PLS methods in regression
Model assessment and inference

Thèse présentée à l'Institut de Statistique
Faculté des sciences économiques
Université de Neuchâtel

Pour l'obtention du grade de docteur en statistique

Par

Athanassios KONDYLIS

Acceptée sur proposition du jury :

Yadolah Dodge	Université de Neuchâtel	directeur de thèse
Yves Tillé	Université de Neuchâtel	rapporteur interne président du jury
Ali S. Hadi	Cornell University - AUC	rapporteur externe
Michel Tenenhaus	HEC Paris	rapporteur externe
Joe Whittaker	Lancaster University	rapporteur externe

Soutenue le 14 septembre 2006

Université de Neuchâtel
2006

Université de Neuchâtel unine

IMPRIMATUR POUR LA THESE

PLS methods in regression : model assessment and Inference

Athanassios KONDYLIS

UNIVERSITE DE NEUCHATEL

FACULTE DES SCIENCES ECONOMIQUES

La Faculté des sciences économiques,
sur le rapport des membres du jury

MM. Y. Dodge (directeur de thèse, Université de Neuchâtel)
Y. Tillé (Université de Neuchâtel)
A. Hadi (Cornell University et Université Américaine du Caire)
M. Tenenhaus (Hautes Ecoles Commerciales, Paris)
J. Whittacker (Lancaster University)

Autorise l'impression de la présente thèse.

Neuchâtel, le 14 septembre 2006

Le doyen

Kilian Stoffel

■ Téléphone : +41 32 718 15 00 ■ Fax : +41 32 718 15 01 ■ E-mail : secretariat.seco@unine.ch ■ www.unine.ch

Mots clés : Régression PLS, régression sur composantes principales, régularisation, choix du modèle, norme L_1, régression LAD, valeurs aberrantes, l'algorithme de BACON, espace de Krylov, espace engendré par les vecteurs propres.

Keywords: Partial least squares, principal components, shrinkage, model selection, LAD regression, BACON algorithm, Krylov space, eigen space, preconditioning.

Résumé : Cette thèse propose l'extension de la régression PLS (Partial Least Squares) vers trois directions :

1. l'utilisation de la norme L_1 dans le contexte de PLS qui aboutit à la régression PLAD,

2. l'application de l'algorithme de BACON pour la détection des valeurs aberrantes ce qui permet de robustifier la régression PLS,

3. l'utilisation des solutions des systèmes d'équations conditionnées en utilisant des approximations à travers des espaces de Krylov. Ces approximations permettent de simplifier l'interprétation des modèles PLS et de ses coefficients, ainsi que de mettre en valeur le lien entre la régression PLS et la régression sur composantes principales.

Le choix du modèle final PLS et l'estimation de sa performance sont réalisés pour les méthodes proposées, et testés sur des données réelles et simulées. Pour introduire le lecteur dans l'univers de notre recherche, on commence avec un chapitre de notations (chapitre 1). Ensuite on donne une présentation générale du problème de régression et de la nécessité de régularisation (chapitre 2), et une vue d'ensemble sur la méthode de PLS (chapitre 3). Les extensions mentionnées ci-dessus se trouvent aux Chapitres 4, 5 et 6. Conclusions et sujets de recherches futures sont donnés au Chapitre 7.

Abstract

Regression modelling is one of the main applications in statistics. It seeks to relate a set of predictors to a response by estimating a statistical function. The latter allows to predict future outcomes for a set of new input, and to interpret the contribution of the predictors on the final outcome. Among numerous regression methods Partial Least Squares (PLS) is at the forefront of exciting topics in modern statistics research. This is mainly due to the fact that PLS methods are dimension reduction methods that provide reliable solutions in cases where the recorded predictors are too many and highly collinear. PLS methods in regression are indeed very efficient in cases where the number of the predictors exceed the available observations. Modern experiments are based on high dimensional records and the use of techniques such as PLS becomes therefore essential.

The present research consists of three parts. The first part extends the PLS regression towards regression modelling based on L_1 norm which is associated with the Least Absolute Deviation (LAD) regression. The second part concerns the sensitivity of PLS regression with respect to outliers. It provides a robust PLS regression method based on the BACON algorithm. Finally, the third part uses preconditioning techniques for the solution of ill posed problems using Krylov subspace iterative methods. It chooses preconditioned Krylov spaces to propose methods for the detection of predictors' intervals of no statistical importance. This method improves substantially the interpretation of the PLS regression coefficients. Furthermore preconditioned Krylov spaces provide a unifying approach for PC and PLS regression.

Model selection and assessment, as well as the statistical properties of the proposed methods are investigated and tested on real and experimental data. The proposed extensions are given after regression modelling and PLS regression are reviewed.

Key words: Partial least squares, Principal components, shrinkage, model selection, LAD regression, BACON algorithm, Krylov space, eigen space, preconditioning.

Acknowledgments

I would like to deeply thank my advisor Professor Yadolah Dodge for suggesting me this topic and giving me the opportunity to continue my statistical education up to the doctoral level. His choice to take me as his PhD student has been from the very beginning motivating. Professor Yves Tillé has importantly sustained my effort to finish my PhD by giving me all the necessary means and by installing a creative working environment in which I ended my research. I would like to thank him a lot for this and to wish him the best for himself and the Institute of Statistics. Besides the internal professors I have had the chance to have in the scientific committee brilliant professors and experts such as Ali S. Hadi, Michel Tenenhaus, and Joe Whittaker. I thank them all for their cooperation, their will and determination to work and exchange ideas with me. I am grateful to them for our fruitful discussions and for their essential contributions. I would like especially to thank Joe Whittaker for his unique ability in teaching statistics and his outstanding consistency in his work. Many thanks are additionally addressed to Ali Hadi for his very significant role in my research work, and his excellent communication skills.

All these years I have been surrounded by a good and international team of brilliant statisticians that have been visiting the University of Neuchâtel either to give courses for the Master degree in Statistics or to give seminar lectures in the Institute of Statistics. I am grateful to most of them for their correspondence and their encouragement. The working conditions in the University of Neuchâtel have been excellent, and the environment within the Institute of Statistics has been very good for doing a scientific research. I thank therefore the University of Neuchâtel, its students as well as my colleagues. Especially, I want to thank Alina, Hocine, and Soazig.

Last, but most importantly, I thank my parents and my whole family, with a special note to my cousin Pothitos who has armed me from my childhood with a human *sheltering sky* and have motivated me to chase my goals with lot of understanding. I would like finally to thank Alina who let me recover an important balance in life and see things in a new and positive way.

Neuchâtel, September 21, 2006

Contents

Chapter 1

Introduction

1.1 Notational preliminaries

In what follows the basic notations which are used throughout the whole thesis are given.

Bold faced lower case symbols are vectors, upper case are matrices. The vector $\boldsymbol{x}_i \in \mathbb{R}^p$ therefore denotes the i^{th} data vector which includes p real elements, the index $i = 1, \ldots, n$ denotes observations, with n being the sample size. We additionally use the index $j = 1, \ldots, p$ to denote predictors. In line with this notation we can write the vector $\boldsymbol{x}_j = (x_{1j}, \ldots, x_{ij}, \ldots, x_{nj})'$, where the superscript $'$ is used to denote the transpose of a vector. It is equally used to denote the transpose of matrices. The data set are written in a matrix form as $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p)$. In regression problems, on which we focus, we use $\boldsymbol{X}$ to denote the predictors matrix, while we use $\boldsymbol{y}$ for the response vector. For multivariate regression problems we use $\boldsymbol{Y}$ instead of $\boldsymbol{y}$ with q denoting the number of columns in $\boldsymbol{Y}$.

It is commonly used in practice to use upper case letters for random variables and bold faced lower case letters for their realizations. For example, the response vector $\boldsymbol{y}$ is the realization of the random variable Y. This generalizes to all vectors. As a rule of thumb we use greek letters in order to denote unknown parameters. We then add the hat symbol when these are estimated. For example in a regression problem $\boldsymbol{\beta}$ commonly denotes the regression coefficient vector which, when it is estimated, is denoted by $\widehat{\boldsymbol{\beta}}$. The use of the hat symbol to denote estimates is generalized to all letters. We additionally use the notation $\mathsf{Prob}(X = x)$ in order to denote the probability that the random variable X takes the value x (for a multivariate random variable X the realization corresponds to a vector $\boldsymbol{x}$), while by $\mathsf{E}(X)$ we denote the expectation of X. Generally, expectations are denoted by $\mathsf{E}(.)$, while the subscript n, when it appears, it is used to emphasize that the expectations are taken over sample quantities. So $\mathsf{E}_n(\cdot)$ is a sample mean, and $\mathsf{var}_n(\cdot)$, $\mathsf{sd}_n(\cdot)$, $\mathsf{cov}_n(\cdot|\cdot)$, and $\mathsf{corr}_n(\cdot|\cdot)$ denote the sample variance, standard deviation, covariance, and correlation, respectively. The latter hold for

vectors. For multivariate data we denote the variance-covariance matrix and the correlation matrix by $\mathbf{\Sigma}$ and $\mathbf{R}$, respectively. Covariance and correlation matrices are often decomposed in submatrices. For example for a multivariate regression problem with data (X, Y) arising from a joint multivariate normal distribution the covariance matrix may be decomposed as follows

$$\begin{pmatrix} \mathbf{\Sigma}_{XX} & \mathbf{\Sigma}_{XY} \\ \mathbf{\Sigma}_{YX} & \mathbf{\Sigma}_{YY} \end{pmatrix},$$

with $\mathbf{\Sigma}_{XY} = \mathbf{\Sigma}_{YX}{}^T$ and $\mathbf{\Sigma}_{XX}$ and $\mathbf{\Sigma}_{YY}$ denoting the variance-covariance matrix for the predictors and the responses, respectively. The correlation matrix $\mathbf{R}$ is similarly decomposed. In most cases considered in this thesis the data are initially properly transformed. That is, they are centered or standardized. Without any loss of generality and unless it is otherwise noted, we assume data to be centered by subtracting expectations, that is the random variables X and Y correspond to

$$X = X - \mathsf{E}_n(X) \quad \text{and} \quad Y = Y - \mathsf{E}_n(Y).$$

By $\mathsf{E}_n(\cdot|\cdot)$ we denote the sample least squares predictor, which corresponds to the least squares fitted values commonly denoted by $\widehat{\boldsymbol{y}}$. As it will be clear in the following chapters when regression methods are applied the predictors X may be regarded as fixed or random. In the former case we use $\mathsf{E}_X(\cdot|\cdot)$ for the predictor, while in the latter we use $\mathsf{E}_{XY}(\cdot|\cdot)$. Fitted values resulting from regression methods other than least squares will be mentioned by a superscript in the right side of $\widehat{\boldsymbol{y}}$. For example $\widehat{\boldsymbol{y}}^{lad}$ corresponds to the LAD fitted values vector. The same holds for estimated parameters, for example the ordinary least squares and the PLS regression coefficients will be denoted by $\widehat{\boldsymbol{\beta}}^{ls}$ and $\widehat{\boldsymbol{\beta}}^{pls}$, respectively.

