\mathsf{T}\boldsymbol{c} - t) + \left(\frac{1}{\sqrt{\lambda}} - \tilde{Y}_i(\boldsymbol{\Psi}_i^\mathsf{T}\boldsymbol{c} - t)\right)^+\right]^{-1}\Bigg].
\end{aligned} \tag{4.11}$$

where $\boldsymbol{\Psi}_i^\mathsf{T} = (\psi_1(\boldsymbol{X}_i), \ldots, \psi_k(\boldsymbol{X}_i))$, $\boldsymbol{c} \in \mathbb{R}^k$ and $\Lambda : \mathcal{H} \times \mathbb{R} \to \mathbb{R}^+$.

This problem differs from the kernel objective function, given in Wang and Zou (2015), where $\boldsymbol{\Psi}^\mathsf{T}\boldsymbol{\Psi}$ is replaced by the kernel matrix $\boldsymbol{K}_n = \{\kappa(i,j) : i,j = 1,\ldots,n\}$ for some positive definite bivariate mapping $\kappa : \Omega_{\boldsymbol{X}} \times \Omega_{\boldsymbol{X}} \to \mathbb{R}$. For the function class $\mathcal{H}$, the reproducing kernel Hilbert space is based on the mapping κ. Many choices of κ exist, some of the more popular choices are the Gaussian radial kernel $\kappa(\boldsymbol{x}_1, \boldsymbol{x}_2) = e^{-\gamma\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2}$, where $\gamma > 0$ and the polynomial kernel $\kappa(\boldsymbol{x}_1, \boldsymbol{x}_2) = (\boldsymbol{x}_1^\mathsf{T}\boldsymbol{x}_2 + c)^r$, where r is a positive integer. For the Gaussian radial kernel, the choice of γ is discussed in Li et al. (2011). Let $\boldsymbol{Q}_n = \boldsymbol{I}_n - \boldsymbol{J}_n/n$, where $\boldsymbol{I}_n$ is the $n \times n$ identity matrix and $\boldsymbol{J}_n$ is an $n \times n$ matrix with entries 1. We can use proposition 1 , where for our problem we need $\boldsymbol{\Psi}^\mathsf{T}\boldsymbol{\Psi}$, where $\boldsymbol{\Psi} = \boldsymbol{W} = (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_n)$ to estimate the eigenfunctions of Σ_n. Since $\boldsymbol{\omega}_i$ is an eigenvector of $\boldsymbol{Q}_n\boldsymbol{K}_n\boldsymbol{Q}_n$, $\boldsymbol{\Psi}^\mathsf{T}\boldsymbol{\Psi}$ becomes close to an identity matrix. Therefore the objective function in (4.11) becomes independent of $\boldsymbol{X}$. For this reason we choose $\boldsymbol{\Psi} = \boldsymbol{K}_n^{1/2}\boldsymbol{W}$. Therefore, the kernel PDWD estimation algorithm is as follows:

1. (Optional) Marginally standardise $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$. This step can be omitted if the components of $\boldsymbol{X}_i$ have similar variances.

2. Choose a kernel κ and create the kernel matrix $\boldsymbol{K}$. Calculate $\boldsymbol{\Psi} = \boldsymbol{K}_n^{1/2}\boldsymbol{W}$.

3. Divide the sample according to LVR or OVA. For each set of slices compute the coefficient vectors $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{\tilde{h}}$ using the kernel DWD algorithm with $\boldsymbol{K}$ replaced with $\boldsymbol{\Psi}^\mathsf{T}\boldsymbol{\Psi}$. For LVR $\tilde{h} = h - 1$ and for OVA $\tilde{h} = \binom{h}{2}$.

4. The sufficient predictors are equivalent to the first d eigenvectors $\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d$ of the matrix $\sum_{i=1}^{\tilde{h}} \boldsymbol{c}_i\boldsymbol{c}_i^\mathsf{T}$.

4.2.3 Numerical studies

We consider the following models

$$\text{Model I:} \quad Y = \frac{X_1}{0.5 + (X_2 + 1)^2} + 0.2\epsilon$$

$$\text{Model II:} \quad Y = (X_1^2 + X_2^2)^{1/2}\log((X_1^2 + X_2^2)^{1/2}) + 0.2\epsilon$$

where $X \sim \mathrm{N}(\boldsymbol{0}, I_p)$, $\epsilon \sim \mathrm{N}(0,1)$. For this choice of models, we only need to compare KPDWD to KPSVM, since these are the models used in Li et al. (2011). In the same format as Li et al. (2011) we will use the absolute value of Spearman's correlation to measure the closeness of the predictors to the true predictors.

We choose $n = 100$, $\lambda = 1$, $p = 10, 20, 30$ and $h = 20$. For Spearman's correlation, the numbers are between 0 and 1, where larger numbers indicate a higher performance. Using the Gaussian kernel basis, Table 4.1 shows that kernel PDWD outperforms kernel PSVM for both models. It is also clear that the performance of kernel PDWD remains good as p increases.

Model	p	KPSVM	KPDWD
I	10	0.91 (0.012)	0.97 (0.009)
	20	0.86 (0.029)	0.97 (0.015)
	30	0.84 (0.033)	0.97 (0.014)
II	10	0.90 (0.018)	0.92 (0.017)
	20	0.82 (0.037)	0.93 (0.020)
	30	0.78 (0.035)	0.93 (0.019)

Table 4.1: Comparison of estimation performance between KPSVM and KPDWD. The table reports the mean performance of 100 iterations (standard errors in parenthesis) for the two methods.

Chapter 5

Parallel SDR

The methods previously mentioned approach SDR directly by evaluating the dataset as a whole. Many methods have been developed using the process of splitting the data into subsets with respect to the sample size, n. Some of these include Liquet and Saracco (2016). Another approach, proposed by Yin and Hilafu (2015), instead partitions the feature space by splitting the variables into subsets and performing a sequential method on the smaller subsets.

We propose a new approach to dimension reduction in the form of parallel programming through feature space partitioning. Similar to Yin and Hilafu (2015), we too partition the feature space however we propose splitting the variables into subsets and performing the method on multiple machines in parallel. This should have a positive impact on the elapsed speed of the method which will be extremely useful for high dimensional data.

Remark 5.1 *For this section we will often define subsets of Y and $\boldsymbol{X}$. A subset of the Y and $\boldsymbol{X}$ over the sample space will be denoted $Y_{(i)}$ and $\boldsymbol{X}_{(i)}$, for $i = 1, \ldots, g$. The sample size of $Y_{(i)}$ and $\boldsymbol{X}_{(i)}$, n_i, is defined as $1 \leq n_i \leq n$. A subset of $\boldsymbol{X}$ over the feature space will be denoted by $\boldsymbol{X}^{(j)}$, for $j = 1, \ldots, m$. The dimension of $\boldsymbol{X}^{(j)}$, p_j, is defined as $1 \leq p_j \leq p$.*

5.1 Literature review

As previously discussed, many methods already exist which split big data before performing SDR. These methods include splitting the sample space or splitting the feature space. The motivation and theory when splitting the sample space compared to the feature space is often very different so we will consider previous methods separately.

5.1.1 Separation of sample space

When the sample size of data is extremely large this can cause analytical complications with respect to both speed and memory. However as we have previously found large n usually produces more accurate results. Below we have included a description of some of the already produced SDR methods that involve sample space partitioning.

5.1.1.1 BIG-SIR

There have been multiple methods of separating the sample space and performing parallel programming versions of SDR. One in particular is BIG-SIR, introduced by Liquet and Saracco (2016). This algorithm was developed to tackle data where the sample size is much larger than the dimension. The BIG-SIR algorithm is as follows:

1. Split the sample space into g slices, to give $n = n_1 + \cdots + n_g$, $\boldsymbol{Y} = (\boldsymbol{Y}_{(1)}, \ldots, \boldsymbol{Y}_{(g)})$ and $\boldsymbol{X} = (\boldsymbol{X}_{(1)}, \ldots, \boldsymbol{X}_{(g)})$, such that $p < n_i$ for all $i = 1, \ldots, g$.

2. Perform SIR on each subset to produce the candidate matrix $\hat{\boldsymbol{M}}_{(i)}$ for all $i = 1, \ldots, g$.

3. Re-collect the data and calculate

$$\hat{\boldsymbol{M}} = \sum_{i=1}^{g} \frac{1}{n_i} \hat{\boldsymbol{M}}_{(i)} \hat{\boldsymbol{M}}_{(i)}^{\mathsf{T}}.$$

4. Finally we let $\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d$ be the d largest eigenvectors of $\hat{\boldsymbol{M}}$. Thus $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\Sigma}}_{xx}^{-1/2}(\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d)$ can be used to estimate the CDRS.

The motivation for this work was to find a method that could be applied to big data. Even though accuracy is not the primary goal of this method it is worth noting the implications that separating the sample has on the accuracy. We have found previously that the larger the sample size the better the accuracy of the method. Therefore, since this method relies upon separating the sample space, and thus reducing the size of n within each subset, the accuracy of this method is not as high as standard SIR.

5.1.1.2 Distributive PSVM

Distributive PSVM, introduced by Jin et al. (2019), produced a new form of PSVM which separates the sample space and distributes the data onto multiple machines. PSVM has been found to often produce more accurate results than other methods

but can be quite slow. The PSVM algorithm is a quadratic programming problem which has computational complexity of approximately $O(n^3)$. Therefore, the computational efficiency of PSVM is heavily dependant on the size of n.

This method was produced in an attempt to tackle dimension reduction for data where the sample size is large and much greater than the dimension. The author proposes two methods beginning with what they name the naive distributed estimation (ND-PSVM). This approach begins by partitioning the data samples into g subsets and performing PSVM on each subset. The estimation algorithm for ND-PSVM is as follows:

1. Partition the data samples into g disjoint subsets, to give $n = n_1 + \cdots + n_g$, such that the jth subset contains the data $(\boldsymbol{X}_{(j)}, Y_{(j)})$ and $p < n_i$ for all $i = 1, \ldots, g$. There is no requirement for the n_i's to be equal. Since the accuracy of the estimator correlates with the size of n it is best for n to be as large as possible. Therefore, the accuracy of the estimator will be partially dependant on the size of the smallest n_i. Choosing equal n_i's will ensure the greatest accuracy for the number of subsets.

2. Perform PSVM on each subset to produce the candidate matrix $\hat{\boldsymbol{M}}_{(j)}$ for all $j = 1, \ldots, g$.

3. Gather the data and calculate
$$\hat{\boldsymbol{M}} = \sum_{j=1}^{g} \frac{1}{g} \hat{\boldsymbol{M}}_{(j)}.$$

4. Finally we let $\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d$ be the d largest eigenvectors of $\hat{\boldsymbol{M}}$. Thus $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\Sigma}}_{xx}^{-1/2}(\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d)$ can be used to estimate the CDRS.

This method is very similar to BIG-SIR discussed previously with the main difference being the implementation of PSVM as opposed to SIR. Therefore, similar to BIG-SIR, the reduction of the sample size within each subset leads to a negative impact on the accuracy of the method compared with the method applied to the entire data.

As previously discussed this work consisted of two approaches for sample space partitioning. The second approach, named refined distributed estimation (RD-PSVM), instead approximates the hinge loss function $u^+ = \max(u, 0)$ within the PSVM objective function. This is done using the smooth function $K_r(u) = uH(u/r)$ where H is a smooth and differentiable function satisfying
$$H(u) = \begin{cases} 1, & \text{for } u \geq 1 \\ 0, & \text{for } u \leq -1 \end{cases}.$$

Let $\boldsymbol{\theta} = (\boldsymbol{\psi}^\mathsf{T}, t)^\mathsf{T}$ and $\boldsymbol{X}^* = (\boldsymbol{X}^\mathsf{T}, -1)^\mathsf{T}$, then the estimation algorithm becomes:

1. Partition the data samples into g disjoint subsets, to give $n = n_1 + \cdots + n_g$, such that the jth subset contains the data $(\boldsymbol{X}_{(j)}, Y_{(j)})$ and $p < n_i$ for all $i = 1, \ldots, g$.

2. Calculate the following for $j = 1, \ldots, g$, where $\mathcal{I}_j$ represents the set of samples in the jth subset:

$$\hat{U}_{(j)} = n^{-1} \sum_{i \in \mathcal{I}_j} X_i^* X_i^{*\mathsf{T}}, \qquad \hat{W}_{(j)} = n^{-1} \sum_{i \in \mathcal{I}_j} X_i^* X_i^{*\mathsf{T}} H'((1 - \tilde{Y}_i \boldsymbol{\theta}_{(0)}^\mathsf{T} X_i^*)/r)/r,$$

$$\hat{V}_{(j)} = n^{-1} \sum_{i \in \mathcal{I}_j} X_i^* \tilde{Y}_i \left[H((1 - \tilde{Y}_i \boldsymbol{\theta}_{(0)}^\mathsf{T} X_i^*)/r) + H'((1 - \tilde{Y}_i \boldsymbol{\theta}_{(0)}^\mathsf{T} X_i^*)/r)/r \right],$$

   where $\boldsymbol{\theta}_{(0)}$ is a 'good' initial value, where a suggested initial value is the estimator $\hat{\boldsymbol{\theta}}_j$ of the jth subset of data.

3. Calculate $\hat{\boldsymbol{\theta}}$ for each slice as

$$\hat{\boldsymbol{\theta}} = \left[\sum_{j=1}^{g} \left(\hat{W}_j + 2\lambda^{-1} \mathrm{diag}(\hat{U}_j, 0) \right) \right]^{-1} \sum_{j=1}^{g} \hat{V}_j.$$

4. Compute the candidate matrix as

$$\hat{M} = \sum_{i=1}^{h} \hat{\boldsymbol{\psi}}_i \hat{\boldsymbol{\psi}}_i^\mathsf{T}$$

   using $\hat{\boldsymbol{\theta}}$, where h is the number of slices.

5. Finally we let $\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d$ be the d largest eigenvectors of $\hat{\boldsymbol{M}}$. Thus $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\Sigma}}_{xx}^{-1/2} (\hat{\boldsymbol{v}}_1, \ldots, \hat{\boldsymbol{v}}_d)$ can be used to estimate the CDRS.

5.1.2 Separation of the feature space

Separation of the feature space may also be required if p is extremely large however the formulation of such methods is commonly motivated by the restrictions on the dimension of the data enforced by the sample size. This being, many dimension reduction methods require $p < n$. The desire is that by splitting the feature space, only the dimension of the subsets of the data need to be less than n. Therefore by separating the feature space before performing traditional dimension reduction methods we hope to avoid this restriction. We aim to investigate the implication this will have on the efficiency of classical methods.

5.1.2.1 DECOrrelated feature space partitioning for distributed sparse regression

The work proposed by, Wang et al. (2016) is not a method of SDR but rather a method for variable selection which will form the groundwork for the new method of SDR we are developing. Variable selection and dimension reduction have similar aims but are slightly different.

Variable selection aims to find a subset of the original data which contains the variables that hold the most information about the response. Dimension reduction on the other hand creates new variables made of combinations of the original variables. To better understand this we shall consider the model $Y = X_1(X_2 + X_3) + \epsilon$. Using variable selection we would determine three variables of interest, X_1, X_2 and X_3. Whereas using dimension reduction we would only find two directions, or dimension, which are X_1 and $X_2 + X_3$.

The basic idea of this method is to decorrelate the design matrix and partition the feature space into m subsets. Traditional variable selection techniques can then be performed of the subsets. A key benefit of this method is the potential for parallel programming, meaning that the desired variable selection technique can be performed on each subset on separate computers and therefore reduce elapsed computation time.

Consider the linear regression model

$$Y = \boldsymbol{X}\boldsymbol{\beta} + \epsilon. \tag{5.1}$$

Let us consider the singular value decomposition of the design matrix as $\boldsymbol{X} = UDV^{\mathsf{T}}$, where U is an $n \times p$ matrix, D is a $p \times p$ diagonal matrix and V is a $p \times p$ orthogonal matrix. Then multiplying (5.1) by $\sqrt{p}D^{-1}U^{\mathsf{T}}$ on the left, we get

$$\sqrt{p}D^{-1}U^{\mathsf{T}}Y = \sqrt{p}V^{\mathsf{T}}\boldsymbol{\beta} + \sqrt{p}D^{-1}U^{\mathsf{T}}\epsilon \tag{5.2}$$

or

$$\tilde{Y} = \boldsymbol{\beta}\tilde{\boldsymbol{X}}\boldsymbol{\beta} + \tilde{\epsilon}, \tag{5.3}$$

where $\tilde{Y} = \sqrt{p}D^{-1}U^{\mathsf{T}}Y$, $\tilde{\boldsymbol{X}} = \sqrt{p}V^{\mathsf{T}}$ and $\tilde{\epsilon} = \sqrt{p}D^{-1}U^{\mathsf{T}}\epsilon$. It is now clear that $\tilde{\boldsymbol{X}}$ is mutually orthogonal. The decorrelation step can also be performed by instead multiplying by $(\boldsymbol{X}\boldsymbol{X}^{\mathsf{T}}/p)^{-1/2}$. The new feature matrix can now be split into m subsets and hence the estimation algorithm is as follows:

1. Decorrelate the data as specified above and split the new feature matrix into m subsets $\tilde{\boldsymbol{X}}^{(1)}, \ldots, \tilde{\boldsymbol{X}}^{(m)}$.