In certain cases, the regression estimates depend upon an hyperparameter, generally denoted here for illustration purposes by ℓ. We use the notation $\widehat{\boldsymbol{y}}_\ell$ or $\widehat{f}(\boldsymbol{X}, \ell)$ in order to emphasize that fitted values or the estimate $\widehat{f}$ depend upon the hyperparameter ℓ. The use of $\boldsymbol{X}$ in the preceding notation allows to indicate whether the original (training) data $\boldsymbol{X}$ are used or if external validation based on test data $\boldsymbol{X}^\star$ is used instead. In order to validate regression models loss functions are used. These are generally denoted by the symbol $\mathcal{L}$ and in the regression context this transforms to $\mathcal{L}(Y, \widehat{f}(\boldsymbol{X}, \ell))$. Given some training data $(\boldsymbol{X}, \boldsymbol{y})$ the loss $\mathcal{L}(\boldsymbol{y}, \widehat{f}(\boldsymbol{X}, \ell))$ or equivalently $\mathcal{L}(\boldsymbol{y}, \widehat{\boldsymbol{y}}_\ell)$ gives the loss for a regression model based on the estimated regression function $\widehat{f}(\boldsymbol{X}, \ell)$. If external validation is applied by means of a test set $(\boldsymbol{X}^\star, \boldsymbol{y}^\star)$ the loss is written as $\mathcal{L}(\boldsymbol{y}^\star, \widehat{\boldsymbol{y}}^\star_\ell)$.

We finally borrow some notation from the numerical literature for which the math serif letters are mostly used. We use the system

$$\mathsf{A}\mathsf{z} = \mathsf{b}$$

for a square symmetric matrix A and the vectors z and b of appropriate dimensions, and we relate this system to the least squares problem for $\mathsf{A} = \boldsymbol{X}'\boldsymbol{X}$ and

$\mathsf{b} = \boldsymbol{X}'\boldsymbol{y}$. We denote by A^{-1} the inverse of matrix A. In case that it does not exist we use A^{-} to denote the generalized inverse of A. We also denote by $\| \mathsf{z} \|$ the common L_2 or euclidian norm for the vector $\mathsf{z} = (\mathsf{z}_j)_{j=1}^{p}$. The use of a subscript q extends to norms other than the L_2 norm. Generally the L_q norm of a vector is denoted by $\| \mathsf{z} \|_q$ and it is equal to

$$\| \mathsf{z} \|_q = \left(\sum_{j=1}^{p} |\mathsf{z}_j|^q \right)^{(1/q)} .$$

When two vectors z_1 and z_2 are orthogonal we have $\mathsf{z}_1'\mathsf{z}_2 = 0$, and we use the notation $\mathsf{z}_1 \perp \mathsf{z}_2$. If z_1 and z_2 are A-orthogonal we write $\mathsf{z}_1 \perp_{\mathsf{A}} \mathsf{z}_2$ that follows from $\mathsf{z}_1' \mathsf{A} \mathsf{z}_2 = 0$. The symbols $\mathcal{R}(.)$ and $\mathcal{C}(.)$ denote the space spanned by the rows and the columns of a matrix, respectively. In general, given that a space S is spanned by a set of vectors $\boldsymbol{u}_i$ with $i = 1, \ldots, n$, we write

$$\mathsf{span}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n) = \mathsf{S}.$$

We also use $\pi(\mathsf{A})$ in order to denote the polynomial of matrix A, while by $\chi(\mathsf{A})$ we denote the characteristic polynomial for matrix A.

Further notations are provided when needed. Chapter 6 for example includes in its first paragraph further notations necessary in order to follow the application of preconditioning methods in PLS regression.

1.2 Overview

Chapter 2: The general regression problem

Chapter 2 gives the general framework of the regression problem. This is presented together with model selection and assessment. The use of linear regression is analyzed and extensions of the linear regression are then briefly given. The presence of multicollinear and high dimensional data in regression is highlighted. Regularized or shrinkage regression methods are then presented in order to overcome the problems arising from collinearity and high dimensional data. Finally, outlier detection methods and robust regression are briefly reviewed. These methods permit to protect regression analysis from influential observations which are often present in data sets.

Chapter 3: Overview on PLS regression

Chapter 3 is an overview of PLS methods in regression. The milestones in the development of PLS methods in regression are initially given followed by the PLS regression algorithmic implementations. PLS regression is then seen as an optimization problem and its relation to similar regression methods is also presented. The statistical properties of PLS regression are also recalled. By focusing our attention on univariate regression problems, we briefly give the connection between

PLS regression and Krylov spaces, the link between PLS regression and Conjugate Gradients, and finally the strong link between PLS regression methods and the Lanczos method. Model selection and assessment techniques for PLS regression are then discussed including cross validation, bootstrap, and linearization techniques. Finally, the shrinkage properties of PLS regression are given. This overview of PLS regression is followed by real examples illustrating its statistical and shrinkage properties.

Chapter 4: The Partial LAD regression

The use of the L_1 norm criteria in the PLS regression setting is the main topic of Chapter 4. After introducing in Chapter 2 the L_1 norm based statistical applications and its connection to M-regression and the LAD regression, this chapter presents an algorithm for PLS regression based on the L_1 norm and the Least Absolute Deviation (LAD) regression. The proposed modification results to the Partial LAD (PLAD) regression method which retains the basic structure of PLS regression while it uses L_1 based estimates for both association and regression. Details and illustrative examples are given in this chapter. Real world and experimental data are used in order to assess the Partial LAD regression by means of the bootstrap.

This chapter is based on the following papers:

1. Dodge, Y. and Kondylis, A. and Whittaker, J. (2004), Extending PLS1 to PLAD regression and the use of the L1-norm in soft modelling. COMPSTAT 2004, Eds. J. Antoch, pages 935–942. Physica/Verlag, Springer,

2. Kondylis, A. and Whittaker, J. (2006), Bootstrapping Partial LAD regression, to appear in the Special Issue on PLS methods in *Computational Statistics*.

Chapter 5: PLS regression using the BACON algorithm

The BACON algorithm for outlier detection and robust estimation plays a vital role for the robust extension of PLS regression given in Chapter 5. The multivariate BACON algorithm is given in the beginning of this chapter. The proposed BACON PLS regression algorithm is then presented together with illustrative examples. An extensive simulation study is conducted in order to validate the proposed algorithm and to compare it to other robust competitors. The rank deficient case is analyzed in more detail and the BACON approach for rank deficient data is presented. The extension of BACON PLS regression to these type of data is then straightforward. Real and simulated examples illustrate its use. Finally, robust model selection based on the BACON PLS proposal are given at the end of Chapter 5.

This chapter is based on the following papers:

1. Kondylis, A. and Ali S. Hadi (2006), Derived Components Regression using the BACON algorithm, In Press in *Computational Statistics & Data Analysis*.

2. Kondylis, A. and Ali S. Hadi (2006), The BACON approach for rank deficient data, *working paper*.

Chapter 6: Preconditioning methods applied in PLS regression

The use of preconditioning techniques in linear algebra for solving systems of linear equations are used in the PLS regression setting in Chapter 6. This chapter introduces preconditioning methods and their application to Krylov spaces and Conjugate Gradients. Preconditioning Krylov spaces are used on two directions. The first one seeks to improve the interpretation of the PLS regression coefficients providing an heuristic algorithm for detection of redundant predictors' intervals in PLS regression. The second approach, uses eigenvectors as preconditioners and solves the linear system through approximations derived inside Krylov subspaces. It gives a unifying approach for Principal Components and PLS regression, and a rich ensemble of solutions arising from preconditioning the Krylov space by different eigenvectors. Real world data are used to illustrate the proposed methods.

Chapter 7: Conclusions and further research

Conclusions and further research topics are briefly discussed in Chapter 7. Conclusions arising from the methods under study are provided in order to retain the basic contributions as well as the limitations found throughout the presented research. Topics of special interest for future research are also given in Chapter 7, highlighting directions of future research in PLS methods in regression.

Bibliography, Mathematical Appendix I and II

The final part of this thesis gives a list of complete references cited inside the preceding chapters, and the Mathematical Appendix I and II with all necessary mathematical preliminaries and proofs.

Chapter 2

The general regression problem

2.1 The regression problem

Regression analysis seeks to associate a set of random variables $X = (X_1, \ldots, X_p)$, with $X \in \mathbb{R}^p$, to the response $Y \in \mathbb{R}$ via the following relation

$$Y = f(X, \boldsymbol{\theta}) + \epsilon, \tag{2.1}$$

where f is the regression function relating Y to X, $\boldsymbol{\theta}$ is a set of parameters that specify the function f, and ϵ is a random error. The regression function f is estimated by minimizing an appropriate loss function $\mathcal{L}$ between Y and $f(X, \boldsymbol{\theta})$, often called risk. A widely used loss function is given by the squared loss according to

$$\mathcal{L}(Y, f(X, \boldsymbol{\theta})) = (Y - f(X, \boldsymbol{\theta}))^2 . \tag{2.2}$$

Other loss functions may replace the squared loss. For example one can use the L_1 loss function given by

$$\mathcal{L}(Y, f(X, \boldsymbol{\theta})) = |Y - f(X, \boldsymbol{\theta})|.$$

The regression problem in (2.1) depends on the nature of the measurements in X. We distinguish between the case where X is fixed and the case where X is random. In the latter case X is a random variable with realizations $\boldsymbol{x}_i$ for $i = 1, \ldots, n$, which are assumed to be independent and identically distributed. Moreover the random variables (X, Y) are assumed to have a joint probability distribution probability function depending on the parameter vector $\boldsymbol{\theta}$. This scenario is the one which we investigate. In order to make the presentation of the regression problem easier, and without loss of generality, we take $\boldsymbol{\theta}$ to be the regression coefficient vector commonly denoted by $\boldsymbol{\beta}$.

A question which arises is how to choose the estimated model $\widehat{f}$. A natural choice is to minimize the expected prediction error (the empirical risk) which is given by

$$\widehat{f} \,:\, \arg \min_{f \in \mathcal{F}} \left\{ \mathsf{E}_{XY} \left(\mathcal{L}\left[Y, f(X, \boldsymbol{\beta})\right] \right) \right\}, \tag{2.3}$$

for a set of functions $\mathcal{F}$. Taking the squared loss function and conditioning on X we get

$$\arg \min_{f \in \mathcal{F}} \left\{ \mathsf{E}_X \; \mathsf{E}_{Y|X} \left((Y - f(X, \boldsymbol{\beta}))^2 | \, X \right) \right\}, \tag{2.4}$$

and the estimated function $\widehat{f}$ is simply the conditional expectation given by

$$\widehat{f}_n = \mathsf{E}_n(Y \,|\, X). \tag{2.5}$$

The subscript n in the last expression indicates that the conditional expectations are taken over the training data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$. Hence, expectation is approximated by averaging over the training sample units.

For simplicity, the notation $Y = f(X)$ will often replace $\widehat{f}(X, \boldsymbol{\beta})$. Going back to expressions (2.4) and (2.5), and averaging over the training data the Mean Squared Error (MSE) for the chosen function can be shown to have the following bias-variance decomposition

$$\begin{aligned} \mathrm{MSE}(\widehat{f}) &= \mathsf{E}_X \; \mathsf{E}_{Y|X} \left((Y - \widehat{f}(X, \boldsymbol{\beta}))^2 | \, X \right) \\ &= \mathsf{E}_X \left[\left(f(X) - \widehat{f}_n(X) \right)^2 + \left(\widehat{f}_n(X) - \mathsf{E} \left[\widehat{f}_n(X) \right] \right)^2 \right] \\ &= \mathsf{E}_X \left[\mathsf{bias}^2 \left(\widehat{f}_n(X) \right) + \mathsf{var} \left(\widehat{f}_n(X) \right) \right]. \end{aligned}$$

The bias component may be interpreted as the error between the fitted approximation and the true function also called the *model bias*, while the second term corresponds to the error between the average estimate $\mathsf{E} \left(\widehat{f}_n(X) \right)$ and the fitted function $\widehat{f}_n(X)$. This is called the *estimation bias*.

A similar decomposition arises when one assets the Expected Prediction Error (EPE) in (2.4) of the estimated regression function $\widehat{f}$ at a data point $\boldsymbol{x}_0$, that is

$$\begin{aligned} \mathrm{EPE}(\widehat{f}(\boldsymbol{x}_0)) &= \mathsf{E}_X \left((Y - \widehat{f}(X))^2 | \, X = \boldsymbol{x}_0 \right) \\ &= \sigma_\epsilon^2 + \left(\mathsf{E}_X \left[\widehat{f}(\boldsymbol{x}_0) \right] - f(\boldsymbol{x}_0) \right)^2 + \mathsf{E}_X \left(\widehat{f}(\boldsymbol{x}_0) - \mathsf{E} \left[\widehat{f}(\boldsymbol{x}_0) \right] \right)^2 \\ &= \sigma_\epsilon^2 + \mathsf{bias}^2 \left(\widehat{f}(\boldsymbol{x}_0) \right) + \mathsf{var} \left(\widehat{f}(\boldsymbol{x}_0) \right). \end{aligned} \tag{2.6}$$

The first term σ_ϵ^2 is the irreducible error term, the second is the squared bias induced by $\widehat{f}(X)$, and the last term is the variance of $\widehat{f}(X)$ around its mean value $\mathsf{E} \left(\widehat{f}(\boldsymbol{x}_0) \right)$. The above expression reveals that good prediction may be reached by a good compromise between bias and variance of the estimated regression function.

2.1.1 Linear regression & beyond

Linear regression models assume that function f is linear in the parameter $\boldsymbol{\theta}$. This is often a convenient approximation to the truth. That means that despite the fact that the true functional may not be linear, a linear approximation

captures sufficiently the relation between X and Y. Therefore, linear regression has been widely used in statistics. Linear regression models are linear on the estimated parameters, and can be generally represented as

$$f(X) \approx X\boldsymbol{\beta}, \tag{2.7}$$

with the regression parameter vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_j, \ldots, \beta_p)$ assumed to have a linear effect on the response Y. Using the linear function (2.7) in equation (2.2) and differentiating for the regression coefficient vector $\boldsymbol{\beta}$ one gets

$$\widehat{\boldsymbol{\beta}} = \mathsf{E}(XX')^{-1}\,\mathsf{E}(XY).$$

Letting $\boldsymbol{x}_i \in \mathbb{R}^p$ denote the i^{th} row vector in matrix $\boldsymbol{X}$, and by replacing in the expression above the expectations E by averages over the training data E_n one recovers the least squares solution

$$\widehat{\boldsymbol{\beta}}^{ls} = \left(\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i'\right)^{-1} \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{y}, \tag{2.8}$$

which corresponds to the solution to the normal equations

$$\frac{1}{n}\boldsymbol{X}'\boldsymbol{X}\,\boldsymbol{\beta} = \frac{1}{n}\boldsymbol{X}'\boldsymbol{y}. \tag{2.9}$$

The least squares solution is important in regression modelling due to the Gauss-Markov theorem, reproduced below.

Theorem 2.1 *(Gauss-Markov)*
For the linear regression model with $\mathsf{E}_n(\boldsymbol{y}|\boldsymbol{X}) = \boldsymbol{X}\,\widehat{\boldsymbol{\beta}}$, $\mathsf{var}(\boldsymbol{y}|\boldsymbol{X}) = \sigma_\epsilon^2\,\boldsymbol{I}_p$, $\boldsymbol{s}'\boldsymbol{X}\,\widehat{\boldsymbol{\beta}}^{ls}$ *has the minimum variance among all linear unbiased estimators of* $\boldsymbol{s}'\boldsymbol{X}\,\boldsymbol{\beta}$, *where* $\boldsymbol{s} \in \mathbb{R}^n$.

Least squares estimates are optimal among all linear unbiased estimates. That is the so-called Best Linear Unbiased Estimates (BLUE). The MSE for the least squares estimate is the minimum among all linear estimates and it is proportional to the trace of the covariance matrix of $\widehat{\boldsymbol{\beta}}^{ls}$, that is

$$\begin{aligned} \mathrm{MSE}(\widehat{\boldsymbol{\beta}}^{ls}) &= \mathsf{E}\left((\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}^{ls})'(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}^{ls})\right) \\ &= \mathsf{trace}(\boldsymbol{X}'\boldsymbol{X})^{-1}\,\sigma_\epsilon^2. \end{aligned} \tag{2.10}$$

The MSE of prediction for the LS, denoted by $\mathrm{MSEP}(\widehat{\boldsymbol{y}}^{ls})$, evaluated at the training data points is equal to

$$\begin{aligned} \mathrm{MSEP}(\widehat{\boldsymbol{y}}^{ls}) &= \mathsf{E}\left((\boldsymbol{y} - \widehat{\boldsymbol{y}}^{ls})'(\boldsymbol{y} - \widehat{\boldsymbol{y}}^{ls})\right) \\ &= \mathsf{trace}\left(\boldsymbol{X}\,(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\right)\sigma_\epsilon^2 \\ &= p\,\sigma_\epsilon^2, \end{aligned} \tag{2.11}$$

with $\mathcal{P}_X = \boldsymbol{X}\,(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'$ being the projection or hat matrix. In line with general expression for the Expected Prediction Error, for the linear regression model this equals to

$$\text{EPE}\left(\widehat{f}(\boldsymbol{x}_0)\right) = \sigma_\epsilon^2 + \mathsf{bias}^2\left(\widehat{f}(\boldsymbol{x}_0)\right) + \mathsf{var}\left(\widehat{f}(\boldsymbol{x}_0)\right), \tag{2.12}$$

with

$$\mathsf{E}_n\left(\mathsf{var}\left(\widehat{f}(\boldsymbol{x}_0)\right)\right) = \frac{p}{n}\,\sigma_\epsilon^2. \tag{2.13}$$

Expression (2.13) is simply the average of $\mathsf{var}\left(\widehat{f}(\boldsymbol{x}_0)\right)$ over the training data. It clarifies together with expression (2.11) that larger and complex models (large p) result in high prediction variance and therefore may not always guarantee good prediction performance for the constructed models.

Linear regression models and LS have been extensively studied and used in statistics (see Seber G.A.F., 2003). Linear regression models have been shown to be flexible and applicable in a wide range of real world experiments. Despite the relatively strong parametric assumption for least squares regression model the use of variables transformation may render non linear models to linear and may additionally stabilize the variance of the error term in case of heteroscedasticity; for a nice overview one can see Chatterjee and Hadi (1988) as well as Draper and Smith (1998).

Despite the flexibility of linear regression and the nice properties of least squares, these are not always sufficient to treat a huge variety of existing problems. Why should one choose to quit the linear regression model for more complicated regression models? The basic reasons are summarized below:

1. Linear regression admits f to be globally linear and this is often too severe.

2. LS estimates make parametric assumptions on the distribution of the error term which are often violated, ie. independent gaussian random errors with constant variance.

3. For large samples $n \to \infty$ we have $\widehat{f}_n \to \mathsf{E}(Y|X)$. Yet, large samples is commonly not the case.

4. The assumption that the conditional probability $\mathsf{Prob}(Y|X)$ is related to X directly and only through the conditional mean is not true in many situations where X is related to the conditional probability above in more complicated ways.

5. Linear regression models and least squares may be highly influenced by a small perturbation of the data arising by outlying observations. The model can be then assumed to be linear but modelling the conditional mean is not the appropriate choice.

6. Finally, as p increases minimizing the criterion in (2.3) results in a large number of solutions and one should restrict the function space $\mathcal{F}$ to more eligible ones.