2. Compute $\hat{\boldsymbol{\beta}}^{(i)}$ for each subset, using the desired variable selection technique.

3. Combine the $\hat{\boldsymbol{\beta}}^{(i)}$'s to produce

$$\hat{\boldsymbol{\beta}} = \begin{pmatrix} \hat{\boldsymbol{\beta}}^{(1)} \\ \vdots \\ \hat{\boldsymbol{\beta}}^{(m)} \end{pmatrix}.$$

5.1.2.2 Sequential SDR

The work by Yin and Hilafu (2015) is different from others previously discussed as its primary goal is to tackle the problem of $p \gg n$. Therefore, the aim is to reduce p so that the dimension is less than n and then use a standard dimension reduction technique to find the sufficient dimension reduction subspace. Let $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ be random vectors and $R(\boldsymbol{X}_1)$ be a vector function of $\boldsymbol{X}_1$. We begin with 3 statements

(a) $\boldsymbol{X}_1 \perp\!\!\!\perp (\boldsymbol{X}_2, Y) | R(\boldsymbol{X}_1)$,

(b) $\boldsymbol{X}_1 \perp\!\!\!\perp \boldsymbol{X}_2 | \{R(\boldsymbol{X}_1), Y\}$ and $\boldsymbol{X}_1 \perp\!\!\!\perp Y | R(\boldsymbol{X}_1)$,

(c) $\boldsymbol{X}_1 \perp\!\!\!\perp Y | \{R(\boldsymbol{X}_1), \boldsymbol{X}_2\}$.

Proposition 1 of Yin and Hilafu (2015) states that either (a) or (b) imply (c). This framework consists of two separate paths, depending on the choice of statement (a) or statement (b). Statement (a) is considered the better choice for a quantitative response variable, whereas statement (b) would be the desired choice if the response variable is categorical. From now on the paths shall be referred to as Path I and Path II, respectively.

Path I: Let $\boldsymbol{X} = (\boldsymbol{X}^{(1)\mathsf{T}}, \ldots, \boldsymbol{X}^{(m)\mathsf{T}})^{\mathsf{T}}$ and $R(\boldsymbol{X}^{(1)}) = \boldsymbol{\beta}^{(1)\mathsf{T}} \boldsymbol{X}^{(1)}$. Then the aim is to estimate $S_{\hat{\boldsymbol{Y}}, \boldsymbol{X}^{(1)}}$, where the response variable is multivariate. There are many methods for estimating this subspace, but Yin and Hilafu chose to use Projective Resampling Sliced Inverse Regression (PRSIR) developed by Li et al. (2008). The estimation procedure (taken from Yin and Hilafu (2015)) is as follows:

1. Decompose $\boldsymbol{X} \in \mathbb{R}^p$ into $\boldsymbol{X}^{\mathsf{T}} = (\boldsymbol{X}^{(1)\mathsf{T}}, \boldsymbol{X}^{(2)\mathsf{T}})$, where $\boldsymbol{X}^{(1)}$ is a $p_1 \times 1$ vector such that $n > p_1$. Consider the problem of estimating $\boldsymbol{X}^{(1)} \perp\!\!\!\perp (\boldsymbol{X}^{(2)}, Y) | \boldsymbol{\beta}^{(1)\mathsf{T}} \boldsymbol{X}^{(1)}$.

2. Apply PRSIR to the problem of $\boldsymbol{Y}^{\mathsf{T}} = (\boldsymbol{X}^{(2)\mathsf{T}}, Y) | \boldsymbol{X}^{(1)}$ and find the reduced variable $\boldsymbol{\beta}^{(1)\mathsf{T}} \boldsymbol{X}^{(1)}$.

3. Replace the predictor $\boldsymbol{X}$ by $(\boldsymbol{\beta}^{(1)\mathsf{T}} \boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)})$ and go back to step 1.

Repeat steps 1-3 until all the variables in the original predictor vector $\boldsymbol{X}$ have been used in step 1.

Path II: In path II both $\boldsymbol{X}^{(1)} \perp\!\!\!\perp \boldsymbol{X}^{(2)} | (\boldsymbol{\beta}^{(1)\mathsf{T}} \boldsymbol{X}^{(1)}, Y)$ and $\boldsymbol{X}^{(1)} \perp\!\!\!\perp Y | \boldsymbol{\beta}^{(1)\mathsf{T}} \boldsymbol{X}^{(1)}$ need to be considered. The first is to identify the partial CDRS $S^{(Y)}_{\boldsymbol{X}^{(2)} | \boldsymbol{X}^{(1)}}$, where

$S^{(Y)}_{\boldsymbol{X}^{(2)}|\boldsymbol{X}^{(1)}}$ denotes the dimension reduction space spanned by $\boldsymbol{\beta}^{(1)}$ such that $\boldsymbol{X}^{(1)} \perp\!\!\!\perp Y|\boldsymbol{\beta}^{(1)\mathsf{T}}\boldsymbol{X}^{(1)}$, and has multivariate response. The second is to identify the usual CDRS, $S_{Y|\boldsymbol{X}^{(1)}}$. Here Projective Resampling Partial Sliced Inverse Regression (PRPSIR), proposed by Hilafu and Yin (2013), is used. The estimation procedure (again taken from Yin and Hilafu (2015)) is as follows:

1. Decompose $\boldsymbol{X} \in \mathbb{R}^p$ into $\boldsymbol{X}^{\mathsf{T}} = (\boldsymbol{X}^{(1)\mathsf{T}}, \boldsymbol{X}^{(2)\mathsf{T}})$ where $\boldsymbol{X}^{(1)}$ is a $p_1 \times 1$ vector such that $n > p_1$. Consider the problem of estimating $S^{(Y)}_{\boldsymbol{X}^{(2)}|\boldsymbol{X}^{(1)}}$ and $S_{Y|\boldsymbol{X}^{(1)}}$.

2. Apply PRPSIR to the problem of $S^{(Y)}_{\boldsymbol{X}^{(2)}|\boldsymbol{X}^{(1)}}$ and find the reduced variable $\boldsymbol{\alpha}_1^{\mathsf{T}}\boldsymbol{X}^{(1)}$.

3. Apply SIR to the problem of $S_{Y|\boldsymbol{X}^{(1)}}$ and find the reduced variable $\boldsymbol{\alpha}_2^{\mathsf{T}}\boldsymbol{X}^{(1)}$.

4. Set $\boldsymbol{\beta}^{(1)} = (\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2)$, replace the predictor $\boldsymbol{X}$ by with $(\boldsymbol{\beta}^{(1)\mathsf{T}}\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)})$ and go back to step 1.

When combining $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$ to obtain $\boldsymbol{\beta}^{(1)}$, the singular value decomposition method can then be used to remove redundant directions in case $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$ have common estimated directions. Repeat steps 1-4 until all variables in the original predictor vector have been used in step 1.

5.1.2.3 Groupwise

Often in practice, data occurs where the predictors naturally fall into several groups. Li et al. (2010) and Guo et al. (2015) developed a method, named groupwise SDR which attempts to estimate $S_{Y|\boldsymbol{X}}$ by performing SDR on the naturally forming groups. For this type of data we instead aim to find

$$Y \perp\!\!\!\perp \boldsymbol{X}|(\boldsymbol{\beta}^{(1)\mathsf{T}}\boldsymbol{X}^{(1)}, \ldots, \boldsymbol{\beta}^{(m)\mathsf{T}}\boldsymbol{X}^{(m)}), \tag{5.4}$$

where we assume there are m groups of predictors and $\text{span}(\boldsymbol{\beta}^{(i)})$ forms the column space of $S_{Y|\boldsymbol{X}^{(i)}}$. This gives

$$\bigcup_{i=1}^{m} S_{Y|\boldsymbol{X}^{(i)}} \supseteq S_{Y|\boldsymbol{X}},$$

where an equality occurs if $\boldsymbol{X}^{(i)} \perp\!\!\!\perp \boldsymbol{X}^{(j)}$ for all $i, j = 1, \ldots, m$. An added assumption of the work by Li et al. (2010) is that $d_i > 0$, where d_i is the estimated dimension size in each group. That is there are directions of interest within each naturally forming group.

5.1.2.4 Sparse SIR via LASSO

The work by Lin et al. (2019) introduces an efficient LASSO variant of SIR for $p \gg n$ problems. Generally in SDR literature $\boldsymbol{\beta} \propto \boldsymbol{\Sigma}^{-1}\boldsymbol{\eta}$, where $\boldsymbol{\Sigma}$ is the general covariance matrix of $\boldsymbol{X}$ and $\boldsymbol{\eta}$ are the leading eigenvalues of the candidate matrix. The common approach of estimating $\boldsymbol{\Sigma}^{-1}\boldsymbol{\eta}$ is to estimate $\boldsymbol{\Sigma}^{-1}$ and $\boldsymbol{\eta}$ separately, however when $p > n$, $\boldsymbol{\Sigma}$ is singular and therefore not invertible. Another approach to avoid directly estimating $\boldsymbol{\Sigma}^{-1}$ is by solving an L_1 penalisation problem.

To begin, we define the classic SIR candidate matrix to be

$$\hat{V} = \frac{1}{h}\boldsymbol{X}_h\boldsymbol{X}_h^{\mathsf{T}}, \tag{5.5}$$

where $\boldsymbol{X}_h$ is a $p \times h$ matrix formed by the h sample means. Now define $\hat{\lambda}$ to be the largest eigenvalue of $\hat{V}$, $\hat{\boldsymbol{\eta}}$ be it's respective eigenvector and let $\boldsymbol{M} = \boldsymbol{I}_h \otimes \mathbf{1}_c$, where $\mathbf{1}$ is a $n_j \times 1$ vector of 1's and $c = n/h$. Therefore,

$$\hat{\lambda}\hat{\boldsymbol{\eta}} = \frac{1}{h}\boldsymbol{X}_h\boldsymbol{X}_h^{\mathsf{T}}\hat{\boldsymbol{\eta}} = \frac{1}{nc}\boldsymbol{X}\boldsymbol{M}\boldsymbol{M}^{\mathsf{T}}\boldsymbol{X}^{\mathsf{T}}\hat{\boldsymbol{\eta}}.$$

Finally defining

$$\boldsymbol{A} = \frac{1}{c\hat{\lambda}}\boldsymbol{M}\boldsymbol{M}^{\mathsf{T}}\boldsymbol{X}^{\mathsf{T}}\hat{\boldsymbol{\eta}}$$

gives $\hat{\boldsymbol{\eta}} = n^{-1}\boldsymbol{X}\boldsymbol{A}$, which yields

$$\frac{1}{n}\boldsymbol{X}\boldsymbol{A} \propto \frac{1}{n}\boldsymbol{X}\boldsymbol{X}^{\mathsf{T}}\boldsymbol{\beta},$$

where $n^{-1}\boldsymbol{X}\boldsymbol{X}^{\mathsf{T}}$ is used as an approximation of $\boldsymbol{\Sigma}$.

Using LASSO regression, we can recover a sparse estimate of $\boldsymbol{\beta}$, where $\boldsymbol{\beta}$ is estimated to be the minimiser of

$$\frac{1}{2n}\|\boldsymbol{A} - \boldsymbol{X}^{\mathsf{T}}\boldsymbol{\beta}\|_2^2 + \mu\|\boldsymbol{\beta}\|_1.$$

Therefore the estimation algorithm, described in Lin et al. (2019), takes the form:

1. Let $\hat{\lambda}_i$ and $\hat{\boldsymbol{\eta}}_i$, for $i = 1, \ldots, d$, be the d leading eigenvalues and eigenvectors of $\hat{V}$, respectively.

2. Let $\boldsymbol{A} = c^{-1}\boldsymbol{M}\boldsymbol{M}^{\mathsf{T}}\boldsymbol{X}^{\mathsf{T}}\hat{\boldsymbol{\eta}}\ \mathrm{diag}(\lambda_1^{-1}, \ldots, \lambda_d^{-1})$.

3. For each $i = 1, \ldots, d$, solve the LASSO optimisation problem

$$\hat{\boldsymbol{\eta}}_i = \arg\min \frac{1}{2n}\|\boldsymbol{A}_{*i} - \boldsymbol{X}^{\mathsf{T}}\boldsymbol{\beta}\|_2^2 + \mu_i|\boldsymbol{\beta}|_1,$$

   where $\mu_i = C\sqrt{\frac{\log(p)}{n\lambda_i}}$ for sufficiently large constant C and $\boldsymbol{A}_{*i}$ is the ith column of $\boldsymbol{A}$.

4. Let $\hat{\boldsymbol{B}}$ be the matrix formed by $\hat{\boldsymbol{\beta}}_1, \ldots, \hat{\boldsymbol{\beta}}_d$. The estimator of $P_{\boldsymbol{\beta}}$ is given by $P_{\hat{\boldsymbol{B}}}$.

Further to this work, Pircalabelu and Artemiou (2021) adapted this method for PSVM. The PSVM objective function also depends on $\boldsymbol{\Sigma}$ which needs to be estimated. For sparse PSVM via LASSO, $\boldsymbol{\Sigma}$ is also estimated using LASSO.

5.2 SDR by decorrelating variables

The work by Li et al. (2010) introduced a parallel programming problem for separating the feature space of grouped data. Similar to the approach proposed by Wang et al. (2016), we aim to develop a new method for SDR by first decorrelating the data, for all types of vector data. Many methods of SDR begin by first standardising the feature matrix before applying more steps. We propose to take advantage of this common standardisation step to instead decorrelate the variable. If $\boldsymbol{Z} = \boldsymbol{\Sigma}^{1/2}(\boldsymbol{X} - \bar{\boldsymbol{X}})$ then the covariance matrix of $\boldsymbol{Z}$ is equal to $\boldsymbol{I}_p$. Therefore

$$\boldsymbol{Z}_i \perp\!\!\!\perp (\boldsymbol{X}_1, \ldots, \boldsymbol{Z}_{i-1}, \boldsymbol{Z}_{i+1}, \ldots, \boldsymbol{Z}_p) \text{ for } i = 1, \ldots, p. \tag{5.6}$$

Before we continue, we shall define some notation. Let m denote the number of subsets and let $\boldsymbol{Z}^{(i)}$ denote the ith subset of $\boldsymbol{Z}$, where $\boldsymbol{Z} = (\boldsymbol{Z}^{(1)\mathsf{T}}, \ldots, \boldsymbol{Z}^{(m)\mathsf{T}})^{\mathsf{T}}$. Then we obtain

$$\boldsymbol{Z}^{(i)} \perp\!\!\!\perp (\boldsymbol{Z}^{(1)}, \ldots, \boldsymbol{Z}^{(i-1)}, \boldsymbol{Z}^{(i+1)}, \ldots, \boldsymbol{Z}^{(m)}) \text{ for } i = 1, \ldots, m. \tag{5.7}$$

Next we can perform a standard SDR methods on $\boldsymbol{Z}^{(i)}$ to yield

$$Y \perp\!\!\!\perp \boldsymbol{Z}^{(i)} | \tilde{\boldsymbol{\beta}}^{(i)\mathsf{T}} \boldsymbol{Z}^{(i)}. \tag{5.8}$$

Now let $\boldsymbol{A} = \text{diag}(\tilde{\boldsymbol{\beta}}^{(i)^{\mathsf{T}}})$ which has dimension $\tilde{d} = \tilde{d}_1 + \cdots + \tilde{d}_m$. Combining (5.7) and (5.8) gives,

$$Y \perp\!\!\!\perp \boldsymbol{Z} | \boldsymbol{A}^{\mathsf{T}} \boldsymbol{Z}. \tag{5.9}$$

Here $d \leq \tilde{d} \leq md$ and hence this process only predicts the principal directions within each subset. Now since we have established that the information, we require from $\boldsymbol{Z}$ remains within $\boldsymbol{A}^{\mathsf{T}}\boldsymbol{Z}$, further evaluation can be performed on $\boldsymbol{A}^{\mathsf{T}}\boldsymbol{Z}$ instead of $\boldsymbol{Z}$, without loss of information.

If one of the combinations of variables lies within separate subsets, then the steps so far will find separate directions for the variables in each subset. This means that even though the space spanned by the columns of $\boldsymbol{A}$ is a dimension reduction subspace, it may not be the minimal dimension reduction subspace. To find the

minimal dimension reduction subspace, we find the d most important directions of $\boldsymbol{A}^\mathsf{T}\boldsymbol{Z}$ using traditional methods to give

$$Y \perp\!\!\!\perp \boldsymbol{A}^\mathsf{T}\boldsymbol{Z} | \boldsymbol{B}^\mathsf{T}\boldsymbol{A}^\mathsf{T}\boldsymbol{Z} \Rightarrow Y \perp\!\!\!\perp \boldsymbol{Z} | (\boldsymbol{AB})^\mathsf{T}\boldsymbol{Z}$$

since $\mathbb{P}(Y|\boldsymbol{Z}) = \mathbb{P}(Y|\boldsymbol{A}^\mathsf{T}\boldsymbol{Z})$. Therefore, we choose $\boldsymbol{\beta} = \hat{\boldsymbol{\Sigma}}_{xx}^{-1/2}\boldsymbol{AB}$ to give

$$Y \perp\!\!\!\perp \boldsymbol{X} | \boldsymbol{\beta}^\mathsf{T}\boldsymbol{X}.$$

.