A wide field of regression methods arise as the solution to the above problems. We briefly recall some of these methods:

(a) *Generalized linear regression models* let for distributional assumptions other than the gaussian, and in certain cases model jointly the conditional mean together with the variance (see Fahrmeir and Tutz, 2001). They relate X linearly to $g(Y)$ where the link function g may be the log function, the logit function, etc. That is, the general linear model approximates f by the conditional expectation

$$\widehat{f}_n \approx g\left(\mathsf{E}_n(Y|X)\right).$$

(b) *Quantile regression* relates the conditional probability $\mathsf{Prob}(Y|X)$ to X through the quantiles, denoted by τ, rather the mean. It provides therefore a whole set of estimates for different quantiles (see Koenker and Bassett, 1978; Koenker, 2005). For a fixed quantile τ, the quantile regression is based on the minimization of

$$\widehat{f}_\tau(X) \,:\, \arg\min_{f_\tau} \sum_{i=1}^{n} \rho_\tau\left(y_i - f_\tau(\boldsymbol{x}_i)\right), \tag{2.14}$$

with $\rho_\tau(u) = (\tau - I(u \leq 0))\, u$, and I stands for the indicator function. For the special case of $\tau = 0.5$, quantile regression models the conditional median (med) instead of the conditional mean of LS, that is,

$$\widehat{f}_n \approx \mathsf{med}(Y|X),$$

and the resulting method is well suited for long tailed error distributions commonly caused by the presence of outliers in the data (see point 5, above).

(c) *Local regression and Kernel methods* provide regression estimates for the regression function f by conditioning on a neighborhood around the training data rather than the exact training points. They may be seen as fitting the weighted least squares criterion around the region x_0, for which

$$\widehat{f}_\theta(\boldsymbol{x}_0) \,:\, \arg\min_{f_\theta} \sum_{i=1}^{n} K_\xi(\boldsymbol{x}_0, \boldsymbol{x}_i)(y_i - f_\theta(\boldsymbol{x}_i))^2. \tag{2.15}$$

By $K_\xi(\boldsymbol{x}_0, \boldsymbol{x}_i)$ the Kernel function is denoted for its bandwidth ξ. The Kernel function assigns weights to $\boldsymbol{x}$ around the region $\boldsymbol{x}_0$. For $f_\theta(\boldsymbol{x}) = \theta_0 + \theta_1\, \boldsymbol{x}$ we recover the local linear regression model. For an overview see Hastie et al. (2004), chapter 6.

(d) *Basis functions regression* models construct linear models on the basis expansions $B_m(X)$. These are functions of the original data X, such as B-spline or wavelets (see Wahba, 1990; Chui, 1992). Such regression models can therefore take into account a wide range of complex models. The regression function solves

$$\widehat{f_\theta}(X) \,:\, \arg\min_{f_\theta} \sum_{i=1}^{n} \left(y_i - f_\theta(\boldsymbol{x}_i)\right)^2 \;\text{ for }\; f_\theta(\boldsymbol{x}) = \sum_{m=1}^{M} \theta_m \, B_m(\boldsymbol{x}), \quad (2.16)$$

with the M denoting the number of basis expansions.

(e) *Projection pursuit regression* models produce a sequence of smaller models which best approximate the function f by

$$\widehat{f}(x) \approx \sum_{m=1}^{M} g_m(\mathbf{u}_m' \boldsymbol{x}), \quad (2.17)$$

for a univariate smooth function g, and a unit length vector $\mathbf{u}_m$ (often called *loading* vector). The former is usually a linear smoother. The summation ranges from 1 to M, where the latter indicates the number of local functions g_m to which function f is decomposed. For more details on projection regression methods and linear smoothers one can see Friedman and Stuetzle (1981) and Buja et al. (1989).

2.2 Multicollinear and high dimensional data

In the previous section we focused attention on regression methods and underlined the departure from the linear model due to non linearities, small sample sizes, and restrictions in the estimation functions. These led to extend the linear regression model to more sophisticated regression techniques such as GAM. Here we focus on two special type of data sets: high dimensional and multicollinear data sets.

Multicollinearity

Multicollinearity refers to data sets with strong linear dependencies among the predictors which have a negative impact on the stability of regression estimates based on the linear regression model. In such cases the data are *ill-conditioned* and the variance of the regression estimates $\widehat{\boldsymbol{\beta}}$ highly inflate. Consequently, small changes in the input result to large changes in $\widehat{\boldsymbol{\beta}}$. Multicollinearity is commonly detected by the condition index of the predictor's variance-covariance matrix $\boldsymbol{\Sigma}_X$. This corresponds to the ratio

$$\mathsf{ci}_j = \frac{\lambda_1}{\lambda_j}, \quad (2.18)$$

with λ_j being the jth largest eigenvalue resulting from the eigen decomposition of matrix $\mathbf{\Sigma}_X$. Equivalently eigenvalues may be replaced by the singular values of the Singular Value Decomposition of the data matrix $\boldsymbol{X}$; see in the Mathematical Appendix I for proper definitions. The study of the condition index ci_j reveals linear dependencies among the variables $\boldsymbol{x}_j$. Large values of ci_j indicate collinearity. Yet, it is not easy to quantify how large is large and several proposals exist; see Belsley (1991) for an extensive study.

High dimensional data

High dimensional data arise in modern statistical practice mainly due to applications where the data are discretized functions rather than vectors. Modern data instrumentation, such as microarray and Near Infra-Red devices, provide data sets with a large number of recorded variables for relatively moderate sample sizes. In such cases dependence among the predictors is not a deficiency, but arises due to the functional and structural nature of the data. The data are ill-conditioned and often rank deficient. Modern statistical areas such as dimension reduction methods (see for example Helland and Almøy (1994), and Carreira (1997) for an overview), and Functional Data Analysis (see Ramsay and Silverman, 2005) treat these type of data. For a complete list of the statistical methods for high dimensional data see Hastie et al. (2004).

2.3 Shrinkage and regularized regression

Regularization methods have been mainly used in numerical algebra in order to find meaningful approximate solutions of ill-conditioned or singular linear systems, where parameters are ill-determined by least squares methods. Their development has been mainly motivated by the case where the number of parameters in a linear system is larger than the number of available equations. For a global overview on regularization methods in linear algebra one can see Neumaier (1998) and the references therein. In statistics, regularization has been for a longtime almost synonymous to Ridge Regression which corresponds to the Tikhonov regularization for mathematicians (see Tikhonov, 1963). While regularization is closely related to numerical algebra, the term *shrinkage* is used by statisticians to denote techniques that improve an estimator by shrinking it, and regularize ill-posed problems. In this sense regularization and shrinkage have the same objectives. An estimator is shrunk when a regularization technique has been applied in the estimation procedure. In statistics, the data explosion due to modern data instrumentation as well as the constantly increasing interest in functional data analysis has made regularization techniques a fundamental tool in statistical data analysis and modelling. The interpretation of regularization or shrinkage regression methods through a Bayesian approach (see Hastie et al., 2004, Chapter 3) has reinforced the statistical framework of such methods.

2.3.1 Regularization: definition and strategies

The following definition of regularization for prediction problems (including the general regression problem) is taken from Breiman (1996).

Definition 2.1 *Let $\mathcal{U}$ be a large class of functions $f(X)$. A regularization procedure consists in finding a sequence of subspaces $\mathcal{U}_s \subset \mathcal{U}$ indexed by a real parameter $s \geq 0$ such that*

$$s \leq s' \Rightarrow \mathcal{U}_s \subset \mathcal{U}_{s'}.$$

Let $\widehat{f}(X,s)$ be the function that minimizes $\mathsf{E}_n\left[\mathcal{L}\left(Y, f(X,s)\right)\right]$. Then $\widehat{f}(X,s)$ is called the sequence of regularized predictors.

Taking by default the squared loss function the minimization problem in the definition above is commonly solved by the least squares. Recall that this corresponds to

$$\widehat{f}_s \,:\, \arg\min_{f\,\in\,\mathcal{F}} \,\|Y - f\left(X,s\right)\|_2\,.$$

For ill-posed and singular systems regularization methods approximate the solution by following mainly three regularization strategies:

- *Penalization (Tikhonov) :* In this case for a penalization parameter δ and $\mathcal{H} \subseteq \mathcal{F}$ the minimization problem is modified to

$$\widehat{f}_\delta \,:\, \arg\min_{f\,\in\,\mathcal{H}} \left\{\,\|Y - f\left(X\right)\|_2 \,+ \delta\,\|f\|_2\,\right\}.$$

- *Explicit subspace methods :* In this case a closed sequence of subsets $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \mathcal{H}_3 \subset \ldots \mathcal{H}_p = \mathcal{H}$ is derived. The vectors $\mathbf{h}_l$ with $l = 1, \ldots, \ell$ satisfy $\mathcal{H}_\ell = \mathsf{span}\left\{\,\mathbf{h}_1, \ldots, \mathbf{h}_\ell\right\}$. The minimization problem now corresponds to

$$\widehat{f}_\ell \,:\, \arg\min_{f\,\in\,\mathcal{H}_\ell} \,\|Y - f\left(X\right)\|_2\,.$$

- *Sequential or iterative methods :* This is the case of approximating the solution to the problem above iteratively inside nested subspaces which are Krylov subspaces (see Mathematical Appendix I) and are denoted by $\mathcal{K}_m(\,\mathsf{b},\,\mathsf{A})$. The solution to the problem now has the form

$$\widehat{f}_m \,:\, \arg\min_{f\,\in\,\mathcal{K}_m(\,\mathsf{b},\,\mathsf{A})} \,\|Y - f\left(X\right)\|_2\,.$$

Note that for Tikhonov regularization $s = \delta$ and $\widehat{f}_\delta \overset{\delta\to 0}{\longrightarrow} f$, for explicit subspace regularization $s = \ell$ and $\widehat{f}_\ell \overset{\ell\to p}{\longrightarrow} f$, while finally for iterative subspace regularization $s = m$ and $\widehat{f}_m \overset{m\to p}{\longrightarrow} f$.

2.3.2 Regularized regression

In this paragraph the goal is to illustrate how regularization techniques are necessary in regression analysis in order to solve problems arising by ill-conditioned or singular linear systems arising from highly collinear predictors and large-p data sets ($n \leq p$). Recall that the MSE of an estimate $\widehat{\theta}$ is given by

$$\begin{aligned} \text{MSE}(\widehat{\theta}) &= \mathsf{E}\left(\left(\theta - \widehat{\theta}\right)'\left(\theta - \widehat{\theta}\right)\right) \\ &= \left(\mathsf{E}\left[\widehat{\theta}\right] - \theta\right)^2 + \mathsf{E}\left(\widehat{\theta} - \mathsf{E}\left[\widehat{\theta}\right]\right)^2 \\ &= \mathsf{bias}^2(\widehat{\theta}) + \mathsf{var}(\widehat{\theta}). \end{aligned} \tag{2.19}$$

Expression (2.19) decomposes the MSE of the estimate θ in the sum of the squared bias plus the variance. A similar decomposition holds also for the expected prediction error as already shown. Collinear predictors and large-p data sets affect seriously the variance term in both estimation and prediction.

To see that we take as an example the case of linear regression, for which we know the least squares regression estimate $\widehat{\boldsymbol{\beta}}^{ls}$ to be the best linear unbiased estimate (BLUE) for $\boldsymbol{\beta}$. The variance term for the LS estimates, which corresponds to the its MSE since least squares estimates are unbiased, is equal to

$$\mathsf{var}(\widehat{\boldsymbol{\beta}}^{ls}) = (\boldsymbol{X}'\boldsymbol{X})^{-1}\,\sigma_\epsilon^2.$$

This expression, given that $\mathsf{trace}(\boldsymbol{X}'\boldsymbol{X}) = \mathsf{trace}(\boldsymbol{\Lambda})$ (see Mathematical Appendix I), can be written in terms of the eigenvalues λ_j according to

$$\mathsf{trace}\left(\ \mathsf{var}(\widehat{\boldsymbol{\beta}}^{ls})\ \right) = \sigma_\epsilon^2 \sum_{j=1}^{p} \frac{1}{\lambda_j}.$$

Finally, we know that the $\text{MSEP}(\widehat{\boldsymbol{y}}^{ls}) = p\,\sigma_\epsilon^2$. From the above it gets clear that in the presence on high collinearity and large-p data sets, least squares estimates inflate. They are still unbiased but they are highly variable. The latter results in large MSE and MSEP. The linear models then have poor prediction performance and uncertainty in their final estimates.

R. Sundberg in a technical report in the University of Stockholm stated that

> *"When, in a multiple regression, regressors are near-collinear, so called regularized or shrinkage regression methods can be highly preferable to ordinary least squares, by trading bias for variance."*

Therefore regularization is strongly related to the bias-variance tradeoff. One lets for some bias in order to decrease variance, and to achieve stable estimates and better predictions. A question which naturally arises is which regression methods shrink and how we control the amount of shrinkage. Below we briefly list some

basic shrinkage regression methods and associate these methods to the regularization strategies of the previous paragraph. The *Ridge Regression* and the *Lasso* are both regression methods which penalize the regression estimates using an appropriate norm (L_2 for Ridge, L_1 for Lasso). The extent of the penalty is controlled by the parameter δ. These two regression methods correspond to Tikhonov-type regularization. *Principal Components* and *Partial Least Squares* regression follow the second and the third regularization strategy with regularization parameters ℓ and m, respectively.

2.3.3 Shrinkage estimators

A shrinkage estimator for the regression coefficient $\boldsymbol{\beta}$ is given by

$$\widehat{\boldsymbol{\beta}}_{shrink} = \sum_{j=1}^{p} f(\lambda_j)\, \boldsymbol{\alpha}_j \,,$$

where $\boldsymbol{\alpha}_j = {\lambda_j}^{-1}\, \mathsf{w}_j\, \mathsf{w}_j'\, \mathsf{b}$, for $\mathsf{b} = (\boldsymbol{X}'\boldsymbol{y})$, and w_j are the eigenvector corresponding to λ_j, the j^{th} largest eigenvalue of the matrix $\mathsf{A} = \boldsymbol{X}'\boldsymbol{X}$ (see Mathematical Appendix I). The matrix $\boldsymbol{X}$ is assumed here to be of full column rank. The term $f(\lambda_j)$ is called the *shrinkage factor*. The importance of the latter in regression modelling is given by the fact that the extent of the shrinkage effect in the mean squared error of estimation and prediction is commonly expressed in terms of $f(\lambda_j)$ as given in the following proposition (see Krämer, 2006).

Proposition 2.1 *Denote the total MSE as TMSE. Then for the shrinkage estimate $\widehat{\boldsymbol{\beta}}_{shrink}$ and for the corresponding fitted values $\widehat{\boldsymbol{y}}_{shrink}$, for $j = 1, \ldots, p$, the following hold:*

$$\text{TMSE}\left(\widehat{\boldsymbol{\beta}}_{shrink}\right) = \sum_{j=1}^{p} \left\{f(\lambda_j) - 1\right\}^2 \left(\mathsf{w}_j'\,\boldsymbol{\beta}\right)^2 + \sigma^2\epsilon \sum_{j=1}^{p} \left\{\frac{f(\lambda_j)}{\lambda_j}\right\},$$

$$\text{TMSEP}\left(\widehat{\boldsymbol{y}}_{shrink}\right) = \sum_{j=1}^{p} \lambda_j\, \left\{f(\lambda_j) - 1\right\}^2 \left(\mathsf{w}_j'\,\boldsymbol{\beta}\right)^2 + \sigma_\epsilon^2 \sum_{j=1}^{p} \left\{f(\lambda_j)\right\}.$$

The variance of the j-th component decreases when $|f(\lambda_j)| < 1$, while it increases when $|f(\lambda_j)| > 1$. We have:

- The shrinkage factor for LS regression is given by $f(\lambda_j) = 1$. This is easily verified given the fact that the LS regression estimate can be expressed in terms of the eigenvalues and eigenvectors as

$$\widehat{\boldsymbol{\beta}}^{ls} = \sum_{j=1}^{p} \lambda_j^{-1}\, \mathsf{w}_j\, \mathsf{w}_j'\, \mathsf{b}. \tag{2.20}$$

- PC regression truncate the terms in (2.20) and retain ℓ out of p terms. PC regression estimate is then expressed as

$$\widehat{\boldsymbol{\beta}}_{\ell}^{pcr} = \sum_{j=1}^{\ell} \lambda_j^{-1} \, \mathsf{w}_j \, \mathsf{w}_j' \, \mathsf{b}, \tag{2.21}$$

  from which it directly follows that $f(\lambda_j)_{pcr} = 1$ if component j is in the model and $f(\lambda_j)_{pcr} = 0$ otherwise.

- Ridge regression is one of statisticians favorites shrinkage regression methods. The ridge regression coefficient vector is expressed according to

$$\widehat{\boldsymbol{\beta}}_{\delta}^{rr} = \sum_{j=1}^{p} \frac{1}{\lambda_j + \delta} \, \mathsf{w}_j \, \mathsf{w}_j' \, \mathsf{b}, \tag{2.22}$$

  from which it follows that $f(\lambda_j) = \lambda_j/(\lambda_j + \delta)$.

2.4 Model selection and assessment

Model selection and assessment are both important topics in constructing and evaluating regression models. The former concerns the choice of the best model among an ensemble of candidate regression models according to a selection criterion. The latter measures the prediction performance of the regression model after model selection has been done. Model selection is strong related to the bias-variance tradeoff already seen for linear regression in (2.12) and (2.13). Prediction performance is the main goal for the constructed regression models. Yet, minimizing the squarred loss given in expression (2.3) it will be shortly seen to lead to models with large estimation bias. Computer intensive methods as well as Information Criteria seek to estimate the optimism induced in estimating the prediction performance by fitting and validating the models on the same data sets.

Let $\mathsf{s} \in \mathcal{S}$ represent an metaparameter belonging to an ensemble $\mathcal{S}$ which indicates the complexity of a linear, generalized linear or a generalized additive model. In the simplest case, in linear regression, s may be a subset of $\{1, \ldots, p\}$ indicating the variables X_j with $j = 1, \ldots, p$ which are included in the regression model, and $\mathcal{S}$ is the ensemble of all subset models. The estimated regression function for a model including the subset s is the conditional expectation

$$\widehat{f}_{n,\mathsf{s}} = \mathsf{E}_n(Y \mid X_{j \in \mathsf{s}}). \tag{2.23}$$

Hence $\widehat{f}_{n,\mathsf{s}}$, for notational convenience simply $\widehat{f}_{\mathsf{s}}$, approximates the conditional expectation of Y for a model including the variables with index j in the subset s. Numerous approximations to f result for different choice of s. The above concept is directly extended to the framework of GAM by noting that s is now

associated to the number M of basis expansions $B_m(x)$, the neighborhood of the Kernel and its bandwidth λ, the number of knots in local polynomial fitting, or the number M of local functions g_m to which f is decomposed in projection pursuit regression.

For all these different cases model selection seeks to define the value s_0 for which the approximation $\widehat{f}_{\mathsf{s}_0}$ is the best one compared to all other approximations to the regression function resulting from a different choice for s. The parameter s defines the complexity of the constructed models. In linear regression for example complexity is expressed by the number of variables in the final model, while in smoothing splines it is expressed in terms of the number M of basis expansions $B_m(x)$. Model selection methods are essentially based on the minimization of the prediction loss criterion. That is, one seeks for

$$\widehat{f} \,:\, \arg\min_{\mathsf{s}} \mathsf{E}_{XY}\left(\,\mathcal{L}\left(Y, f_{\mathsf{s}}(X)\right)\right). \tag{2.24}$$

Generally speaking, increasing s leads to more complicated models. These are regression models of high complexity.

Indeed, expression (2.24) is nothing but the empirical risk introduced in (2.3). For the squared error loss expression (2.24) leads to the bias-variance decomposition given in (2.6). For the linear regression model the empirical risk is rewritten according to

$$\mathrm{EPE}(\widehat{f}(x_0)) = \sigma_\epsilon^2 + \mathsf{E}_n\left(f(x_0) - \mathsf{E}_n\left(\widehat{f}(x_0)\right)\right)^2 + \frac{p}{n}\,\sigma_\epsilon^2, \tag{2.25}$$

and model complexity is just a function of the total number of variables p. Retaining a large number of variables leads to a minimal loss on the training data (the data at hand) but the model performs poorly on new data set of observations. This is known as *overfitting*. Hence, (2.24) is rather *optimistic* in what concerns the true or generalized prediction performance of the constructed models.

The concept of overfitting can be also understood by the fact that the same data are used in order to fit and assess a model. In order to avoid overfitting one has to rely on a risk criterion other than the the empirical risk. One should indeed add to the empirical risk the *optimism* to get a reliable estimate for the true or Final Prediction Error. It can be shown that $\omega = (2/N)\sum_{i=1}^{N} \mathsf{cov}(\widehat{y}_i, y_i)$, which for a linear fit with p parameters or m basis expansions simplifies to $\sum_{i=1}^{N} \mathsf{cov}(\widehat{y}_i, y_i) = d\,\sigma_\epsilon^2$, where d stands for either p or m, and σ_ϵ^2 denotes the variance of the error term.