5.2.1 Estimation algorithm

The method we proposed has been clearly outlined in the previous section. Below we include a step-by-step algorithm of this method.

1. Decorrelate the variables by standardising $\boldsymbol{X}$ and split the feature space of $\boldsymbol{X}$ into m subsets such that $\boldsymbol{X} = (\boldsymbol{X}^{(1)\mathsf{T}}, \ldots, \boldsymbol{X}^{(m)\mathsf{T}})^\mathsf{T}$ where the dimension of $\boldsymbol{X}^{(i)}$ is p_i and $p = p_1 + \cdots + p_m$.

2. Perform the desired SDR method on each subset to obtain $\hat{\boldsymbol{A}}^{(i)}$ of size $p_i \times \tilde{d}$, where $d \leq \tilde{d} \leq md$ and $\tilde{d} = \tilde{d}_1 + \cdots + \tilde{d}_m$. Now form a matrix $\hat{\boldsymbol{A}}$ as follows

$$\hat{\boldsymbol{A}} = \begin{pmatrix} \hat{\boldsymbol{A}}^{(1)} & \boldsymbol{0}_{p_1 \times d_1} & \cdots & \boldsymbol{0}_{p_1 \times d_1} \\ \boldsymbol{0}_{p_2 \times d_2} & \hat{\boldsymbol{A}}^{(2)} & \cdots & \boldsymbol{0}_{p_2 \times d_2} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0}_{p_m \times d_m} & \boldsymbol{0}_{p_m \times d_m} & \cdots & \hat{\boldsymbol{A}}^{(m)} \end{pmatrix}.$$

3. Perform the chosen SDR method on $\hat{\boldsymbol{A}}^\mathsf{T}\boldsymbol{X}$ to obtain $\hat{\boldsymbol{B}}$ of size $md \times d$.

4. Let $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\Sigma}}_{xx}^{-1/2}\hat{\boldsymbol{A}}\hat{\boldsymbol{B}}$ and so we can use the subspace spanned by $\hat{\boldsymbol{\beta}} = (\hat{\boldsymbol{\beta}}_1, \ldots, \hat{\boldsymbol{\beta}}_d)$ to estimate the CDRS.

5.2.2 Separating the sample space and feature space

Using the ideas developed by Liquet and Saracco (2016) we can now extend our method so that we can separate both the sample space and the feature space, as follows:

1. Decorrelate the variables by standardising $\boldsymbol{X}$ and split the feature space of $\boldsymbol{X}$ into m subsets such that $\boldsymbol{X} = (\boldsymbol{X}^{(1)\mathsf{T}}, \ldots, \boldsymbol{X}^{(m)\mathsf{T}})^\mathsf{T}$ where the dimension of $\boldsymbol{X}^{(i)}$ is p_i and $p = p_1 + \cdots + p_m$.

2. Split the sample space into g slices, to give $n = n_1 + \cdots + n_g$, $\boldsymbol{Y} = (\boldsymbol{Y}_{(1)}, \ldots, \boldsymbol{Y}_{(g)})$ and
$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_{(1)}^{(1)} & \cdots & \boldsymbol{X}_{(g)}^{(1)} \\ \vdots & \ddots & \vdots \\ \boldsymbol{X}_{(1)}^{(m)} & \cdots & \boldsymbol{X}_{(g)}^{(m)} \end{pmatrix},$$
such that $p < n_i$ for all $i = 1, \ldots, g$.

3. Perform the desired method on each subset to produce the candidate matrix $\hat{\boldsymbol{M}}_{(i)}^{(j)}$ for all $i = 1, \ldots, g$ and $j = 1, \ldots, m$.

4. Recollect the data and calculate
$$\hat{\boldsymbol{M}}^{(j)} = \sum_{i=1}^{g} \frac{1}{n_i} \hat{\boldsymbol{M}}_{(i)}^{(j)} \hat{\boldsymbol{M}}_{(i)}^{(j)\mathsf{T}}$$
for all $j = 1, \ldots, m$.

5. Finally, we let $\boldsymbol{\eta}^{(j)}$ be the d largest eigenvectors of $\hat{\boldsymbol{M}^{(j)}}$. Thus $\hat{\boldsymbol{A}}^{(j)} = \boldsymbol{\eta}^{(j)}$ for all $j = 1, \ldots, m$.

6. Using $\hat{\boldsymbol{A}}^{(j)}$ of size $p_j \times \tilde{d}$, where $d \leq \tilde{d} \leq md$ and $\tilde{d} = \tilde{d}_1 + \cdots + \tilde{d}_m$, form a matrix $\hat{\boldsymbol{A}}$ as follows
$$\hat{\boldsymbol{A}} = \begin{pmatrix} \hat{\boldsymbol{A}}^{(1)} & \boldsymbol{0}_{p_1 \times d_1} & \cdots & \boldsymbol{0}_{p_1 \times d_1} \\ \boldsymbol{0}_{p_2 \times d_2} & \hat{\boldsymbol{A}}^{(2)} & \cdots & \boldsymbol{0}_{p_2 \times d_2} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0}_{p_m \times d_m} & \boldsymbol{0}_{p_m \times d_m} & \cdots & \hat{\boldsymbol{A}}^{(m)} \end{pmatrix}.$$

7. Again separate $\hat{\boldsymbol{A}}^{\mathsf{T}}\boldsymbol{X}$ into g subsets as above and perform the desired method on each subset to obtain $\hat{\boldsymbol{N}}_{(i)}$ for all $i = 1, \ldots, g$. Calculate
$$\hat{\boldsymbol{N}} = \sum_{i=1}^{g} \frac{1}{n_i} \hat{\boldsymbol{N}}_{(i)} \boldsymbol{N}_{(i)}^{\mathsf{T}}.$$

8. Now $\hat{\boldsymbol{B}} = \boldsymbol{\nu}$, where $\boldsymbol{\nu}$ is the d largest eigenvectors of $\hat{\boldsymbol{N}}$ and $\hat{\boldsymbol{B}}$ is of size $md \times d$.

9. Let $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\Sigma}}_{xx}^{-1/2} \hat{\boldsymbol{A}} \hat{\boldsymbol{B}}$ and so the subspace spanned by $\hat{\boldsymbol{\beta}} = (\hat{\boldsymbol{\beta}}_1, \ldots, \hat{\boldsymbol{\beta}}_d)$ can be used to estimate the CDRS.

Many classical methods which do not attempt to separate the feature space are restricted such that they require $p < n$. Therefore, for each subset we require $p_i < n$. However, since we standardise first, which requires a non-singular covariance matrix, we too require $p < n$ and thus separating the hyperplane in this way does not allow us to perform the method on high dimension low sample size data.

5.2.3 Synthetic analysis

We have chosen to again use the distance measure defined in (2.37). This measure can take values from 0 to $\sqrt{2d}$, with lower numbers indicating the two vectors are closer together and therefore implying a higher level of accuracy. We will start by evaluating the performance of our proposed method through synthetic studies. Since we are splitting the data into m subsets, we once again need to evaluate our method when all of the features of interest fall into one subset and when the features of interest are separated into different subsets. For this reason, we will consider the following models with r to be defined later.

$$\text{Model I:} \quad Y = X_1 + X_r + 0.2\epsilon$$

$$\text{Model II:} \quad Y = \frac{X_1}{0.5 + (X_r + 1)^2} + 0.2\epsilon$$

$$\text{Model III:} \quad Y = X_1(X_1 + X_r + 1) + 0.2\epsilon$$

Since our method involves the separation of the feature space it is important that our numerical investigations include examples with correlation. Therefore, we choose $X \sim N(\mathbf{0}, \mathbf{\Sigma})$ and $\epsilon \sim N(\mathbf{0}, I_p)$, with $\Sigma_{ij} = s^{|i-j|}$. The following results have been obtained using $h = 20$ and $\lambda = 0.1$ for PSVM and PDWD.

5.2.3.1 Performance when combinations of variables fall in different subsets

As previously stated since we are separating the variables into different subsets, and then performing our method on each individual subset, we are interested in the performance of our method when all the important variables fall into the same subset and when they fall into different subsets. For this reason, we will consider the case when $r = 2$ and $r = p$. Tables 5.1 and 5.2 show the performance of our method for $s = 0$ and $s = 0.2$ respectively.

On first inspection of tables 5.1 and 5.2, we see little difference between the accuracy for $s = 0$ and $s = 0.2$. For model I, separating the variables for $r = 2$ and $r = p$ has a negative impact on the performance for $p < n$, and a slight increase in the performance for $p = n$. This is apparent for SIR, PDWD and PSVM, with minor variation between choices of r. For models II and III, separating the variables generally has a positive effect on the performance with a slight decrease occurring for larger choices of m.

Figure 5.1 shows the performance of our method for more choices of p and gives a clearer view of our method compared with no separation, with $n = 100$. Separation of the variables produces better results for 70.6% of the scenarios, where the mean distance of all results is 1.29 for no separation of variables and 1.23 for our method.

Model	p	m	SIR		PDWD		PSVM	
			$r = 2$	$r = p$	$r = 2$	$r = p$	$r = 2$	$r = p$
I	10	1	0.10 (0.022)	0.06 (0.013)	0.09 (0.033)	0.12 (0.042)	0.12 (0.016)	0.09 (0.008)
		2	0.16 (0.010)	0.12 (0.007)	0.14 (0.104)	0.17 (0.062)	0.13 (0.024)	0.13 (0.038)
		5	0.20 (0.017)	0.13 (0.023)	0.11 (0.021)	0.11 (0.040)	0.14 (0.010)	0.14 (0.012)
	50	1	0.43 (0.022)	0.29 (0.043)	0.35 (0.056)	0.44 (0.081)	0.44 (0.066)	0.39 (0.036)
		2	0.44 (0.031)	0.45 (0.029)	0.39 (0.074)	0.54 (0.019)	0.48 (0.036)	0.52 (0.164)
		5	0.45 (0.060)	0.42 (0.082)	0.51 (0.083)	0.47 (0.057)	0.49 (0.077)	0.47 (0.051)
	100	1	1.41 (0.012)	1.41 (0.002)	1.37 (0.027)	1.31 (0.020)	1.34 (0.084)	1.40 (0.014)
		5	1.30 (0.130)	1.35 (0.091)	1.35 (0.050)	1.34 (0.038)	1.33 (0.018)	1.37 (0.023)
		10	1.27 (0.114)	1.33 (0.089)	1.40 (0.007)	1.39 (0.012)	1.23 (0.126)	1.32 (0.047)
II	10	1	0.66 (0.063)	0.87 (0.061)	0.62 (0.002)	0.77 (0.230)	0.42 (0.181)	0.59 (0.027)
		2	0.87 (0.317)	0.66 (0.214)	0.76 (0.176)	0.67 (0.130)	0.44 (0.068)	0.55 (0.078)
		5	0.51 (0.022)	0.67 (0.118)	0.62 (0.002)	0.77 (0.230)	0.54 (0.196)	0.79 (0.133)
	50	1	1.67 (0.026)	1.62 (0.007)	1.33 (0.004)	1.44 (0.127)	1.51 (0.199)	1.50 (0.138)
		2	1.39 (0.196)	1.43 (0.194)	1.36 (0.015)	1.50 (0.116)	1.45 (0.010)	1.45 (0.062)
		5	1.52 (0.060)	1.31 (0.027)	1.29 (0.064)	1.39 (0.157)	1.31 (0.075)	1.37 (0.128)
	100	1	1.98 (0.019)	1.96 (0.011)	1.97 (0.011)	1.92 (0.046)	1.95 (0.060)	1.93 (0.040)
		5	1.96 (0.019)	1.94 (0.021)	1.97 (0.020)	1.91 (0.024)	1.95 (0.072)	1.94 (0.034)
		10	1.93 (0.011)	1.95 (0.002)	1.97 (0.008)	1.95 (0.052)	1.96 (0.017)	1.94 (0.020)
III	10	1	1.18 (0.497)	1.44 (0.304)	0.78 (0.100)	1.18 (0.042)	0.84 (0.005)	1.05 (0.171)
		2	0.96 (0.000)	1.02 (0.044)	0.53 (0.159)	0.80 (0.113)	1.05 (0.415)	0.86 (0.192)
		5	0.76 (0.189)	1.09 (0.140)	0.78 (0.100)	1.18 (0.042)	0.89 (0.005)	1.23 (0.453)
	50	1	1.84 (0.170)	1.88 (0.093)	1.78 (0.093)	1.82 (0.102)	1.91 (0.014)	1.87 (0.009)
		2	1.72 (0.082)	1.67 (0.042)	1.75 (0.206)	1.60 (0.041)	1.65 (0.064)	1.70 (0.179)
		5	1.67 (0.187)	1.64 (0.038)	1.72 (0.138)	1.78 (0.179)	1.73 (0.111)	1.76 (0.003)
	100	1	1.99 (0.002)	1.98 (0.004)	1.98 (0.017)	1.97 (0.022)	1.98 (0.014)	1.98 (0.009)
		5	1.99 (0.008)	1.97 (0.020)	1.99 (0.018)	1.97 (0.026)	1.97 (0.015)	1.97 (0.010)
		10	1.92 (0.074)	1.99 (0.001)	1.98 (0.010)	1.98 (0.008)	1.95 (0.028)	1.95 (0.008)

Table 5.1: Comparison of different amounts of subsets for PDWD, PSVM and SIR. The table shows the mean performances (standard deviation in parenthesis) of each method with $n = 100$ and $s = 0$ of 100 iterations.

Figure 5.1 indicates more clearly that separation of the variables produces better results more often than not separating the variables. We will now investigate whether specific choices model, p, m, r, s and classic method show different proportion of increased performance for separation of variables. Figure 5.2 shows the 720 scenarios by model, p, m, r, s and choice of classic method.

The choice of method, r and s has little impact on the number of results that our method had an improved performance. It is clear that our method introduces more performance improvements for models II and III, since it can be seen that most of the better performances occur for models II and III. Our method produced better results only 35.8% of the time for model I, whereas our method created improved results 86.3% of the time for model II and 89.6% of the time for model III. For the choices of p and m, the charts show that the number of results where our method shows a better performance increases, as p and m increase. For the choice of method, we see that for PDWD the points remain extremely close to the line which implies the separation of variables has little effect on the accuracy of the method. This trend can also be seen for PSVM however the points do show a larger spread than those

Model	p	m	SIR		PDWD		PSVM	
			$r = 2$	$r = p$	$r = 2$	$r = p$	$r = 2$	$r = p$
I	10	1	0.14 (0.009)	0.10 (0.006)	0.15 (0.019)	0.09 (0.014)	0.11 (0.041)	0.10 (0.014)
		2	0.16 (0.007)	0.13 (0.001)	0.18 (0.020)	0.19 (0.024)	0.14 (0.025)	0.15 (0.091)
		5	0.15 (0.007)	0.13 (0.027)	0.21 (0.026)	0.14 (0.042)	0.15 (0.060)	0.14 (0.022)
	50	1	0.32 (0.035)	0.29 (0.018)	0.52 (0.155)	0.36 (0.103)	0.64 (0.040)	0.52 (0.025)
		2	0.54 (0.041)	0.44 (0.025)	0.56 (0.036)	0.47 (0.184)	0.50 (0.031)	0.33 (0.058)
		5	0.55 (0.017)	0.47 (0.007)	0.50 (0.118)	0.40 (0.074)	0.64 (0.086)	0.47 (0.077)
	100	1	1.40 (0.021)	1.40 (0.017)	1.36 (0.037)	1.38 (0.030)	1.39 (0.030)	1.28 (0.179)
		5	1.32 (0.102)	1.15 (0.234)	1.31 (0.034)	1.38 (0.003)	1.39 (0.033)	1.28 (0.188)
		10	1.32 (0.055)	1.33 (0.007)	1.35 (0.006)	1.25 (0.072)	1.21 (0.055)	1.31 (0.084)
II	10	1	1.00 (0.327)	0.94 (0.197)	0.72 (0.115)	0.57 (0.085)	0.70 (0.157)	0.70 (0.040)
		2	0.96 (0.368)	0.92 (0.091)	0.80 (0.108)	0.71 (0.092)	0.55 (0.001)	0.67 (0.101)
		5	0.63 (0.078)	0.84 (0.150)	0.72 (0.115)	0.57 (0.085)	0.66 (0.061)	0.72 (0.037)
	50	1	1.58 (0.021)	1.74 (0.000)	1.37 (0.061)	1.50 (0.002)	1.54 (0.092)	1.46 (0.068)
		2	1.56 (0.005)	1.44 (0.162)	1.32 (0.078)	1.42 (0.144)	1.34 (0.195)	1.44 (0.117)
		5	1.36 (0.061)	1.49 (0.171)	1.33 (0.055)	1.50 (0.021)	1.51 (0.005)	1.34 (0.101)
	100	1	1.99 (0.007)	1.98 (0.012)	1.88 (0.072)	1.95 (0.047)	1.93 (0.034)	1.98 (0.010)
		5	1.96 (0.029)	1.96 (0.048)	1.89 (0.061)	1.95 (0.047)	1.94 (0.021)	1.97 (0.002)
		10	1.90 (0.091)	1.97 (0.009)	1.93 (0.089)	1.94 (0.011)	1.96 (0.011)	1.95 (0.017)
III	10	1	1.69 (0.053)	1.35 (0.325)	1.15 (0.153)	1.18 (0.063)	0.91 (0.608)	0.81 (0.081)
		2	0.94 (0.125)	0.65 (0.177)	1.38 (0.003)	1.05 (0.126)	0.85 (0.097)	0.96 (0.220)
		5	1.46 (0.332)	1.05 (0.583)	1.15 (0.153)	1.18 (0.063)	0.89 (0.576)	0.98 (0.064)
	50	1	1.92 (0.014)	1.93 (0.014)	1.77 (0.139)	1.68 (0.085)	1.84 (0.100)	1.86 (0.029)
		2	1.79 (0.101)	1.80 (0.017)	1.78 (0.081)	1.90 (0.031)	1.82 (0.028)	1.61 (0.063)
		5	1.74 (0.090)	1.73 (0.131)	1.75 (0.167)	1.61 (0.133)	1.71 (0.150)	1.73 (0.129)
	100	1	1.98 (0.002)	1.98 (0.012)	1.99 (0.013)	1.96 (0.032)	1.98 (0.005)	1.95 (0.009)
		5	1.99 (0.003)	1.99 (0.001)	1.99 (0.002)	1.95 (0.031)	1.99 (0.004)	1.95 (0.005)
		10	1.98 (0.011)	1.96 (0.035)	1.97 (0.005)	1.95 (0.007)	1.97 (0.014)	1.97 (0.020)

Table 5.2: Comparison of different amounts of subsets for PDWD, PSVM and SIR. The table shows the mean performances (standard deviation in parenthesis) of each method with $n = 100$ and $s = 0.2$ of 100 iterations.

for PDWD.

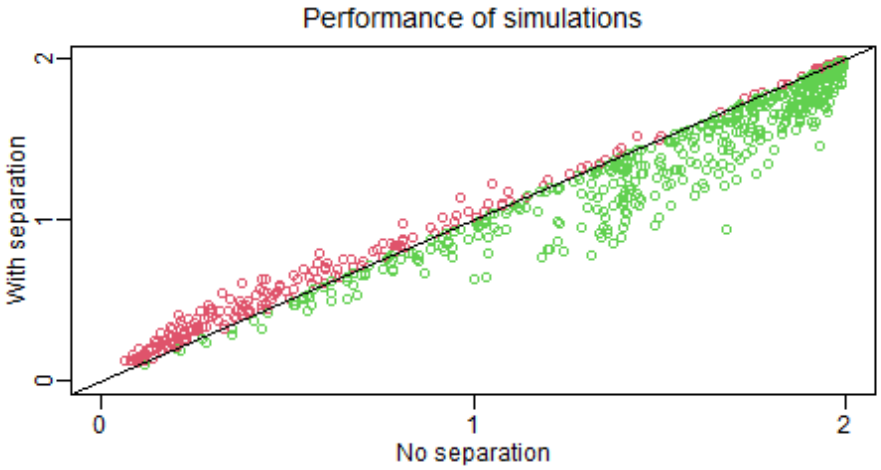

Figure 5.1: Performance of our method of 720 scenarios for all models and multiple choices of p, m, r, s, including results for SIR, PDWD and PSVM. Green points indicate results where the performance of our method was better than not separating and red show the alternative.

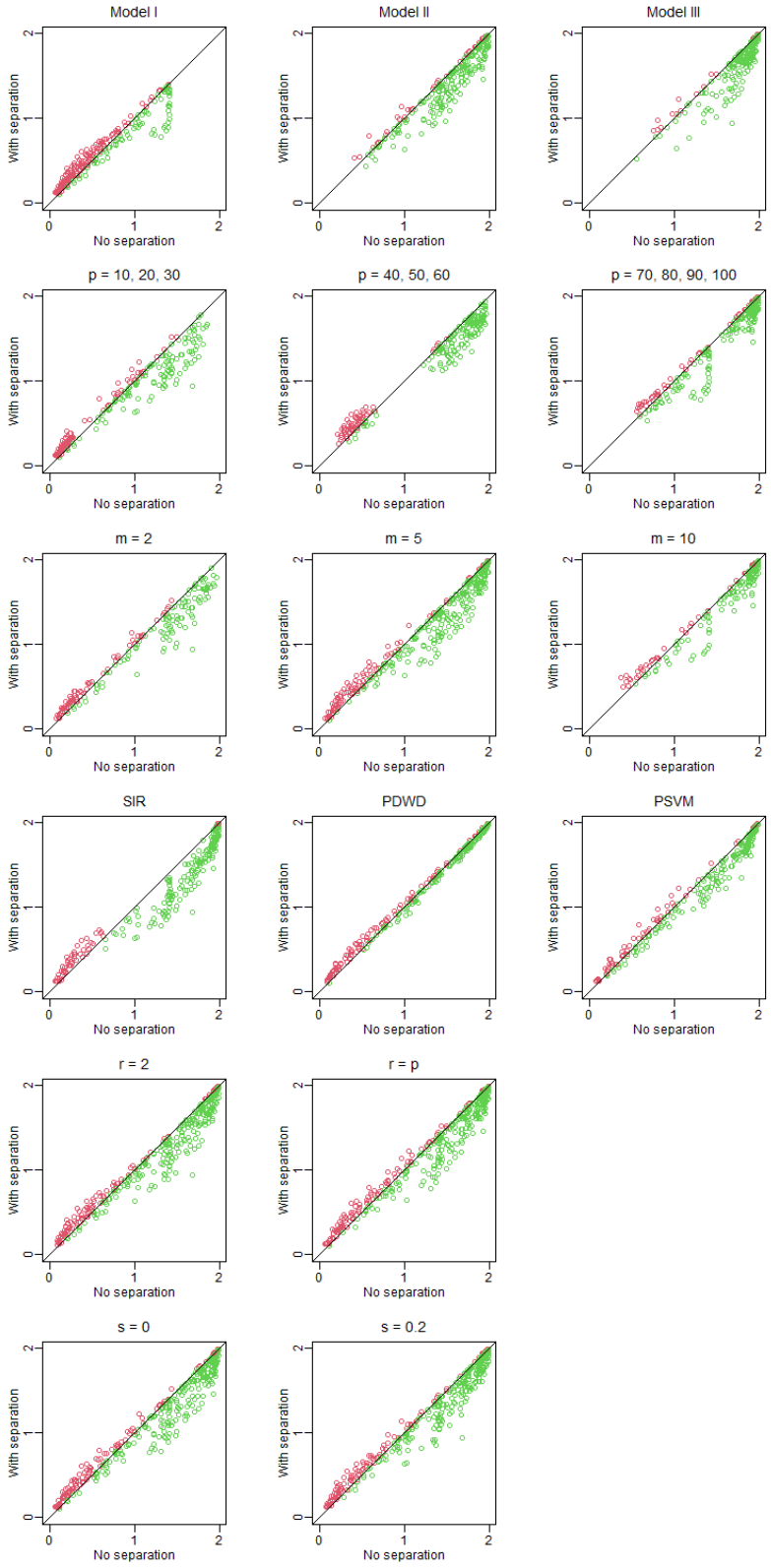

Figure 5.2: Performance of our method of 720 scenarios by model, p, m, r, s and choice of classic method. Green points indicate results where the performance of our method was better than not separating and red show the alternative.

5.2.3.2 Performance and time

Table 5.3 shows the distance and time for $r = 2$ and $s = 0$. We have included the distance for each of the cases to give a clearer picture of the overall performance of our method for each case. If we first consider SIR, which is generally not considered to have high computational complexity, we can see that the separation of variables causes an increase in the time of the method by the separation of the feature space. This is consistent across all p and for all models. However, for PDWD and PSVM we see much greater discrepancies with respect to time. The time taken for PDWD appears to be greatly improved by sub-setting the variables for larger p. Lastly, PSVM shows very similar results to PDWD, where the time is significantly improved by separating the variables for all choices of p.

Model	p	m	SIR		PDWD		PSVM	
			Distance	Time	Distance	Time	Distance	Time
I	50	1	0.06 (0.014)	0.05	0.08 (0.001)	0.22	0.07 (0.002)	2.82
		5	0.13 (0.008)	0.30	0.15 (0.000)	0.41	0.12 (0.026)	0.93
		10	0.12 (0.032)	0.31	0.12 (0.001)	0.40	0.13 (0.009)	0.86
	250	1	0.16 (0.009)	0.28	0.19 (0.005)	2.69	0.22 (0.014)	4.97
		5	0.34 (0.013)	0.62	0.36 (0.015)	0.72	0.30 (0.002)	2.93
		10	0.34 (0.002)	0.61	0.35 (0.004)	0.67	0.30 (0.042)	1.75
	500	1	0.26 (0.019)	1.39	0.37 (0.004)	12.19	0.42 (0.002)	11.17
		5	0.56 (0.017)	2.05	0.57 (0.026)	2.09	0.51 (0.018)	7.08
		10	0.59 (0.006)	2.25	0.57 (0.018)	2.12	0.50 (0.034)	3.90
II	50	1	0.51 (0.003)	0.03	0.50 (0.026)	0.22	0.42 (0.006)	10.60
		5	0.37 (0.028)	0.32	0.52 (0.052)	0.41	0.26 (0.023)	1.07
		10	0.55 (0.008)	0.30	0.54 (0.063)	0.42	0.34 (0.015)	1.18
	250	1	1.19 (0.017)	0.28	1.10 (0.024)	2.82	1.37 (0.003)	42.45
		5	0.97 (0.009)	0.61	1.16 (0.038)	0.78	0.71 (0.002)	3.15
		10	1.02 (0.119)	0.63	1.17 (0.000)	0.70	0.68 (0.013)	2.04
	500	1	1.53 (0.014)	1.39	1.46 (0.003)	11.87	1.55 (0.026)	11.45
		5	1.35 (0.040)	2.06	1.52 (0.031)	2.11	1.28 (0.080)	7.50
		10	1.25 (0.063)	2.10	1.49 (0.034)	2.11	1.09 (0.003)	4.25
III	50	1	0.85 (0.001)	0.03	0.74 (0.025)	0.21	0.68 (0.011)	19.34
		5	0.63 (0.032)	0.30	0.83 (0.025)	0.42	0.55 (0.155)	1.28
		10	0.72 (0.058)	0.30	0.72 (0.026)	0.40	0.42 (0.008)	1.62
	250	1	1.52 (0.086)	0.28	1.38 (0.089)	2.66	1.75 (0.089)	281.22
		5	1.28 (0.084)	0.64	1.47 (0.073)	0.73	1.17 (0.017)	3.34
		10	1.20 (0.281)	0.61	1.54 (0.005)	0.71	1.13 (0.122)	2.44
	500	1	1.81 (0.068)	1.39	1.81 (0.064)	11.57	1.92 (0.001)	17.24
		5	1.66 (0.218)	2.06	1.81 (0.059)	2.11	1.56 (0.037)	8.28
		10	1.64 (0.064)	2.07	1.75 (0.012)	2.14	1.35 (0.050)	4.61

Table 5.3: Comparison of the time taken of SIR, PDWD and PSVM for different amount of subsets for $n = 1000$. The table shows the mean performance/distance (standard deviation in parenthesis) and time (in seconds) of 100 iterations.

5.2.3.3 Splitting n and p

Similar to the simulations produced without splitting the sample space, we are once again interested in the difference between the results with and without structured correlation. Therefore, tables 5.4 and 5.5 show the simulation results for $s = 0$ and $s = 0.2$, respectively. Both tables explore the results of all three models for $r = 2$, with multiple choices of m and g.

From both sets of simulations, increasing g has a negative impact on the performance, which is apparent for both choices of r. We once again see that separating the feature space has a positive effect on the performance, however the positive effect does not appear to outweigh the negative impact of separating the sample space.

Model	m	g	SIR	PDWD	PSVM
I	1	1	0.06 (0.007)	0.07 (0.008)	0.06 (0.007)
		2	0.12 (0.063)	0.07 (0.008)	0.07 (0.010)
		5	0.19 (0.111)	0.08 (0.011)	0.08 (0.025)
	2	1	0.10 (0.011)	0.10 (0.012)	0.11 (0.013)
		2	0.11 (0.015)	0.11 (0.015)	0.11 (0.015)
		5	0.11 (0.016)	0.11 (0.015)	0.11 (0.018)
	5	1	0.11 (0.015)	0.12 (0.012)	0.11 (0.018)
		2	0.12 (0.017)	0.13 (0.016)	0.12 (0.017)
		5	0.13 (0.018)	0.13 (0.016)	0.12 (0.019)
II	1	1	0.58 (0.055)	0.54 (0.054)	0.60 (0.103)
		2	0.60 (0.060)	0.96 (0.423)	0.79 (0.267)
		5	0.85 (0.363)	1.09 (0.397)	0.99 (0.358)
	2	1	0.50 (0.061)	0.50 (0.058)	0.51 (0.074)
		2	0.85 (0.389)	0.86 (0.394)	0.90 (0.414)
		5	0.99 (0.384)	1.00 (0.384)	1.03 (0.389)
	5	1	0.50 (0.061)	0.48 (0.062)	0.51 (0.054)
		2	0.88 (0.396)	0.89 (0.420)	0.92 (0.424)
		5	1.01 (0.382)	1.02 (0.396)	1.05 (0.395)
III	1	1	0.83 (0.087)	0.71 (0.060)	0.66 (0.066)
		2	1.00 (0.220)	1.04 (0.346)	0.92 (0.318)
		5	1.24 (0.390)	1.17 (0.341)	1.12 (0.381)
	2	1	0.68 (0.076)	0.68 (0.073)	0.64 (0.050)
		2	0.94 (0.296)	0.97 (0.316)	0.93 (0.311)
		5	1.04 (0.297)	1.07 (0.307)	1.07 (0.331)
	5	1	0.67 (0.069)	0.67 (0.067)	0.68 (0.049)
		2	0.97 (0.322)	0.98 (0.340)	0.96 (0.342)
		5	1.07 (0.314)	1.08 (0.320)	1.07 (0.327)

Table 5.4: Comparison of different values of m and g for PDWD, PSVM and SIR. The table shows the mean performances (standard deviation in parenthesis) of each method with $n = 1000$, $p = 50$ and $s = 0$ of 100 iterations.

Model	m	g	SIR	PDWD	PSVM
I	1	1	0.07 (0.009)	0.08 (0.010)	0.07 (0.008)
		2	0.13 (0.069)	0.08 (0.009)	0.08 (0.011)
		5	0.21 (0.119)	0.08 (0.012)	0.09 (0.021)
	2	1	0.11 (0.013)	0.12 (0.014)	0.12 (0.009)
		2	0.12 (0.016)	0.12 (0.016)	0.12 (0.015)
		5	0.12 (0.016)	0.12 (0.017)	0.13 (0.018)
	5	1	0.13 (0.014)	0.13 (0.016)	0.13 (0.014)
		2	0.14 (0.017)	0.13 (0.017)	0.13 (0.014)
		5	0.14 (0.017)	0.14 (0.018)	0.13 (0.013)
II	1	1	0.60 (0.056)	0.57 (0.057)	0.59 (0.070)
		2	0.62 (0.057)	0.96 (0.405)	0.75 (0.244)
		5	0.86 (0.356)	1.09 (0.382)	0.97 (0.368)
	2	1	0.53 (0.077)	0.51 (0.070)	0.52 (0.061)
		2	0.88 (0.381)	0.89 (0.406)	0.88 (0.403)
		5	1.02 (0.373)	1.03 (0.390)	1.03 (0.393)
	5	1	0.52 (0.066)	0.52 (0.066)	0.53 (0.057)
		2	0.90 (0.398)	0.91 (0.404)	0.92 (0.412)
		5	1.03 (0.380)	1.03 (0.381)	1.04 (0.382)
III	1	1	0.94 (0.089)	0.80 (0.069)	0.72 (0.050)
		2	1.10 (0.203)	1.11 (0.321)	1.06 (0.367)
		5	1.31 (0.348)	1.23 (0.315)	1.21 (0.372)
	2	1	0.74 (0.083)	0.75 (0.076)	0.77 (0.082)
		2	1.04 (0.322)	1.05 (0.319)	1.02 (0.293)
		5	1.15 (0.311)	1.15 (0.310)	1.13 (0.295)
	5	1	0.75 (0.076)	0.76 (0.071)	0.72 (0.060)
		2	1.06 (0.328)	1.05 (0.317)	1.05 (0.349)
		5	1.16 (0.311)	1.15 (0.303)	1.17 (0.333)

Table 5.5: Comparison of different values of m and g for PDWD, PSVM and SIR. The table shows the mean performances (standard deviation in parenthesis) of each method with $n = 1000$, $p = 50$ and $s = 0.2$ of 100 iterations.