In many cases, for example when σ_ϵ^2 is hard to get or for non linear class of estimates, in order to avoid overfitting the final model is selected by minimizing with respect to the complexity parameter s the following argument

$$\mathsf{E}_N\left[\,\mathsf{E}_n\left(\,\mathcal{L}(\boldsymbol{y}^\star, \widehat{\boldsymbol{y}}^\star_{\mathsf{s}})\right)\right] \tag{2.26}$$

with $\boldsymbol{y}^\star$ denoting the vector of N new response values measured on $\boldsymbol{X}^\star$ (in the best case scenario these coincide with the training points $\boldsymbol{X}$), and $\widehat{\boldsymbol{y}}^\star_{\mathsf{s}}$ the fitted

value vector from the regression model of complexity s constucted on the training data $(\boldsymbol{X}, \boldsymbol{y})$. If new observations are not available resampling or data splitting methods are used. These are desribed in the next paragraph.

2.4.1 Cross validation and bootstrap

In many applications the final prediction error is commonly estimated using computer intensive methods such as the Cross validation and the Bootstrap.

The *cross validation* computes directly the *out-of-sample* prediction error by splitting data $\mathcal{D}$ in a training set $\mathcal{D}_{train}$ (model construction) and a test set $\mathcal{D}_{test} = \mathcal{D}^{\star}$ (model validation), where $\mathcal{D}_{train} \cap \mathcal{D}_{test} = \emptyset$ and $\mathcal{D}_{train} \cup \mathcal{D}_{test} = \mathcal{D}$. The cross validated MSEP is given by

$$\text{MSEP}^{\,\text{cv}} = \mathsf{E}_M \left[\mathsf{E}_m \left(\mathcal{L} \left(\boldsymbol{y}^{\star}, \widehat{\boldsymbol{y}}^{\star\,(-m)} \right) \right) \right], \tag{2.27}$$

where the superscript * indicates observations in $\mathcal{D}^{\star}$, $m = 1, \ldots, M$ is the part of the M groups of the data which are left out ($M = 1$: one-random-split cv, $M = n$: leave-one-out cv). By E_M we denote expectation over the M different splits, and by E_m expectation over the number of observations inside the m^{th} test set. Finally, the superscript $^{(-m)}$ indicates that the fitted values are given by models constructed on the data set excluding the m^{th} part.

The *bootstrap* estimates directly the *optimism* (ω) by generating B bootstrap samples with replacement from the initial data (they are denoted as $\mathcal{D}^{\star,b}$ with $b = 1, \ldots, B$). The bootstrap MSEP is given by

$$\text{MSEP}^{\text{boot}} = \mathsf{E}_n \left[\mathcal{L} \left(\boldsymbol{y}, \widehat{\boldsymbol{y}} \right) \right] + \widehat{\omega}. \tag{2.28}$$

The bootstrap estimate of the optimism $\widehat{\omega}$ is calculated according to

$$\widehat{\omega} = \mathsf{E}_n \, \mathsf{E}_B \left[\mathcal{L}(\boldsymbol{y}, \boldsymbol{X}\widehat{\boldsymbol{\beta}}^{\star\,b}) - \mathcal{L}(\boldsymbol{y}^{\star,b}, \boldsymbol{X}^{\star\,b}\widehat{\boldsymbol{\beta}}^{\star\,b}) \right], \tag{2.29}$$

where $\mathcal{L}(\boldsymbol{y}^{\star,b}, \boldsymbol{X}^{\star\,b}\widehat{\boldsymbol{\beta}}^{\star\,b})$ is the resampling error, $\mathcal{L}(\boldsymbol{y}, \boldsymbol{X}\widehat{\boldsymbol{\beta}}^{\star\,b})$ is the loss induced by testing predictions for models constructed on the bootstrap samples $\left((\boldsymbol{X}^{\star,b}, \boldsymbol{y}^{\star,b}) \rightarrow \widehat{\boldsymbol{\beta}}^{\star\,b} \right)$ on the original data $(\boldsymbol{X}, \boldsymbol{y})$, E_n denotes expectation over the n sample values, and finally E_B denotes expectation over bootstrap replicates.

2.4.2 Model selection in shrinkage regression. An example

Choosing the regularization parameter (generally denoted by s) in shrinkage regression methods is strongly related to model selection. Tuning the values of s (δ, ℓ, and m for each case) results in a different bias-variance tradeoff. Choosing s affects the model complexity, and models of high complexity will fit perfectly the training data but will not generalize their performance to new observations, coming from the same population. Hence, models of high complexity have low bias

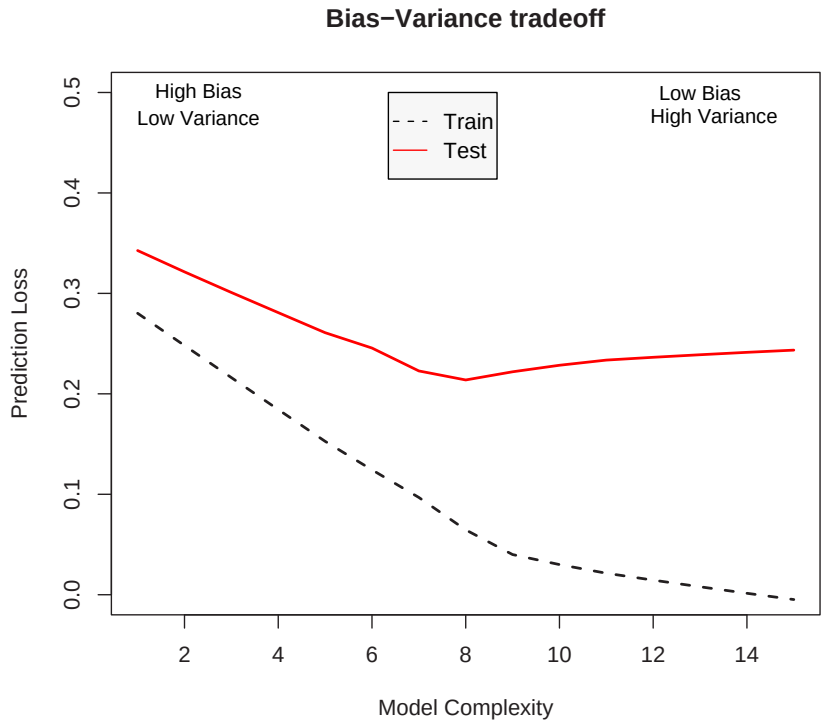


Figure 2.1: Test and training prediction error as a function of model complexity.

and large variance, while low complexity models have higher bias for relatively low variance.

The bias-variance tradeoff is illustrated in Figure 2.1 which is taken from a real experiment in chemometrics (Phetelamines data). Principal components regression has been used as regularization regression method and the prediction loss on the training (dashed black line) and the test (solid red line) set is illustrated for PC regression models including $\ell = 1$ to $\ell = 15$ principal components. The dashed black line illustrates the case of overfitting and the constant decrease of the prediction error for models of high complexity. The solid red line corresponds to the prediction loss for a new set of observations. It is decreasing up to $\ell = 8$ and then recovers for model complexity larger than $\ell = 8$ avoiding therefore the overfitting of the dashed line.

2.5 Outliers and robust regression

Data sets are often contain abnormal values which sometimes invalidate statistical procedures. Outliers are sample cases with large values on the response and/or the predictors. They often have large influence on the regression para-

meter estimates. They are commonly due to heavy tailed error distributions as well as mixtures of different error distributions or a deficient design matrix $\boldsymbol{X}$. Diagnostic tools and robust methods propose remedies, either to detect these values or to remove their influence in the applied statistical procedures.

As already mentioned LS criterion is sensitive to the presence of outliers. In the linear regression, unusual observations and departure from normality assumptions, inherent to LS, may invalidate the constructed regression models and weaken its prediction performance. The same also holds for the generalized additive models; outlying observations may lead to overfitting and poor prediction performance of the regression models.

In order to overcome such problems robust regression and diagnostic methods have been for a longtime developed. *Diagnostics* are measures computed from the data which detect influential samples; by using these tools one tries to detect the influential observations. The statistical analysis may then be applied to the non outlying samples. *Robust methods* seek to apply a statistical technique without being influenced by the outlying samples. Robust methods guarantee that the statistical analysis will not be influenced by outliers without previously detecting them.

2.5.1 Outlier detection

Outlier detection is applied either on multivariate data or in regression problems. *Multivariate outliers* are defined as the observations which stand far away from the bulk of the multivariate data. Therefore, in order to detect them one needs a reliable estimate of location and scatter. Then usually some type of multivariate distances are computed for each observation i and a theoretical critical limit c is computed which serves in order to separate the samples in good and bad ones. The latter are the outliers. Assuming that the multivariate data come from a symmetric joint probability distribution, a common choice for the distances is given by

$$d_i(\bar{\boldsymbol{x}}, \mathbf{S}) = \sqrt{(\boldsymbol{x}_i - \bar{\boldsymbol{x}})' \, \mathbf{S}^{-1} \, (\boldsymbol{x}_i - \bar{\boldsymbol{x}})}, \quad i = 1, \ldots, n. \tag{2.30}$$

These are Mahalanobis distances where $\boldsymbol{x}_i'$ is the ith row of $\boldsymbol{X}$, and $\bar{\boldsymbol{x}}$ and $\mathbf{S}$ are the mean and variance-covariance matrix of the variables in $\boldsymbol{X}$, respectively. This measures the outlyingness for each observation i from the bulk of the data. The theoretical limit c is then approximated by the upper quantile of a χ^2 statistic.

Regression outliers represent samples with large scaled residuals in a regression problem as well as values with large leverages. The latter are detected by inspection of the regression projection or hat matrix $\mathcal{P}_{\boldsymbol{x}}$. The simplest way to detect outliers in the Y-space when LS are being used, is to calculate the *standardized residuals* for each observation. Standardized residuals are scaled regression residuals following a standard normal distribution. Therefore regression outliers is straightforward to detect. Outlying observations do not always have a bad influence in regression models. If we want to measure their influence on the fitted

model we can compute their Cook distance (see Cook, 1977), given by:

$$CD^2_{(i)} = \frac{(\widehat{\boldsymbol{\beta}} - \widehat{\boldsymbol{\beta}}_{(i)})' \boldsymbol{X}' \boldsymbol{X} (\widehat{\boldsymbol{\beta}} - \widehat{\boldsymbol{\beta}}_{(i)})}{p\, \widehat{\sigma}^2},$$

where $\widehat{\boldsymbol{\beta}}_{(i)}$ is the LS estimate of $\boldsymbol{\beta}$ based on the data without case i. Cook's distance is a measure of the distance between coefficients calculated with and without the ith observation. Cook suggested checking observations with $CD^2_{(i)} > F(.50, p, n-p)$, where F is a value from a F distribution. Asserting the influence of these observations to the fit we can calculate the $DFITS_{(i)}$. They are the difference between the scaled fitted values obtained from two models. The first model is built on the whole data set, the second model is built on all data points except observation i. Belsley et al. (1980) suggested that observations with $DFITS_{(i)} > 2\sqrt{\frac{p}{n}}$ should be considered as unusual.