5.2.3.4 Order determination

Our proposed method is a two-step method and so we are interested in assessing the efficiency of our order determination choice for each step and the overall method. The first step consists of separating the variables which means we will need to estimate d for each subset. Our method assumes the combined dimension $\tilde{d}$ of $\hat{\boldsymbol{A}}$ found in the first step to be $d \leq \tilde{d} \leq md$, where $\tilde{d}$ is equal to the sum of the $\tilde{d}_i$'s for each subset. Therefore, we only require $d_i \leq \tilde{d}_i \leq d$, for all $i = 1, \ldots, m$.

To estimate the dimension, we will use the ladle estimator and will be considering models I, II and III, where we have $d = 1, 2, 2$, respectively. We are once again interested in the effects on the efficiency of the ladle estimator as a choice for order determination when the important variables fall into different subsets. Therefore,

we will evaluate the efficiency for $r = 2$ and $r = p$.

We are once again interested in the difference between results with and without structured correlation, hence tables 5.6 and 5.7 show the results for $s = 0$ and $s = 0.2$, respectively. The ladle estimator produces better results for models I and II for both $s = 0$ and $s = 0.2$. Generally, as m increases the results suffer, however for models I and II the percentage of correctly estimated values remains above 80 percent for both $r = 2$ and $r = p$. There seems to be some difference between the results with and without structures correlation, which is apparent in model III for SIR and PSVM.

Model	m	step	SIR		PDWD		PSVM	
			$r = 2$	$r = p$	$r = 2$	$r = p$	$r = 2$	$r = p$
1	1	-	100	100	100	100	100	100
	2	first	100	95	99	99	97	97
		second	81	96	100	100	100	100
		both	81	95	99	99	97	97
	5	first	100	99	100	100	100	100
		second	96	97	100	100	100	100
		both	96	96	100	100	100	100
2	1	-	97	98	99	100	98	100
	2	first	100	99	100	100	100	80
		second	91	97	100	100	100	64
		both	91	97	100	100	100	56
	5	first	100	100	100	100	100	38
		second	95	99	100	100	100	72
		both	95	99	100	100	100	25
3	1	-	61	66	100	100	94	93
	2	first	88	98	100	100	96	95
		second	65	93	100	100	96	62
		both	60	91	100	100	96	60
	5	first	97	99	100	100	93	54
		second	75	82	99	100	93	78
		both	75	82	99	100	93	45

Table 5.6: Comparison of different values of m of the ladle estimator for PDWD, PSVM and SIR. The table shows the percentage of correct estimations of each method with $n = 300$, $p = 10$ and $s = 0$ of 100 iterations.

Model	m	step	SIR		PDWD		PSVM	
			$r=2$	$r=p$	$r=2$	$r=p$	$r=2$	$r=p$
1	1	-	100	100	100	100	100	100
	2	first	100	94	99	100	95	100
		second	82	95	100	100	100	100
		both	82	93	99	100	95	100
	5	first	98	98	99	100	99	100
		second	93	98	100	100	100	100
		both	92	96	99	100	99	100
2	1	-	96	95	99	100	100	100
	2	first	100	100	100	100	100	81
		second	84	100	100	100	100	53
		both	84	100	100	100	100	44
	5	first	100	100	100	100	100	31
		second	94	97	100	100	100	80
		both	94	97	100	100	100	27
3	1	-	54	66	99	100	88	95
	2	first	78	99	100	100	84	90
		second	61	92	100	100	84	52
		both	54	91	100	100	84	47
	5	first	94	99	100	99	77	57
		second	67	86	99	100	76	68
		both	65	85	99	99	76	49

Table 5.7: Comparison of different values of m of the ladle estimator for PDWD, PSVM and SIR. The table shows the percentage of correct estimations of each method with $n = 300$, $p = 10$ and $s = 0.2$ of 100 iterations.

5.2.4 Real data analysis

Finally, we turn our attention to a real data analysis. We aim to assess the effects of introducing random data into the real dataset. Our chosen data is the Fertility dataset taken from the UCI Machine Learning Repository. The data contains 100 samples and 9 predictor variables. We will add random variables to give $\tilde{p} = 9 +$ number of added variables. We will then perform our method on the new data and compare the predictors with our original predictor. Again, we are interested in when the true variables fall into the same subset and into different subsets. Therefore, we will produce 2 results for each added data, one where we add the variables at the end and one where we add the variables in the middle.

Table 5.8 shows the performance for multiple numbers of added variables. The left column is the performance when the true variables fall into the same subset and the right column show the performance when the true variables fall into multiple subsets. We can see that ensuring all the true variables fall into the same subset has no significant impact on the results but increasing m consistently improves the

results which implies that a higher value of m makes each method more robust against unrelated data being included.

$\tilde{p}$	m	SIR		PDWD		PSVM	
15	1	0.13 (0.046)		0.12 (0.025)		0.12 (0.050)	
	2	0.11 (0.031)	0.12 (0.041)	0.11 (0.029)	0.11 (0.030)	0.13 (0.045)	0.13 (0.034)
	3	0.12 (0.026)	0.12 (0.030)	0.13 (0.062)	0.10 (0.021)	0.10 (0.028)	0.10 (0.045)
20	1	0.18 (0.051)		0.16 (0.040)		0.18 (0.037)	
	2	0.16 (0.038)	0.15 (0.044)	0.18 (0.057)	0.14 (0.025)	0.16 (0.039)	0.16 (0.028)
	3	0.15 (0.021)	0.14 (0.037)	0.15 (0.059)	0.15 (0.042)	0.16 (0.073)	0.16 (0.028)
30	1	0.23 (0.029)		0.26 (0.063)		0.24 (0.050)	
	2	0.20 (0.040)	0.25 (0.068)	0.23 (0.069)	0.23 (0.040)	0.26 (0.114)	0.28 (0.125)
	3	0.23 (0.040)	0.21 (0.025)	0.22 (0.046)	0.23 (0.048)	0.24 (0.040)	0.25 (0.062)
110	1	-		-		-	
	2	1.03 (0.359)	0.92 (0.252)	0.95 (0.348)	0.98 (0.275)	0.97 (0.348)	1.14 (0.260)
	3	1.13 (0.380)	1.03 (0.291)	1.07 (0.357)	1.09 (0.390)	1.07 (0.276)	1.05 (0.268)

Table 5.8: Comparison of different amounts of random data for PDWD, PSVM and SIR. The table shows the mean performances (standard deviation in parenthesis) of each method of 100 iterations. Left column: all true variables fall into same subset, right column: true variables fall into different subsets.

5.3 SDR without decorrelating variables

Previously we attempted to perform dimension reduction through feature space partitioning by decorrelating the variables before separation. We propose a new approach that does not require the decorrelation step. By excluding this step, we hope to develop a "divide and conquer" method that can be performed on high dimension low sample space (HDLSS) data and more importantly when the dimension size exceeds the sample size, which is a strict restriction that many previous methods suffer from. From this point forward we will refer to the method defined in the previous section as method 1, and the following method as method 2.

Method 1 begins by taking advantage of the common standardisation step found in many classical methods and using this step to decorrelate the features. By including this step, it was clear that the consistency of the overall method required p to remain less that n. We also believe that avoiding this step will have a positive impact on the computational time of the method.

5.3.1 Estimation algorithm

Similar to method 1 we will require a two-step method since the first step will once again estimate a CDRS, but not necessarily the minimal CDRS. The estimation procedure is as follows:

1. Split the feature space of $\boldsymbol{X}$ into m subsets such that $\boldsymbol{X} = \boldsymbol{X}^{(1)}, \ldots, \boldsymbol{X}^{(m)}$, where the dimension of $\boldsymbol{X}^{(i)}$ is p_i and $p = p_1 + \cdots + p_m$.

2. Perform the desired SDR method on each subset to obtain $\hat{\boldsymbol{A}}^{(i)}$ of size $p_i \times d^{(1)}$, where $d \leq d^{(1)} \leq md$ and $d^{(1)} = d_1 + \cdots + d_m$. Now form a matrix $\hat{\boldsymbol{A}}$ as follows

$$\hat{\boldsymbol{A}} = \begin{pmatrix} \hat{\boldsymbol{A}}^{(1)} & \boldsymbol{0}_{p_1 \times d_1} & \cdots & \boldsymbol{0}_{p_1 \times d_1} \\ \boldsymbol{0}_{p_2 \times d_2} & \hat{\boldsymbol{A}}^{(2)} & \cdots & \boldsymbol{0}_{p_2 \times d_2} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0}_{p_m \times d_m} & \boldsymbol{0}_{p_m \times d_m} & \cdots & \hat{\boldsymbol{A}}^{(m)} \end{pmatrix}.$$

3. Perform the chosen SDR method on $\hat{\boldsymbol{A}}^{\mathsf{T}}\boldsymbol{X}$ to obtain $\hat{\boldsymbol{B}}$ of size $md \times d$.

4. Let $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{A}}\hat{\boldsymbol{B}}$ and so we can use the subspace spanned by $\hat{\boldsymbol{\beta}} = (\hat{\boldsymbol{\beta}}_1, \ldots, \hat{\boldsymbol{\beta}}_d)$ to estimate the CDRS.

5.3.2 Separating the sample space and feature space

As previously discussed, there have been multiple methods of separating the sample space and performing parallel programming versions of SDR. One in particular is BIG-SIR, introduced by Liquet and Saracco (2016). Similar to the extension we obtained when decorrelating the variables, we can now extend our method so that we can separate both the sample space and the feature space. The extended method is as follows:

1. Split the feature space of $\boldsymbol{X}$ into m subsets such that $\boldsymbol{X} = (\boldsymbol{X}^{(1)\mathsf{T}}, \ldots, \boldsymbol{X}^{(m)\mathsf{T}})^{\mathsf{T}}$, where the dimension of $\boldsymbol{X}^{(i)}$ is p_i and $p = p_1 + \cdots + p_m$.

2. Split the sample space into g slices, to give $n = n_1 + \cdots + n_g$, $\boldsymbol{Y} = (\boldsymbol{Y}_{(1)}, \ldots, \boldsymbol{Y}_{(g)})$ and

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_{(1)}^{(1)} & \cdots & \boldsymbol{X}_{(g)}^{(1)} \\ \vdots & \ddots & \vdots \\ \boldsymbol{X}_{(1)}^{(m)} & \cdots & \boldsymbol{X}_{(g)}^{(m)} \end{pmatrix},$$

   such that $p < n_i$ for all $i = 1, \ldots, g$.

3. Perform the desired method on each subset to produce the candidate matrix $\hat{\boldsymbol{M}}_{(i)}^{(j)}$ for all $i = 1, \ldots, g$ and $j = 1, \ldots, m$.

4. Recollect the data and calculate

$$\hat{\boldsymbol{M}}^{(j)} = \sum_{i=1}^{g} \frac{1}{n_i} \hat{\boldsymbol{M}}_{(i)}^{(j)} \hat{\boldsymbol{M}}_{(i)}^{(j)\mathsf{T}}$$

   for all $j = 1, \ldots, m$.

5. Finally, we let $\boldsymbol{\eta}^{(j)}$ be the d largest eigenvectors of $\hat{\boldsymbol{M}}^{(j)}$. Thus $\hat{\boldsymbol{A}}^{(j)} = \hat{\boldsymbol{\Sigma}}_{xx}^{-1/2}\boldsymbol{\eta}^{(j)}$ for all $j = 1, \ldots, m$.

6. Using $\hat{\boldsymbol{A}}^{(j)}$ of size $p_j \times d^{(1)}$, where $d \leq d^{(1)} \leq md$ and $d^{(1)} = d_1 + \cdots + d_m$, form a matrix $\hat{\boldsymbol{A}}$ as follows

$$\hat{\boldsymbol{A}} = \begin{pmatrix} \hat{\boldsymbol{A}}^{(1)} & \mathbf{0}_{p_1 \times d_1} & \cdots & \mathbf{0}_{p_1 \times d_1} \\ \mathbf{0}_{p_2 \times d_2} & \hat{\boldsymbol{A}}^{(2)} & \cdots & \mathbf{0}_{p_2 \times d_2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{p_m \times d_m} & \mathbf{0}_{p_m \times d_m} & \cdots & \hat{\boldsymbol{A}}^{(m)} \end{pmatrix}.$$

7. Again separate $\hat{\boldsymbol{A}}^{\mathsf{T}}\boldsymbol{X}$ into g subsets as above and perform the desired method on each subset to obtain $\hat{\boldsymbol{N}}_{(i)}$ for all $i = 1, \ldots, g$. Calculate

$$\hat{\boldsymbol{N}} = \sum_{i=1}^{g} \frac{1}{n_i} \hat{\boldsymbol{N}}_{(i)} \boldsymbol{N}_{(i)}^{\mathsf{T}}.$$

8. Now $\hat{\boldsymbol{B}} = \hat{\boldsymbol{\Sigma}}_{A^{\mathsf{T}}x}\boldsymbol{\nu}$, where $\boldsymbol{\nu}$ is the d largest eigenvectors of $\hat{\boldsymbol{N}}$ and $\hat{\boldsymbol{B}}$ is of size $md \times d$.

9. Let $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{A}}\hat{\boldsymbol{B}}$ and so the subspace spanned by $\hat{\boldsymbol{\beta}} = (\hat{\boldsymbol{\beta}}_1, \ldots, \hat{\boldsymbol{\beta}}_d)$ can be used to estimate the CDRS.

5.3.3 Synthetic simulation studies

We will begin with synthetic simulation studies to evaluate the efficiency of method 2 with respect to both accuracy and time using the same measure and models as for method 1. Again, since method 2 also involves the separation of the feature space it is important that we consider models with structured correlation. Therefore, we choose $X \sim N(\mathbf{0}, \boldsymbol{\Sigma})$ and $\epsilon \sim N(\mathbf{0}, I_p)$, with $\Sigma_{ij} = s^{|i-j|}$. Analogous to the previous section, h is chosen to be 20 and λ is chosen to be 0.1.

5.3.3.1 Performance when combinations of variables fall in different subsets

As stated with the models, we need to evaluate our method when all the important variables fall into one subset and when they fall into different subsets. We have run synthetic simulations for the models with $n = 100$ and for different values for p and m.

Immediate inspection of the results shows that separating the variables has a positive impact on the performance, with a greater impact occurring as p increases. This is evident for both choices of $s = 0$ and 0.2, represented in table 5.9 and

table 5.10 respectively. There is a slight decrease in accuracy when the important variables fall into different subsets however it is clear that as m increases the results still improve analogous to when the important variables fall into the same subset. There is no obvious change in performance when structured covariance is added (when $s > 0$) as it seems to follow the same pattern as already described.

An important feature of interest is the case for models II and III, which have effective dimension 2 and thus $md = 10$ for $m = 5$. Therefore when $p = 10$, we would expect the results for $m = 1$ and $m = 5$ to be similar which is clearly indicated in both table 5.9 and table 5.10.

Figure 5.3 shows the comparison of separating the feature space against not separating the feature space for 720 scenarios, including all models, multiple values of p, m, r and s, and for the three classical methods that we have previously considered. Figure 5.3 clearly shows that separating the variables produces accurate results more often, in fact separating the features produced better results 85.7% of the time and the mean performance for not separating and separating is 1.29 and 1.05, respectively.

Figure 5.4 shows the results shown in figure 5.3, broken down by model, p, m, r, s and classical method. We see similar results for each choice of classic method and for each choice of s, which shows that introducing structured covariance has little impact on the accuracy of method 2. It is also clear that the choice of model effects the accuracy of our method in comparison to no separation, where for model I we see many more occurrences of no separation performing better.

Model	p	m	SIR		PDWD		PSVM	
			$r=2$	$r=p$	$r=2$	$r=p$	$r=2$	$r=p$
I	10	1	0.10 (0.022)	0.06 (0.013)	0.09 (0.033)	0.12 (0.042)	0.12 (0.016)	0.09 (0.008)
		2	0.11 (0.029)	0.44 (0.241)	0.07 (0.048)	0.34 (0.028)	0.07 (0.007)	0.28 (0.038)
		5	0.05 (0.009)	0.13 (0.073)	0.08 (0.016)	0.15 (0.034)	0.08 (0.010)	0.17 (0.117)
	50	1	0.43 (0.022)	0.29 (0.043)	0.35 (0.056)	0.44 (0.081)	0.44 (0.066)	0.39 (0.036)
		2	0.19 (0.059)	0.97 (0.125)	0.23 (0.006)	0.68 (0.023)	0.23 (0.025)	0.88 (0.158)
		5	0.17 (0.005)	0.55 (0.037)	0.12 (0.028)	0.55 (0.102)	0.12 (0.054)	0.56 (0.001)
	100	1	1.41 (0.012)	1.41 (0.002)	1.37 (0.027)	1.31 (0.020)	1.34 (0.084)	1.40 (0.014)
		5	0.16 (0.017)	0.75 (0.189)	0.25 (0.068)	0.81 (0.054)	0.26 (0.007)	0.88 (0.031)
		10	0.18 (0.044)	0.56 (0.115)	0.17 (0.047)	0.55 (0.091)	0.15 (0.011)	0.49 (0.004)
	200	1	-	-	-	-	-	-
		5	0.42 (0.081)	1.32 (0.082)	0.52 (0.081)	1.15 (0.084)	0.56 (0.090)	1.23 (0.066)
		10	0.30 (0.062)	1.05 (0.147)	0.39 (0.076)	0.93 (0.107)	0.33 (0.069)	0.97 (0.126)
II	10	1	0.66 (0.063)	0.87 (0.061)	0.62 (0.002)	0.77 (0.230)	0.42 (0.181)	0.59 (0.027)
		2	0.79 (0.446)	0.56 (0.090)	0.74 (0.108)	0.64 (0.172)	0.35 (0.098)	0.73 (0.262)
		5	0.66 (0.063)	0.87 (0.061)	0.62 (0.002)	0.77 (0.230)	0.42 (0.181)	0.59 (0.027)
	50	1	1.67 (0.026)	1.62 (0.007)	1.33 (0.004)	1.44 (0.127)	1.51 (0.199)	1.50 (0.138)
		2	1.44 (0.030)	1.52 (0.003)	1.17 (0.090)	1.32 (0.112)	1.45 (0.032)	1.38 (0.127)
		5	1.43 (0.046)	1.35 (0.159)	1.10 (0.102)	1.13 (0.264)	0.86 (0.316)	1.33 (0.147)
	100	1	1.98 (0.019)	1.96 (0.011)	1.97 (0.011)	1.92 (0.046)	1.95 (0.060)	1.93 (0.040)
		5	1.61 (0.056)	1.49 (0.091)	1.42 (0.039)	1.34 (0.091)	1.34 (0.096)	1.42 (0.019)
		10	1.36 (0.220)	1.53 (0.080)	1.48 (0.064)	1.35 (0.047)	1.36 (0.068)	1.48 (0.017)
	200	1	-	-	-	-	-	-
		5	1.82 (0.072)	1.82 (0.073)	1.69 (0.067)	1.72 (0.065)	1.71 (0.090)	1.73 (0.077)
		10	1.75 (0.083)	1.75 (0.069)	1.63 (0.072)	1.65 (0.077)	1.51 (0.114)	1.63 (0.089)
III	10	1	1.18 (0.497)	1.44 (0.304)	0.78 (0.100)	1.18 (0.042)	0.84 (0.005)	1.05 (0.171)
		2	1.41 (0.126)	1.51 (0.012)	0.49 (0.165)	0.80 (0.102)	0.98 (0.400)	1.06 (0.504)
		5	1.18 (0.497)	1.44 (0.304)	0.78 (0.100)	1.18 (0.042)	0.84 (0.005)	1.05 (0.171)
	50	1	1.84 (0.170)	1.88 (0.093)	1.78 (0.093)	1.82 (0.102)	1.91 (0.014)	1.87 (0.009)
		2	1.82 (0.070)	1.85 (0.069)	1.69 (0.227)	1.53 (0.065)	1.61 (0.217)	1.74 (0.245)
		5	1.59 (0.220)	1.63 (0.205)	1.42 (0.216)	1.52 (0.256)	1.56 (0.082)	1.76 (0.043)
	100	1	1.99 (0.002)	1.98 (0.004)	1.98 (0.017)	1.97 (0.022)	1.98 (0.014)	1.98 (0.009)
		5	1.82 (0.140)	1.96 (0.036)	1.71 (0.116)	1.74 (0.287)	1.91 (0.065)	1.75 (0.019)
		10	1.84 (0.140)	1.82 (0.207)	1.81 (0.054)	1.87 (0.099)	1.78 (0.102)	1.81 (0.081)
	200	1	-	-	-	-	-	-
		5	1.98 (0.018)	1.98 (0.026)	1.95 (0.034)	1.94 (0.045)	1.96 (0.027)	1.96 (0.028)
		10	1.97 (0.030)	1.97 (0.040)	1.93 (0.045)	1.92 (0.057)	1.94 (0.049)	1.93 (0.049)

Table 5.9: Comparison of different amounts of subsets for SIR, PDWD and PSVM. The table shows the mean performances (standard deviation in parenthesis) of each method with $n = 100$ and $s = 0$ of 100 iterations.

Model	p	m	SIR		PDWD		PSVM	
			$r=2$	$r=p$	$r=2$	$r=p$	$r=2$	$r=p$
I	10	1	0.14 (0.009)	0.10 (0.006)	0.15 (0.019)	0.09 (0.014)	0.11 (0.041)	0.10 (0.014)
		2	0.06 (0.021)	0.40 (0.023)	0.07 (0.006)	0.35 (0.003)	0.07 (0.049)	0.32 (0.043)
		5	0.09 (0.019)	0.26 (0.002)	0.10 (0.048)	0.13 (0.047)	0.06 (0.009)	0.22 (0.200)
	50	1	0.32 (0.035)	0.29 (0.018)	0.52 (0.155)	0.36 (0.103)	0.64 (0.040)	0.52 (0.025)
		2	0.29 (0.045)	0.86 (0.041)	0.26 (0.011)	0.72 (0.162)	0.31 (0.045)	0.74 (0.202)
		5	0.11 (0.009)	0.50 (0.216)	0.21 (0.075)	0.63 (0.020)	0.14 (0.026)	0.49 (0.050)
	100	1	1.40 (0.021)	1.40 (0.017)	1.36 (0.037)	1.38 (0.030)	1.39 (0.030)	1.28 (0.179)
		5	0.18 (0.024)	1.15 (0.326)	0.20 (0.063)	0.70 (0.099)	0.22 (0.028)	0.86 (0.111)
		10	0.14 (0.026)	0.48 (0.160)	0.20 (0.038)	0.36 (0.011)	0.11 (0.025)	0.65 (0.109)
	200	1	-	-	-	-	-	-
		5	0.29 (0.046)	1.35 (0.055)	0.36 (0.041)	1.05 (0.081)	0.42 (0.098)	1.17 (0.062)
		10	0.22 (0.033)	0.97 (0.123)	0.27 (0.041)	0.78 (0.094)	0.25 (0.032)	0.91 (0.090)
II	10	1	1.00 (0.327)	0.94 (0.197)	0.72 (0.115)	0.57 (0.085)	0.70 (0.157)	0.70 (0.040)
		2	1.00 (0.122)	1.20 (0.021)	0.75 (0.145)	0.65 (0.096)	0.34 (0.015)	0.88 (0.037)
		5	1.00 (0.327)	0.94 (0.197)	0.72 (0.115)	0.57 (0.085)	0.70 (0.157)	0.70 (0.040)
	50	1	1.58 (0.021)	1.74 (0.000)	1.37 (0.061)	1.50 (0.002)	1.54 (0.092)	1.46 (0.068)
		2	1.55 (0.117)	1.54 (0.120)	1.18 (0.092)	1.21 (0.119)	1.29 (0.244)	1.41 (0.011)
		5	1.26 (0.343)	1.55 (0.022)	1.06 (0.021)	1.30 (0.146)	0.76 (0.075)	1.21 (0.015)
	100	1	1.99 (0.007)	1.98 (0.012)	1.88 (0.072)	1.95 (0.047)	1.93 (0.034)	1.98 (0.010)
		5	1.60 (0.006)	1.44 (0.007)	1.41 (0.228)	1.42 (0.039)	1.18 (0.066)	1.35 (0.008)
		10	1.49 (0.040)	1.61 (0.118)	1.15 (0.133)	1.48 (0.163)	1.32 (0.156)	1.49 (0.127)
	200	1	-	-	-	-	-	-
		5	1.75 (0.088)	1.76 (0.057)	1.62 (0.055)	1.63 (0.063)	1.59 (0.080)	1.67 (0.138)
		10	1.67 (0.079)	1.69 (0.073)	1.55 (0.076)	1.57 (0.083)	1.40 (0.158)	1.56 (0.079)
III	10	1	1.69 (0.053)	1.35 (0.325)	1.15 (0.153)	1.18 (0.063)	0.91 (0.608)	0.81 (0.081)
		2	1.62 (0.012)	0.93 (0.058)	1.35 (0.017)	1.04 (0.113)	0.58 (0.368)	0.88 (0.248)
		5	1.69 (0.053)	1.35 (0.325)	1.15 (0.153)	1.18 (0.063)	0.91 (0.608)	0.81 (0.081)
	50	1	1.92 (0.014)	1.93 (0.014)	1.77 (0.139)	1.68 (0.085)	1.84 (0.100)	1.86 (0.029)
		2	1.97 (0.012)	1.81 (0.139)	1.74 (0.054)	1.80 (0.152)	1.85 (0.037)	1.74 (0.078)
		5	1.79 (0.170)	1.86 (0.058)	1.56 (0.367)	1.45 (0.017)	1.41 (0.206)	1.56 (0.157)
	100	1	1.98 (0.002)	1.98 (0.012)	1.99 (0.013)	1.96 (0.032)	1.98 (0.005)	1.95 (0.009)
		5	1.87 (0.112)	1.93 (0.053)	1.88 (0.065)	1.81 (0.068)	1.83 (0.022)	1.71 (0.103)
		10	1.87 (0.086)	1.88 (0.035)	1.83 (0.103)	1.89 (0.034)	1.79 (0.032)	1.78 (0.106)
	200	1	-	-	-	-	-	-
		5	1.97 (0.022)	1.95 (0.041)	1.91 (0.057)	1.89 (0.054)	1.94 (0.032)	1.93 (0.040)
		10	1.96 (0.037)	1.92 (0.058)	1.89 (0.069)	1.85 (0.066)	1.88 (0.089)	1.88 (0.063)

Table 5.10: Comparison of different amounts of subsets for PDWD, PSVM and SIR. The table shows the mean performances (standard deviation in parenthesis) of each method with $n = 100$ and $s = 0.2$ of 100 iterations.

It is clear that the choice of r has an impact on the accuracy of method 2, where we once again see that most of the occurrences of no separation producing better results appearing for $r = p$. Finally, similar to method 1, we see that increasing p and m increases the performance of method 2 compared with $m = 1$.

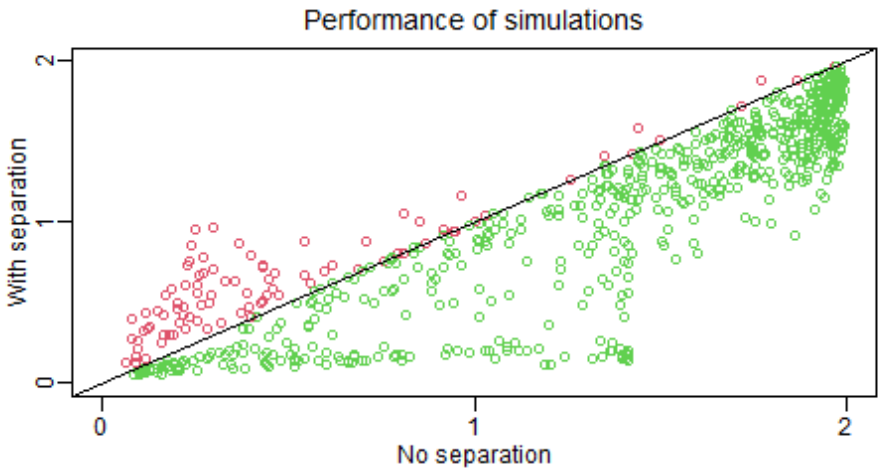

Figure 5.3: Performance of our method of 720 scenarios for all models and multiple choices of p, m, r, s, including results for SIR, PDWD and PSVM. Green points indicate results where the performance of our method was better than not separating and red show the alternative.

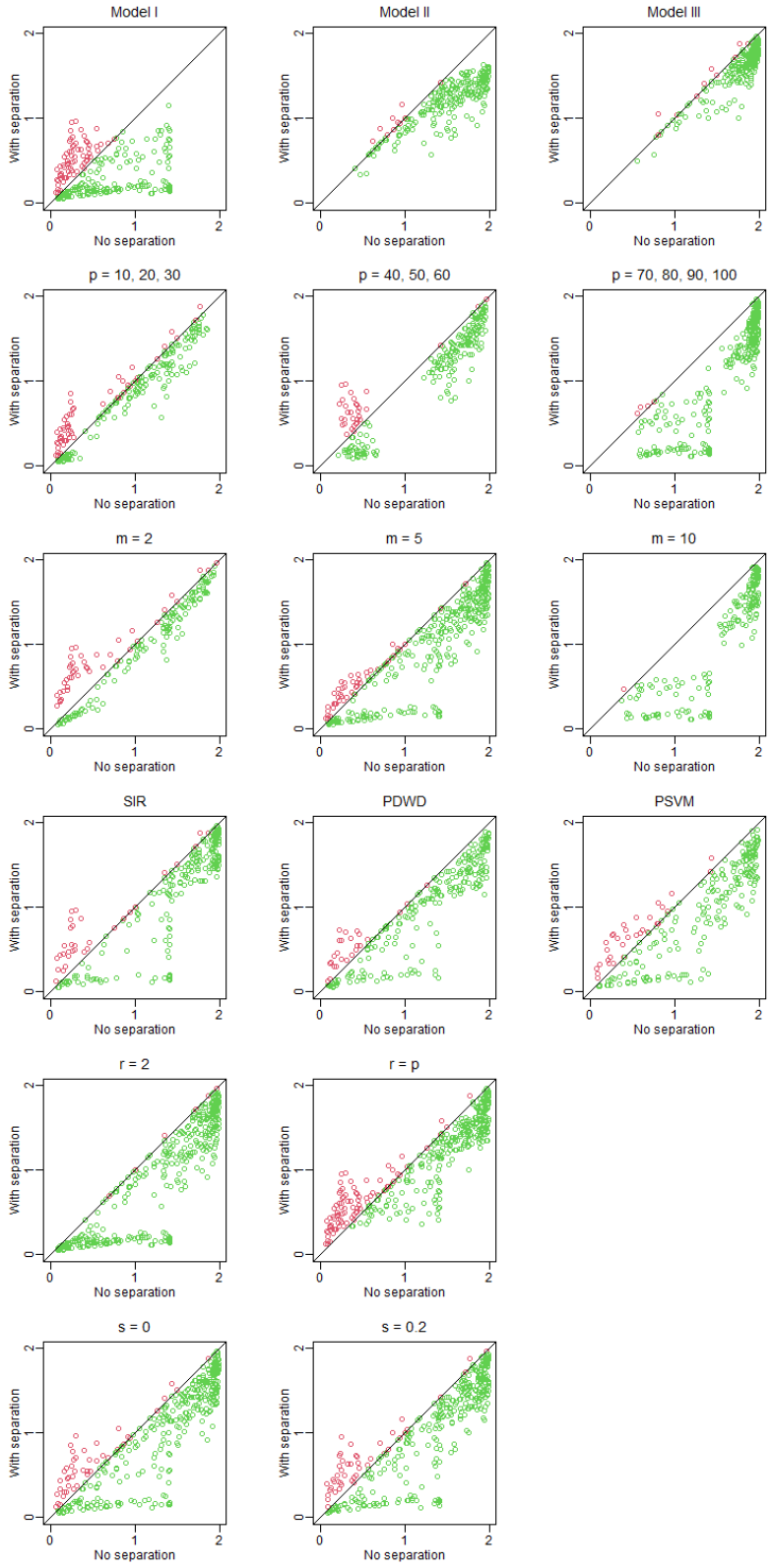

Figure 5.4: Performance of our method of 720 scenarios by model, p, m, r, s and choice of classic method. Green points indicate results where the performance of our method was better than not separating and red show the alternative.

5.3.3.2 Performance and time

We are also interested in the computational time advantages of our proposed method. Since we are investigating the time of the method rather than the performance, we will only consider $r = 1$ and $s = 0$. Here we have chosen $n = 1000$ with multiple values for p and m. The times for $m > 1$ indicate an overall time for the method when the data is analysed on separate machines. If all analysis is performed on one machine in succession, then the time would likely increase as m increases. Table 5.11 shows the performance and time in each case.