The above measures hold in principal for linear regression models and for linear estimators of the general form $\widehat{\boldsymbol{y}} = \mathcal{P}_\star \boldsymbol{y}$ for which the projection matrix $\mathcal{P}_\star$ does not depend upon $\boldsymbol{y}$. Nevertheless, they have also been used in outlier detection for generalized linear models. Naturally the use of these tools for outlier detection for regression models other than the linear regression should be cautious. For more details and information on outlier detection together with real world examples one can see Gnanadesikan and Kettenring (1972); Cook (1977); Belsley et al. (1980); Devlin et al. (1981); Leroy and Rousseeuw (1987); Chatterjee and Hadi (1988); Hadi (1992b); Billor et al. (2000).

2.5.2 Robust regression

The prediction error in approximating a regression function f has been estimated in paragraph 2.2 by using the squared loss function

$$\mathcal{L}(Y, f(X)) = (Y - f(X))^2.$$

M-regression (see Huber, 1981) uses as $\mathcal{L}(Y, f(X))$ the Huber loss given by

$$\mathcal{L}(Y, f(X)) = \begin{cases} (Y - f(X))^2 & \text{if } |Y - f(X)| \leq c, \\ c\,(|Y - f(X)| - c/2) & \text{otherwise,} \end{cases} \tag{2.31}$$

for a positive value c. It reduces therefore the influence of outlying observations in the data sets providing estimates which are protected against outliers. M-estimates were introduced by Huber (1964), and they have been extended in location, dispersion and regression context.

In the linear regression context the M-estimator seeks for

$$\widehat{\boldsymbol{\beta}}^M = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \rho\left(\frac{y_i - \boldsymbol{x}_i' \boldsymbol{\beta}}{s}\right), \tag{2.32}$$

with ρ given by the loss function in (2.31). In the expression above sample expectation has been naturally replaced by the sum over the training samples i. Taking the partial first derivatives with respect to $\boldsymbol{\beta}$ and setting them equal to zero the system of the normal equations is written as

$$\sum_{i=1}^{n} \psi\left(\frac{y_i - \boldsymbol{x}_i'\boldsymbol{\beta}}{s}\right) \boldsymbol{x}_i = 0, \tag{2.33}$$

where $\psi(x) = x$ for $|x| \leq c$, and $\psi(x) = c \cdot \mathsf{sgn}(x)$ otherwise; by $\mathsf{sgn}(x)$ we denote the sign of the variable x. M-estimators are studentized by an appropriate scale statistic s in order to be scale invariant and to retain regression equivariance. M-estimates are essentially Maximum Likelihood estimators, this is verified by letting $\rho(z) = (z)^2$.

M-estimates have been extensively used in all fields of statistical analysis. They are known to be robust to outliers and highly efficient. M-estimates of scale, location and regression are presented and analyzed in Huber (1981, 1987) and Gnanadesikan and Kettenring (1972), among others. M-estimates bound influence in the Y-direction, but are not robust on outliers arising in the predictors space.

A special case of M-estimates are the L_1-estimates given by setting $\rho(.) = |.|$ which in the regression framework transforms to

$$\sum_{i=1}^{n} \rho\left(y_i - \boldsymbol{x}_i'\boldsymbol{\beta}\right) = \sum_{i=1}^{n} |y_i - \boldsymbol{x}_i^T\boldsymbol{\beta}|. \tag{2.34}$$

Minimizing expression (2.34) with respect to $\boldsymbol{\beta}$ results in the *Least Absolute Deviation* (LAD) regression (see Dodge, 1987, 1992, 1997). Similarly to the M-regression a scale parameter s is taken into account and it is commonly chosen to be either the Interquartile Range (IQR) or the Median Absolute Deviation (mad). Recall that $IQR = Q_3 - Q_1$ with Q_t denoting the t^{th} quantile, and

$$\mathsf{mad}(Z) = \mathsf{med}(|Z - \mathsf{med}(Z)|),$$

where $\mathsf{med}(Z)$ is the median of the random variable Z. LAD regression is highly robust respecting outliers on the response but it is sensitive to outliers on the predictor's space; just like M-regression. The LAD regression has been extensively studied in Dodge and Jureckova (1987); Dodge (1992); Birkes and Dodge (1993); Cade and Richards (1996); Dodge (1997). The LAD objective function has been also used in the framework of robust wavelet denoising and robust shrinkage regression and variable selection via the Lasso in Sardy (1998) and Wang et al. (2006), respectively.

The *Generalized M-estimates* (GM) bound the influence of outliers arising in both Y and X-direction. They do so by weighting the M-estimate by $d_i = d(\boldsymbol{x}_i)$ with the latter being a function of the distance of observation i in the X-space.

The GM-estimate solve

$$\widehat{\boldsymbol{\beta}}^{GM} : \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \rho \left(\frac{y_i - \boldsymbol{x}_i' \boldsymbol{\beta}}{s\, d_i} \right), \tag{2.35}$$

and the system of the normal equations is now written as

$$\sum_{i=1}^{n} d_i\, \psi \left(\frac{y_i - \boldsymbol{x}_i' \boldsymbol{\beta}}{s\, d_i} \right) \boldsymbol{x}_i = 0. \tag{2.36}$$

The solution to (2.36) provides estimates robust to both X and Y direction. The influence in the X-direction has been bounded by the use of the weights d_i. Various choices for the functions ρ and ψ do also exist. For an overview see Hampel et al. (1986).

The optimization problems for M- and GM-estimates are usually solved by means of Iteratively Reweighted Least Squares (IRLS) methods. The squared error criterion is iteratively run in order to obtain for iteration k the solution to the LS problem

$$\sum_{i=1}^{n} w \left(r_i^{(k-1)} \right) r_i^2,$$

with w a weight function of the residuals r_i of the previous iteration. Cut-off values as well as more elaborate functions are used in order to downweight the outlying observations through the function w, see Hampel et al. (1986).

Robust methods are characterized and are evaluated mainly by their *efficiency* and their *breakdown point*. The former indicates the loss induced by using a robust procedure when the data do not contain outliers. The latter corresponds to the minimum fraction of contaminated data for which the statistical procedure is invalidated. High breakdown estimates are usually not efficient, while efficient estimates do not yield high breakdown points.

Chapter 3

Overview on PLS regression

3.1 PLS methods and their applications

Partial Least Squares methods relate linearly two data matrices, often called *blocks*, by means of an underlying *latent variables model* generally given as

$$\boldsymbol{Y} = \boldsymbol{T}\,\boldsymbol{Q} + \boldsymbol{E},$$

with $\boldsymbol{T}_{(n\times k)}$ denoting the latent variables matrix (or score matrix) and $\boldsymbol{E}$ denoting the residual matrix. In order to set the general frame we give below the structure of the data with the X-block and the Y-block in matrix form:

$$\boldsymbol{X}_{(n\times p)} = \begin{pmatrix} x_{11} & x_{12} & \dots & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & \dots & x_{2p} \\ \vdots & \dots & \dots & \dots & \vdots \\ x_{n1} & x_{n2} & \dots & \dots & x_{np} \end{pmatrix}, \quad \boldsymbol{Y}_{(n\times q)} = \begin{pmatrix} y_{11} & \dots & y_{1q} \\ y_{21} & \dots & y_{2q} \\ \vdots & \dots & \vdots \\ y_{n1} & \dots & y_{nq} \end{pmatrix}.$$

In line with common factor analysis techniques PLS methods extract latent variables $\boldsymbol{T}_{(n\times k)} = (\boldsymbol{t}_1, \dots, \boldsymbol{t}_k)$. In PLS methods these take into account both $\boldsymbol{X}$ and $\boldsymbol{Y}$. PLS methods extract latent variables from directions which maximize the empirical covariance between latent or score vectors on the $\boldsymbol{X}$ and the $\boldsymbol{Y}$ block. This maximization distinguishes PLS methods from other multivariate statistical techniques such as the Principal Components and the Canonical Correlation Analysis. Hence, the essential point in PLS is the inclusion in the maximization problem of both $\boldsymbol{X}$ and $\boldsymbol{Y}$. The maximization in PLS will be further explained and better understood in regression problems which are the topic of this thesis. For the moment one should retain that $\boldsymbol{t}_k = \boldsymbol{X}_{k-1}\boldsymbol{w}_k$ and $\boldsymbol{u}_k = \boldsymbol{Y}_{k-1}\boldsymbol{q}_k$ are the X-score and Y-score vectors, while $\boldsymbol{w}_k$ and $\boldsymbol{q}_k$ are commonly called the weight vectors on $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively. The subscript k should not create any confusion for the moment, and it can be safely discarded.

A design with the basic matrices and vectors implicated in PLS methods is given in Figure 3.1. From this figure one can recognize the score matrices

$\boldsymbol{T}_k$ and $\boldsymbol{U}_k = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k)$, the X-weights $\boldsymbol{w}$ and the Y-loadings $\boldsymbol{q}$. Figure 3.1 includes also the X-loadings $\boldsymbol{p}$ and the Y-weight $\boldsymbol{c}$ which will be seen shortly. All matrices and vectors in Figure 3.1 are placed in order to indicate their block and their dimension.

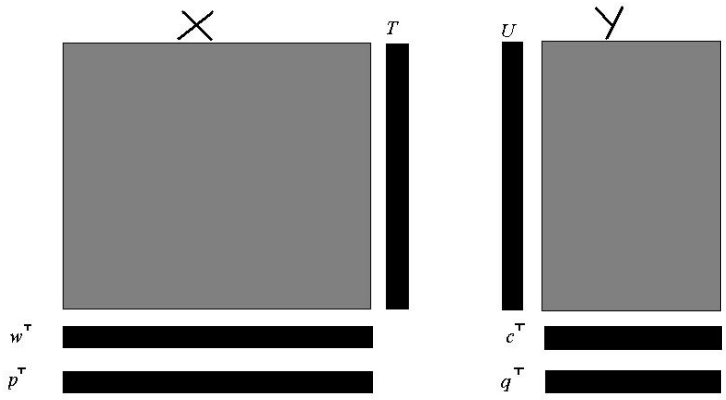


Figure 3.1: Representation of the ingredient matrices and vectors in the multivariate PLS.