Model	p	m	SIR		PDWD		PSVM	
			Distance	Time	Distance	Time	Distance	Time
I	50	1	0.06 (0.014)	0.05	0.08 (0.001)	0.22	0.07 (0.002)	2.82
		5	0.03 (0.001)	0.05	0.09 (0.002)	0.13	0.03 (0.006)	0.67
		10	0.02 (0.004)	0.04	0.08 (0.005)	0.13	0.02 (0.003)	0.57
	250	1	0.16 (0.009)	0.28	0.19 (0.005)	2.69	0.22 (0.014)	4.97
		5	0.06 (0.000)	0.06	0.18 (0.000)	0.27	0.07 (0.005)	2.62
		10	0.06 (0.006)	0.05	0.16 (0.000)	0.18	0.04 (0.006)	1.28
	500	1	0.26 (0.019)	1.39	0.37 (0.004)	12.19	0.42 (0.002)	11.17
		5	0.09 (0.015)	0.09	0.25 (0.015)	0.56	0.10 (0.001)	7.35
		10	0.07 (0.000)	0.05	0.21 (0.021)	0.27	0.07 (0.004)	2.75
II	50	1	0.51 (0.003)	0.03	0.50 (0.026)	0.22	0.42 (0.006)	10.60
		5	0.33 (0.020)	0.05	0.48 (0.028)	0.13	0.19 (0.019)	0.82
		10	0.50 (0.029)	0.03	0.50 (0.059)	0.16	0.28 (0.079)	0.93
	250	1	1.19 (0.017)	0.28	1.10 (0.024)	2.82	1.37 (0.003)	42.45
		5	0.80 (0.037)	0.05	1.00 (0.026)	0.27	0.49 (0.010)	2.86
		10	0.90 (0.134)	0.06	1.00 (0.004)	0.20	0.42 (0.014)	1.61
	500	1	1.53 (0.014)	1.39	1.46 (0.003)	11.87	1.55 (0.026)	11.45
		5	1.04 (0.149)	0.10	1.24 (0.028)	0.56	0.91 (0.020)	7.32
		10	0.90 (0.083)	0.06	1.20 (0.027)	0.29	0.56 (0.078)	3.04
III	50	1	0.85 (0.001)	0.03	0.74 (0.025)	0.21	0.68 (0.011)	19.34
		5	0.59 (0.001)	0.04	0.78 (0.029)	0.14	0.51 (0.165)	1.17
		10	0.68 (0.045)	0.03	0.68 (0.005)	0.17	0.42 (0.034)	1.41
	250	1	1.52 (0.086)	0.28	1.38 (0.089)	2.66	1.75 (0.089)	281.22
		5	1.07 (0.112)	0.05	1.33 (0.074)	0.26	0.92 (0.011)	3.78
		10	1.12 (0.233)	0.06	1.38 (0.019)	0.20	0.94 (0.127)	2.27
	500	1	1.81 (0.068)	1.39	1.81 (0.064)	11.57	1.92 (0.001)	17.24
		5	1.38 (0.103)	0.08	1.66 (0.093)	0.58	1.23 (0.140)	10.59
		10	1.45 (0.092)	0.07	1.55 (0.006)	0.29	0.88 (0.059)	3.90

Table 5.11: Comparison of the time taken of SIR, PDWD and PSVM for different amount of subsets for $n = 1000$ and $r = 2$. The table shows the mean performance/distance (standard deviation in parenthesis) and time (in seconds) of 100 iterations.

To be expected for lower choices of p there is only a small difference in computational time, with $m = 1$ outperforming $m > 1$ for SIR. However as p increases, especially notably for $p = 500$, there is a vast improvement in the time taken for all methods, but PDWD and PSVM in particular. We have included the performance

of each case to give a clear picture of the multiple benefits of separating the data. Table 5.11 clearly shows that separating the data leads to a steep increase in both the accuracy and the computational efficiency for all three methods and models, especially for larger p.

5.3.3.3 Splitting n and p

We have run simulation studies for models I, II and III with $n = 1000$, $p = 50$ and $r = 1$. Tables 5.12 and 5.13 show the performance for $s = 0$ and $s = 0.2$, respectively, of different methods for different values of m and g. We can see as before separating the feature space has a positive impact on the performance, however separating the sample space has a negative impact on the performance. Similar to the previous chapter, we see that as g increases we see a decrease in the accuracy and in increase in the accuracy as m increases. Unfortunately, once again we see that in decrease in the performance inflicted by increasing g outweighs the increase gained by increasing m, however the results are closer than for method 1. We have seen previously that the size of n seems to positively correlate with the performance of dimension reduction estimators. Therefore, since we are decreasing the size of n we are losing accuracy. Alternatively, we have seen that the size of p negatively correlated with the accuracy of many methods. Therefore, separating the feature space and performing the dimension reduction method on the subset, with a smaller dimension, produces an increased accuracy.

Model	m	g	SIR	PDWD	PSVM
I	1	1	0.06 (0.007)	0.07 (0.008)	0.06 (0.007)
		2	0.12 (0.063)	0.07 (0.008)	0.07 (0.010)
		5	0.19 (0.111)	0.08 (0.011)	0.08 (0.025)
	2	1	0.05 (0.006)	0.05 (0.008)	0.05 (0.006)
		2	0.06 (0.020)	0.05 (0.007)	0.05 (0.008)
		5	0.07 (0.023)	0.05 (0.008)	0.05 (0.008)
	5	1	0.04 (0.007)	0.04 (0.008)	0.04 (0.007)
		2	0.06 (0.030)	0.04 (0.008)	0.04 (0.006)
		5	0.07 (0.036)	0.04 (0.007)	0.04 (0.007)
II	1	1	0.58 (0.055)	0.54 (0.054)	0.60 (0.103)
		2	0.60 (0.060)	0.96 (0.423)	0.79 (0.267)
		5	0.85 (0.363)	1.09 (0.397)	0.99 (0.358)
	2	1	0.48 (0.063)	0.47 (0.059)	0.42 (0.061)
		2	0.72 (0.351)	0.84 (0.402)	0.81 (0.431)
		5	0.87 (0.387)	0.98 (0.393)	0.96 (0.417)
	5	1	0.46 (0.069)	0.42 (0.067)	0.37 (0.037)
		2	0.46 (0.117)	0.85 (0.443)	0.61 (0.328)
		5	0.59 (0.254)	0.98 (0.416)	0.71 (0.342)
III	1	1	0.83 (0.087)	0.71 (0.060)	0.66 (0.066)
		2	1.00 (0.220)	1.04 (0.346)	0.92 (0.318)
		5	1.24 (0.390)	1.17 (0.341)	1.12 (0.381)
	2	1	0.68 (0.081)	0.65 (0.073)	0.56 (0.065)
		2	0.81 (0.241)	0.94 (0.325)	0.79 (0.301)
		5	0.96 (0.313)	1.05 (0.320)	0.95 (0.345)
	5	1	0.68 (0.094)	0.61 (0.072)	0.52 (0.049)
		2	0.71 (0.150)	0.93 (0.353)	0.60 (0.146)
		5	0.91 (0.331)	1.04 (0.334)	0.70 (0.210)

Table 5.12: Comparison of different values of m and g for PDWD, PSVM and SIR. The table shows the mean performances (standard deviation in parenthesis) of each method with $n = 1000$, $p = 50$ and $s = 0$ of 100 iterations.

Model	m	g	SIR	PDWD	PSVM
I	1	1	0.07 (0.009)	0.08 (0.010)	0.07 (0.008)
		2	0.13 (0.069)	0.08 (0.009)	0.08 (0.011)
		5	0.21 (0.119)	0.08 (0.012)	0.09 (0.021)
	2	1	0.05 (0.008)	0.06 (0.007)	0.05 (0.007)
		2	0.06 (0.020)	0.06 (0.008)	0.05 (0.008)
		5	0.07 (0.024)	0.06 (0.009)	0.05 (0.008)
	5	1	0.04 (0.008)	0.05 (0.008)	0.04 (0.010)
		2	0.07 (0.034)	0.04 (0.008)	0.04 (0.009)
		5	0.08 (0.040)	0.04 (0.009)	0.04 (0.009)
II	1	1	0.60 (0.056)	0.57 (0.057)	0.59 (0.070)
		2	0.62 (0.057)	0.96 (0.405)	0.75 (0.244)
		5	0.86 (0.356)	1.09 (0.382)	0.97 (0.368)
	2	1	0.50 (0.076)	0.48 (0.067)	0.46 (0.058)
		2	0.75 (0.342)	0.87 (0.415)	0.82 (0.414)
		5	0.90 (0.374)	1.01 (0.400)	0.98 (0.416)
	5	1	0.48 (0.072)	0.45 (0.068)	0.36 (0.048)
		2	0.50 (0.145)	0.86 (0.426)	0.50 (0.279)
		5	0.62 (0.251)	0.99 (0.403)	0.65 (0.342)
III	1	1	0.94 (0.089)	0.80 (0.069)	0.72 (0.050)
		2	1.10 (0.203)	1.11 (0.321)	1.06 (0.367)
		5	1.31 (0.348)	1.23 (0.315)	1.21 (0.372)
	2	1	0.75 (0.089)	0.71 (0.078)	0.69 (0.104)
		2	0.93 (0.271)	1.02 (0.328)	0.88 (0.284)
		5	1.08 (0.318)	1.13 (0.321)	1.00 (0.308)
	5	1	0.78 (0.097)	0.70 (0.071)	0.62 (0.081)
		2	0.82 (0.169)	1.01 (0.331)	0.76 (0.224)
		5	1.01 (0.325)	1.12 (0.319)	0.84 (0.286)

Table 5.13: Comparison of different values of m and g for PDWD, PSVM and SIR. The table shows the mean performances (standard deviation in parenthesis) of each method with $n = 1000$, $p = 50$ and $s = 0.2$ of 100 iterations.

5.3.3.4 Order determination

As discussed preciously, since our method is a two-step method, we need to evaluate the efficiency of the ladle estimator for each step and the overall performance. For the first step we only require the estimator $\hat{d}$ to satisfy, $d_i \leq \hat{d} \leq d$. To evaluate the estimator we will consider models I, II and III for $r = 2$ and $r = p$, where we have $d = 1$, 2 and 2, respectively.

Tables 5.14 and 5.15 show the percentages of correctly estimated dimensions with $n = 300$ and $p = 10$, for $s = 0$ and $s = 0.2$ respectively. We have evaluated the results for multiple values of m and highlighted the percentage of correct estimations for the first step, the second step and for both steps. It is clear from the results that the ladle estimator is more successful at estimating the correct dimension for PDWD and PSVM compared with SIR. It is also noticeable that the results are affected by the choice of m where the performance generally decreases as m increases.

It is clear that choosing $r = p$ has a much greater impact on the accuracy of the ladle estimator for method 2, than it did for method 1. This is likely due to the correlation between the variables for method 2, which does not exist for method 1.

Model	m	step	SIR		PDWD		PSVM	
			$r=2$	$r=p$	$r=2$	$r=p$	$r=2$	$r=p$
1	1	-	99	100	100	100	100	100
	2	first	97	98	98	99	100	100
		second	99	96	100	100	100	100
		both	96	94	98	99	100	100
	5	first	100	96	100	100	100	100
		second	99	97	100	100	100	100
		both	99	95	100	100	100	100
2	1	-	92	93	100	99	100	99
	2	first	100	100	100	100	100	87
		second	97	36	100	100	100	56
		both	97	36	100	100	100	51
	5	first	100	100	100	100	100	44
		second	99	24	100	100	99	75
		both	99	24	100	100	99	36
3	1	-	58	68	100	100	97	98
	2	first	82	96	99	99	96	88
		second	74	32	100	100	96	58
		both	73	32	99	99	96	53
	5	first	94	98	100	100	87	49
		second	80	14	99	99	85	65
		both	79	14	99	99	85	40

Table 5.14: Comparison of different values of m of the ladle estimator for PDWD, PSVM and SIR. The table shows the percentage of correct estimations of each method with $n = 300$, $p = 10$ and $s = 0$ of 100 iterations.

Model	m	step	SIR		PDWD		PSVM	
			$r=2$	$r=p$	$r=2$	$r=p$	$r=2$	$r=p$
1	1	-	100	100	100	100	100	100
	2	first	99	97	98	100	96	100
		second	96	96	100	100	100	100
		both	96	95	98	100	96	100
	5	first	99	99	100	100	100	99
		second	98	95	100	100	100	100
		both	97	94	100	100	100	99
2	1	-	95	95	100	100	100	100
	2	first	98	100	100	100	100	86
		second	98	50	100	100	100	53
		both	96	50	100	100	100	46
	5	first	100	100	100	100	100	45
		second	99	31	100	100	100	74
		both	99	31	100	100	100	35
3	1	-	51	57	100	100	90	92
	2	first	78	97	100	100	91	87
		second	72	46	99	100	89	52
		both	69	46	99	100	89	47
	5	first	93	98	100	99	76	59
		second	76	24	100	100	75	71
		both	76	24	100	99	75	45

Table 5.15: Comparison of different values of m of the ladle estimator for PDWD, PSVM and SIR. The table shows the percentage of correct estimations of each method with $n = 300$, $p = 10$ and $s = 0.2$ of 100 iterations.

5.3.4 Real data analysis

Lastly, we will now further our investigate through a real data analysis. We will be using the same methodology and dataset that was used for the real data analysis of method 1. Table 5.16 shows the performance for multiple numbers of added variables. The left column is the performance when the true variables fall into the same subset and the right column show the performance when the true variables fall into multiple subsets. We can see that ensuring all the true variables fall into the same subset has a more significant impact on the results for method 2 than for method 1, however there is still little difference in most cases. This is likely due to the omission of the decorrelation step. Once again increasing m improves the results which indicates that an increased value of m leads to a more robust method with respect to random variables being added.

$\tilde{p}$	m	SIR		PDWD		PSVM	
15	1	0.15 (0.046)		0.11 (0.019)		0.24 (0.156)	
	2	0.11 (0.039)	0.19 (0.119)	0.11 (0.054)	0.12 (0.033)	0.12 (0.020)	0.18 (0.075)
	3	0.12 (0.035)	0.12 (0.071)	0.11 (0.039)	0.12 (0.033)	0.12 (0.036)	0.12 (0.035)
20	1	0.27 (0.154)		0.20 (0.038)		0.30 (0.107)	
	2	0.16 (0.080)	0.18 (0.043)	0.15 (0.034)	0.19 (0.034)	0.16 (0.032)	0.29 (0.147)
	3	0.19 (0.085)	0.23 (0.122)	0.18 (0.058)	0.17 (0.055)	0.16 (0.039)	0.17 (0.038)
30	1	0.42 (0.349)		0.24 (0.041)		0.52 (0.211)	
	2	0.28 (0.222)	0.30 (0.083)	0.21 (0.038)	0.25 (0.047)	0.46 (0.195)	0.38 (0.172)
	3	0.22 (0.077)	0.38 (0.377)	0.22 (0.027)	0.25 (0.046)	0.27 (0.079)	0.23 (0.049)
110	1	-		-		-	
	2	0.77 (0.298)	0.79 (0.315)	0.61 (0.222)	0.73 (0.201)	0.60 (0.328)	0.58 (0.205)
	3	0.83 (0.377)	0.92 (0.360)	0.63 (0.153)	0.58 (0.206)	0.77 (0.220)	0.95 (0.311)

Table 5.16: Comparison of different amounts of random data for PDWD, PSVM and SIR. The table shows the mean performances (standard deviation in parenthesis) of each method of 100 iterations. Left column: all true variables fall into same subset, right column: true variables fall into different subsets.

5.4 Comparison of previous methods

We have previously conducted extensive simulation studies for both methods 1 and 2, to compare each method with $m = 1$. We will now turn our attention to the accuracy of method 1 compared with method 2. Figure 5.5 shows the performance of method 1 against method 2 of 720 scenarios, for all models, multiple choices of p, m, r and s, and all the classic methods used previously. The chart indicates that method 2 produces better results more often than method 1, where method 2 performs better 76.8% of the time and the mean distances of each method is 1.23 and 1.05, respectively.

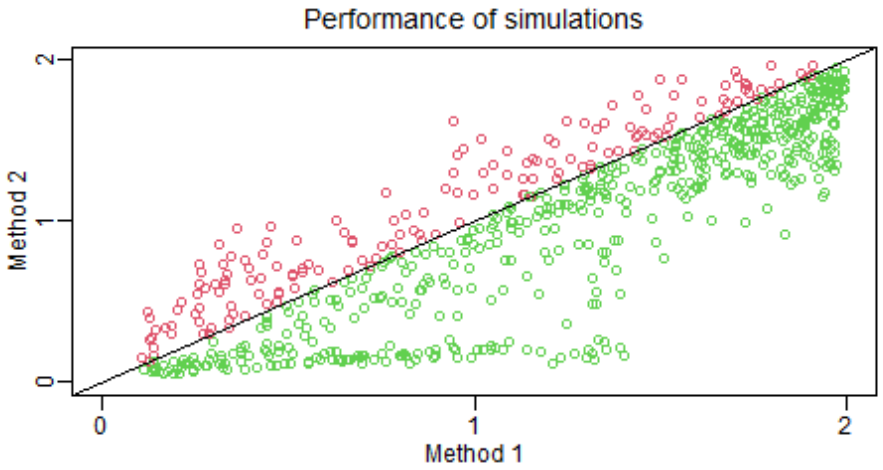

Figure 5.5: Performance of our method of 720 scenarios for all models and multiple choices of p, m, r, s, including results for SIR, PDWD and PSVM. Green points indicate results where the performance of our method was better than not separating and red show the alternative.

It is clear that method 2 performs better than method 1, however for which scenarios it performed better is also of interest. Figure 5.6 shows the results of figure 5.5 broken down by model, p, m, r, s and classic method. Again, we see that the choice of s does not seem to effect the distribution of the results, however we do see that SIR shows stronger results for method 1 than method 2 when compared with PDWD and PSVM. As expected, we see that method 1 performs better for $r = p$ than for $r = 2$, which was clear in previous analysis. This mirrors the results we saw earlier when comparing each method to the dimension reduction techniques performed with no separation.

The spread of the performances seems to be similar for models I, II and III, with method 1 showing better results for all three. Finally, it appears that as p and m increase, the number of scenarios where method 2 outperforms method 1 also increases.

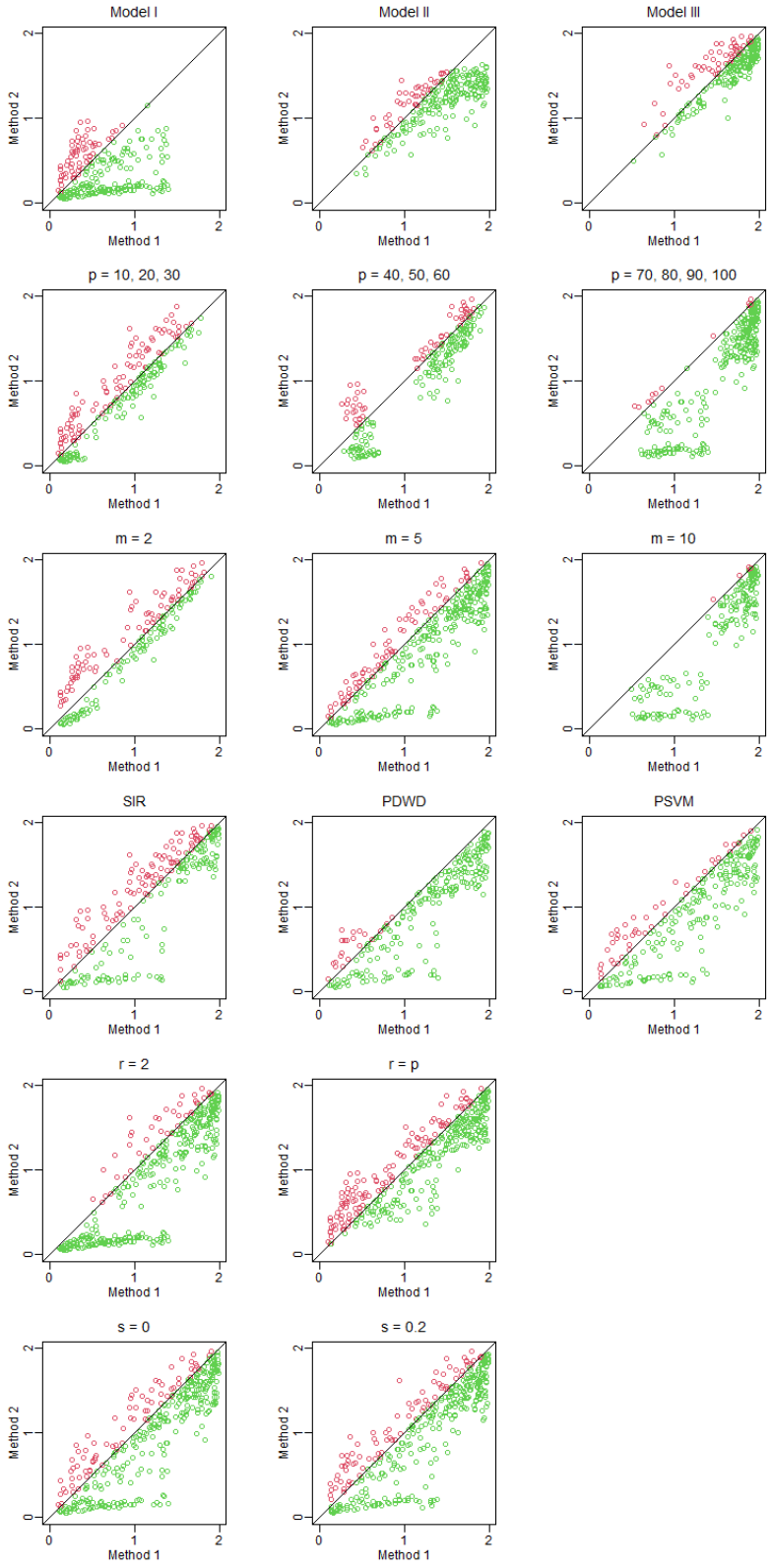

Figure 5.6: Performance of our method of 720 scenarios by model, p, m, r, s and choice of classic method. Green points indicate results where the performance of our method was better than not separating and red show the alternative.

Chapter 6

Conclusion

6.1 Recap of work

6.1.1 Development of new linear method

The work produced by Li et al. (2011) using classification methods as tools for dimension reduction yielded a great improvement of accuracy compared with more classic methods. We noticed that a significant drawback of this work was the extensive computational costs compared with many methods, including Li (1991). Our aim was to try a similar approach which maintained the accuracy of PSVM without the heavy computational cost. By comparing SVM and DWD in the classification setting we found that a significant difference was that DWD is smooth and therefore differentiable everywhere. This opens the door for algorithms that find the zeroes of the derivative of the optimisation problem instead of solving the optimisation problem directly. Wang and Zou (2015) developed a new algorithm for DWD which takes advantage of DWD's differentiability and strict convexity to produce a much faster algorithm for solving the DWD objective function.

Given that the DWD classification method was significantly more computationally efficient than the SVM algorithm it presented a possibility of achieving our aim of developing a faster dimension reduction technique. As discussed in Marron et al. (2007), DWD was proposed as an alternative for SVM when p was large since SVM suffers from data piling as p increases. This fact was of interest since we could also assess the impact of data piling in a dimension reduction setting. Using the framework create by Li et al. (2011), we were able to begin by developing a linear dimension reduction technique using DWD, and prove the consistency of our estimator in the linear case.

The synthetic simulation studies that we ran compared both PDWD and PSVM. The results showed many advantages of our method over the method proposed by Li et al. (2011), with respect to accuracy but more importantly time. We found that

PDWD often produce higher accuracy results and showed significant improvements computationally, especially as n increased. The real data analysis also implied that PDWD was more robust against noise with respect to random features.

Qiao and Zhang (2015) developed a new classifier in an attempt to manipulate the positive features of both SVM and DWD into one classifier since DWD is more sensitive to data with an uneven amount of samples in class than another. In an attempt to investigate the effects of this sensitivity in the dimension reduction setting we also considered the weighted DWD objective function proposed by Qiao et al. (2010). The introduction of the weights into the objective function showed little change in the results in the dimension reduction setting and hence the results have been omitted.

6.1.2 Non-linear principal distance-weighted discrimination

Once we had confirmed that linear PDWD showed improvements compared with PSVM, it was beneficial to extend this into the non-linear setting under a unified framework. Analogous to the finding of Li et al. (2011), much of the theory in the non-linear setting followed a similar structure to the linear case. The simulation studies that we performed, simply mirrored the findings that we had discovered in the linear case, where PDWD often outperformed PSVM.

6.1.3 Separation of feature space

As stated, PDWD is computationally faster that PSVM which was a vast improvement however we also proved that our PDWD estimator only remains consistent whilst $p < n$. Countering this restriction is a particular area of interest in dimension reduction due to the drastic increase in the dimension of data which has occurred in recent years. The work proposed by Yin and Hilafu (2015) introduced a sequential method for dimension reduction by sub-setting the features. We aimed to adapt a method of dimension reduction by separating the feature space, but in an attempt to also improve the computational efficiency, our method is instead a parallel programming problem.

Using the concept proposed by Wang et al. (2016), our method begins by decorrelating the variables before sub-setting the features. We then performed a standard SDR method on each subset on separate machines and recollected the outputs. We realised that since the dimension reduction method was being performed on separate subsets, the method may miss any relationships between variables in different subsets. Therefore, our original method was not estimating the minimum central dimension reduction subspace. To fix this problem we added a second step making our problem partially sequential. Our original concerns of the enforced sequential nature

of our method having adverse effects on the computation time were eased after extensive synthetic simulation studies were performed. This method also showed positive effects on the accuracy, with 70.6% of the simulations producing better results than without separation.

6.1.4 Separation of feature space without decorrelation

Our aims when first partitioning the feature space was to produce a method which remains a viable option when p surpasses n. Unfortunately, the decorrelation step reintroduced this restriction which led us to investigate another approach. Skipping the decorrelation step would help loosen this restriction and increase the computational time, however we were unsure of the effect this would cause on the accuracy of the method.

The theory of the previous method depends heavily on the decorrelation of the features, which led us to begin with accessing the simulation results of omitting this step. As expected we found that skipping the decorrelation step greatly reduced the computational time for all classic methods tested. Surprisingly, this approach had positive effects on the accuracy compared with separation with decorrelation and no separation. We found that omitting the decorrelation step out performed the method with decorrelation 76.8% of the time and outperformed the methods with no separation in 85.7% of the simulations.

6.2 Work still to consider

6.2.1 Sufficient dimension reduction using FLAME

Our research into PDWD with weights implied that the sensitivity that DWD suffers from with imbalanced data in the classification setting is not replicated in a dimension reduction setting. However, more investigation into this proposition using the FLAME estimator may yield interesting features that we have not considered.

6.2.2 Methodology of feature space partitioning without decorrelation

Our simulation experiments into the performance of our dimension reduction method through feature space partitioning without decorrelation unfortunately came close to the end of this work. Given more time we would have attempted to formulate some theory for this method since the benefits made clear by the simulations are significant. Our investigation into the theory of this method implies that any theory

that can be produced will need to be different from the theory with the decorrelation step.

Bibliography

Ahn, J. and Marron, J. S. (2005). The direction of maximal data piling in highe-dimensional spaces. Technical report, University of North Carolina, Dept. of Statistics and Operational Research.

Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404.

Artemiou, A. and Dong, Y. (2016). Principal L_q support vector machines for sufficient dimension reduction. *Electronic Journal of Statistics*, 10:783–805.

Artemiou, A., Dong, Y., and Shin, S. J. (2020). Real-time sufficient dimension reduction through principal least squares support vector machines. *Pattern Recognition*. accepted.

Artemiou, A. and Shu, M. (2014). A cost based reweighted scheme of principal support vector machine. In *Topics in Nonparametric Statistics, Springer Proceedings in Mathematics and Statistics*.

Artemiou, A. and Tian, L. (2015). Using sliced inverse mean difference for sufficient dimension reduction. *Statistics and Probability Letters*, 106:184–190.

Babos, S. and Artemiou, A. (2020). Sliced inverse median difference regression. *Statistical Methods and Applications*, 29(4):937–954.

Berlinet, A. and Thomas-Agnan, C. (2004). *Reproducing kernel Hilbert spaces in probability and statistics*. Springer.

Bura, E. and Yang, B. J. (2011). Dimension estimation in sufficient dimension reduction: A unified approach. *Journal of Multivariate Analysis*, 102:130–142.

Burgess, C. J. C. and Crisp, S. J. (1999). Uniqueness of the svm solution. *In proceedings of neural information processing systems*, 12:223–229.

Cook, R. D. (1996). Graphics for regressions with a binary response. *Journal of the American Statistical Association*, 91:983–992.

Cook, R. D. (1998). *Regression Graphics: Ideas for studying regressions through graphics*. New York: Wiley.

Cook, R. D. and Weisberg, S. (1991). Discussion of "sliced inverse regression for dimension reduction". *Journal of the American Statistical Association*, 86:316–341.

Fukumizu, K., Bach, F. R., and Jordan, M. I. (2009). Kernel dimension reduction in regression. *The Annals of Statistics*, 37:1871–1905.

Guo, Z., Li, L., Lu, W., and Li, B. (2015). Groupwise dimension reduction via envelope method. *Journal of the American Statistical Society*, 110:1515–1527.

Hilafu, H. and Yin, X. (2013). Sufficient dimension reduction in multivariate regression with categorical predictors. *Computational Statistics and Data Analysis*, 63(C):139–147.

Hjort, N. L. and Pollard, D. (1993). Asymptotics for minimisers of convex processes. Technical report, Yale University.

Jiang, B., Zhang, X., and Cai, T. (2008). Estimating the confidence interval for prediction errors of support vector machine classifiers. *Journal of Machine Learning Research*, 9:3182–3210.

Jin, J., Ying, C., and Yu, Z. (2019). Distributed estimation of principal support vector machines for sufficient dimension reduction. *arXiv preprint arXiv:1911.12732*.

Li, B. (2018). *Sufficient dimension reduction: Methods and applications with R*. CRC Press.

Li, B., Artemiou, A., and Li, L. (2011). Principal support vector machines for linear and non-linear sufficient dimension reduction. *The Annals of Statistics*, 39:3185–3210.

Li, B. and Wang, S. (2007). On directional regression for dimension reduction. *Journal of the American Statistical Association*, 102:997–1008.

Li, B., Wen, S., and Zhu, L. (2008). On a projective resampling method for simension reduction with multivariate responses. *Journal of the American Statistical Association*, 103:1177–1186.

Li, B., Zha, H., and Chiaromonte, F. (2005). Contour regression: A general approach to dimension reduction. *The Annals of Statistics*, 33:1580–1616.

Li, K. C. (1991). Sliced inverse regression for dimension reduction (with discussion). *Journal of the American Statistical Association*, 86:316–342.

Li, K. C. (1992). On principal hessian directions for data visualisation and dimension refuction: Another application of stein's lemma. *Journal of the American Statistical Association*, 87:1025–1039.

Li, L., Li, B., and Zhu, L.-X. (2010). Groupwise dimension reduction. *Journal of the American Statistical Association*, 105(491).

Lin, Q., Zhou, Z., and Liu, J. S. (2019). Sparse sliced inverse regression via lasso. *Journal of the American Statistical Association*, 114(528):1726–1739.

Liquet, B. and Saracco, J. (2016). Big-sir a sliced inverse regression approach for massive data. *Statistics and its Interface*, 9:509–520.

Luo, W. and Li, B. (2016). Combining eigenvalues and variation of eigenvectors for order determination. *Biometrika*, 103(4):875–887.

Marron, J. S., Todd, M. J., and Ahn, J. (2007). Distance weighted discrimination. *Journal of the American Statistical Association*, 102:1267–1271.

Pircalabelu, E. and Artemiou, A. (2021). The lasso psvm approach for sufficient dimension reduction using principal projections. submitted.

Qiao, X., Zhang, H. H., Liu, Y., Todd, M. J., and Marron, J. S. (2010). Weighted distance weighted discrimination and its asymptotic properties. *Journal of the American Statistical Association*, 105:401–414.

Qiao, X. and Zhang, L. (2015). Flexible high-dimensional classification machines and their asymptotic properties. *Journal of Machine Learning Research*, 16:1547–1572.

Randall, H., Artemiou, A., and Qiao, X. (2020). Sufficient dimension reduction based on distance-weighted discrimination. *Scandinavian Journal of Statistics*. accepted.

Shin, S. J. and Artemiou, A. (2017). Penalised principal logistic regression for sparse sufficient dimension reduction. *Computational Statistics and Data Analysis*, 111:48–58.

Shin, S. J., Wu, Y., Zhang, H. H., and Liu, Y. (2017). Principal weighted support vector machines for sufficient dimension reduction in binary classification. *Biometrika*, 104:67–87.

Smallman and Artemiou, A. (2017). A study on imbalance support vector machine algorithms for sufficient dimension reduction. *Communications in Statistic and Theory and Methods*, 46:2751–2763.

Suykens, J. A. K., Gestel, T. V., Brabanter, J. D., Moor, N. D., and Vandewalle, J. (2002). *Least square support vector machines*. World scientific pub. co, Singapore.

Vapnik, V. N. (1998). *Statistical learning theory*. Wiley: New York.

Wang, B. and Zou, H. (2015). Another look at distance weighted discrimination. *Journal of the Royal Statistical Society, Series B*, 80:177–198.

Wang, B. and Zou, H. (2016). Sparse distance weighted discrimination. *Journal of Computational and Graphical Statistics*, 25:826–838.

Wang, Q. and Yin, X. (2008). A nonlinear multi-dimensional variable selection method for high dimensional data: Sparse mave. *Computational Statistics and Data Analysis*, 52:4512–4520.

Wang, X., Dunson, D., and Leng, C. (2016). Decorrelated feature space partitioning for distributed sparse regression. In *30th Conference on Neural Information Processing Systems (NIPS 2016)*, Barcelona, Spain.

Wu, H. M. (2008). Kernel sliced inverse regression with applications on classification. *Journal of Computational and Graphical Statistics*, 17:590–610.

Ye, Z. and Weiss, R. E. (2003). Using boostrap to select one of a new class of dimension reduction methods. *Journal of the American Statistical Association*, 98:968–979.

Yeh, I. C. (1998). Modelling of strength of high performance concrete using artificial neural networks. *Cement and Concrete Research*, 28:1797–1808.

Yeh, Y. R., Huang, S. Y., and Lee, Y. Y. (2009). Nonlinear dimension reduction with kernel sliced inverse regression. *IEEE Transactions on Knowledge and Data Engineering*, 21:1590–1603.

Yin, X. and Hilafu, H. (2015). Sequential sufficient dimension reduction for large p, small n problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology*, 77:879–892.

Yin, X., Li, B., and Cook, R. D. (2008). Successive direction extraction for estimating the cental subspace in a multi-index regression. *Journal of Multivariate Analysis*, 99:1733–1757.

Zhou, J. and Zhu, L. (2016). Principal minimax support vector machines for sufficient dimension reduction with contaminated data. *Computational Statistics and Data Analysis*, 94:33–48.

Zhu, A., Miao, B., and Peng, H. (2006). On sliced inverse regression with high-dimensional covariates. *Journal of the American Statistical Association*, 101:630–643.