As illustrated in Figure 3.1 the number of variables in $\boldsymbol{X}$ is commonly much larger than the number of $\boldsymbol{Y}$ columns. Indeed, in the simplest and commonly used case, the Y-block reduces to one single vector. In these case we use the term univariate PLS, in contrast to multivariate PLS for a matrix $\boldsymbol{Y}$.

PLS methods are applied in regression as well as in classification problems. PLS methods are very suitable to analyze highly collinear and *fat data*. It is indeed very common in PLS practise that the number of the recorded variables is equal or larger than the number of the observations. In these cases we use the term *fat data*. Therefore, PLS methods were naturally applied in order to solve such problems. To mention just a few applications, PLS methods have been applied in chemometrics and Near Infra-Red experiments, in microarray and gene experiments, in Electroencephalogram (EEG) experiments, as well as in environmetrics. In all cases above one thing is in common: a huge number of recorded variables is available for a relatively moderate sample size. This is principally due to modern instrumentation. Nevertheless, PLS methods have been also applied in more common cases with $n > p$. They have been extremely useful especially in cases where the data matrix $\boldsymbol{X}$ is ill-conditioned. That is, in cases where strong linear dependencies exist among the predictors.

3.2 PLS methods in regression

A very important field in statistical analysis is regression. The general regression problem has been already presented in Chapter 2. PLS methods have found a lot of applications and they have been significantly developed within the regression framework. Some of the most important dates in this development are given below:

1. Wold (1966, 1975) - **N**onlinear **I**terative **PA**rtial **L**east **S**quares (**NIPALS**),
2. Wold et al. (1983) - Univariate PLS regression (PLS1),
3. Höskuldsson (1988) - PLS regression optimization properties,
4. Helland (1988) - PLS regression and **Krylov spaces**,
5. Stone and Brooks (1990) - **Continuum Regression**,
6. Phatak (1993) - PLSR and **Lanczos** - **Conjugate Gradient** methods,
7. Rosipal and Trejo (2001) - **Kernel PLS** in Reproducing Kernel Hilbert spaces.
8. Bastien et al. (2005) - PLS generalized linear regression.

3.2.1 The NIPALS algorithm and the bilinear decomposition

The NIPALS algorithm was firstly present in Wold (1966) and further developed in Wold (1975). In the former the NIPALS algorithm was used in order to extract principal components. Then deflation techniques removed the extracted component and the algorithm continued for the extraction of the second principal component, and so forth. For more insight on the eigen pairs extraction and deflation methods, see Mathematical Appendix I . The NIPALS algorithm has been modified (see Wold, 1975) in order to take into account the response(s). It has then resulted in the PLS regression algorithm on orthogonal scores presented in Wold et al. (1983). The general NIPALS algorithm for PLS is given in Algorithm 3.1.

Algorithm 3.1 carries out the *bilinear decomposition* given by

$$\boldsymbol{X} = \boldsymbol{t}_1 \, \boldsymbol{p}_1' + \ldots + \boldsymbol{t}_k \, \boldsymbol{p}_k' + \mathbf{F}, \tag{3.2}$$

$$\boldsymbol{Y} = \boldsymbol{u}_1 \, \boldsymbol{q}_1' + \ldots + \boldsymbol{u}_k \, \boldsymbol{q}_k' + \mathbf{E}, \tag{3.3}$$

with $\mathbf{E}$ and $\mathbf{F}$ corresponding to residual terms. The decomposition in (3.2) and (3.3) is extensively analyzed in Martens and Naes (1989). Indeed the bilinear decomposition links matrix $\boldsymbol{Y}$ to matrix $\boldsymbol{X}$ using k latent vectors $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_k$. Expressions (3.2) and (3.3) are successively used together with scores ($\boldsymbol{t}_k$ and $\boldsymbol{u}_k$) and weight vectors ($\boldsymbol{w}_k$ and $\boldsymbol{q}_k$) as defined in Algorithm 3.1. The use of the symbol

Algorithm 3.1 NIPALS algorithm

Input: $(\boldsymbol{X}_0 \leftarrow \boldsymbol{X};\ \boldsymbol{Y}_0 \leftarrow \boldsymbol{Y})$;
For $k = 1, \ldots, p$, and $\boldsymbol{u}_k$ a column of $\boldsymbol{Y}$
Until convergence is met do:
Step 1. $\boldsymbol{w}_k \propto \boldsymbol{X}'_{k-1}\boldsymbol{u}_k \quad \in \quad \mathcal{C}(\boldsymbol{X})$;
Step 2. $\boldsymbol{t}_k \propto \boldsymbol{X}_{k-1}\boldsymbol{w}_k \quad \in \quad \mathcal{R}(\boldsymbol{X})$;
Step 3. $\boldsymbol{q}_k \propto \boldsymbol{Y}'_{k-1}\boldsymbol{t}_k \quad \in \quad \mathcal{C}(\boldsymbol{Y})$;
Step 4. $\boldsymbol{u}_k \propto \boldsymbol{Y}_{k-1}\boldsymbol{q}_k \quad \in \quad \mathcal{R}(\boldsymbol{Y})$;
If convergence is met
Compute loading vector $\boldsymbol{p}_k = \dfrac{\boldsymbol{X}'_{k-1}\boldsymbol{t}_k}{\boldsymbol{t}'_k\boldsymbol{t}_k}$;
Store: $\boldsymbol{T}[,k] \leftarrow \boldsymbol{t}_k; \quad \boldsymbol{U}[,k] \leftarrow \boldsymbol{u}_k; \quad \boldsymbol{P}[,k] \leftarrow \boldsymbol{p}_k; \quad \boldsymbol{Q}[,k] \leftarrow \boldsymbol{q}_k;$
Take as new data

$$\boldsymbol{X}_k = \boldsymbol{X}_{k-1} - \boldsymbol{t}_k\boldsymbol{p}'_k \quad \text{and} \quad \boldsymbol{Y}_k = \boldsymbol{Y}_{k-1} - \boldsymbol{t}_k\boldsymbol{q}'_k\,; \tag{3.1}$$

and go to *Step 1.* with $\boldsymbol{X} = \boldsymbol{X}_k$ and $\boldsymbol{Y} = \boldsymbol{Y}_k$.

$\propto$ emphasizes the fact that different normalizations can be used. Normalizing for example the extracted vectors, by using the normalizing constants $\boldsymbol{w}'_k\boldsymbol{w}_k$, $\boldsymbol{t}'_k\boldsymbol{t}_k$, etc, allows to form the whole procedure as a sequence of simple regressions. The derived vectors then correspond to the slopes in simple univariate regressions (see Bastien et al., 2005). This approach permits to effectively handle missing data in a simple and direct way (see Tenenhaus, 1998). The notation $\mathcal{C}$ and $\mathcal{R}$ indicate the column and the row space of matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$. Indeed the score vectors $\boldsymbol{t}_k$ and $\boldsymbol{u}_k$ lie in $\mathcal{R}$, while $\boldsymbol{w}_k$ and $\boldsymbol{q}_k$ lie in $\mathcal{C}$ of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively. This is also true for all $2 \leq k \leq p$ with data being deflated according to expression (3.1). The deflated data can be expressed in terms of the original data $\boldsymbol{X}_0$ and $\boldsymbol{Y}_0$ at any k, according to the equations below

$$\boldsymbol{X}_k = (\mathbf{I} - \mathcal{P}_{T_k})\,\boldsymbol{X}_0, \tag{3.4}$$

$$\boldsymbol{Y}_k = (\mathbf{I} - \mathcal{P}_{T_k})\,\boldsymbol{Y}_0, \tag{3.5}$$

with $\mathcal{P}_{T_k}$ denoting the projection or the hat score matrix $\boldsymbol{T}_k\,(\boldsymbol{T}'_k\boldsymbol{T}_k)^{-1}\boldsymbol{T}'_k$.

3.2.2 Maximization in PLSR and connection to PCR

Note that the decomposition in expression (3.2) provides a decomposition for $\boldsymbol{X}$ similar to the *eigen decomposition* given in Principal Component (PC) regression. The difference between them is better understood given the maximization criteria that the two methods solve.

In PLS regression the optimization criterion is given by

$$\arg \max_{\boldsymbol{w}_k, \boldsymbol{q}_k} \left\{ \mathsf{cov}(\boldsymbol{X}_{k-1}\boldsymbol{w}_k,\, \boldsymbol{Y}_{k-1}\boldsymbol{q}_k) \right\}, \tag{3.6}$$

subject to:

$$\boldsymbol{w}_k'\boldsymbol{w}_k = \boldsymbol{q}_k'\boldsymbol{q}_k = 1, \text{ and } \mathsf{cov}(\boldsymbol{X}_{k-1}\boldsymbol{w}_k, \boldsymbol{X}_{k'-1}\boldsymbol{w}_{k'}) = 0 \text{ for } k \neq k'.$$

The maximization criterion for PCR corresponds to:

$$\arg\max_{\boldsymbol{w}_k} \{\, \mathsf{var}(\boldsymbol{X}\boldsymbol{w}_k)\} \,, \tag{3.7}$$

subject to :

$$\boldsymbol{w}_k'\boldsymbol{w}_k = 1 \text{ and } \mathsf{cov}(\boldsymbol{X}\boldsymbol{w}_k, \boldsymbol{X}\boldsymbol{w}_{k'}) = 0 \text{ for } k \neq k'.$$

The PCR vector $\boldsymbol{w}_k$ in (3.7) corresponds to the eigenvector associated with the k^{th} largest eigenvalue of the matrix $\boldsymbol{X}'\boldsymbol{X}$ for $k = 1, \ldots, p$. That is, PCR vector $\boldsymbol{w}_k$ solves

$$\boldsymbol{X}'\boldsymbol{X}\,\boldsymbol{w}_k = \lambda_k\,\boldsymbol{w}_k,$$

with PC score given as $\boldsymbol{t}_k = \boldsymbol{X}\boldsymbol{w}_k$. The PLS weight vectors $\boldsymbol{w}_k$ and $\boldsymbol{q}_k$ in (3.6) correspond to the first left and right singular values from the SVD of the cross-covariance matrix of $\boldsymbol{X}$ and $\boldsymbol{Y}$. Hence, they solve

$$\boldsymbol{X}'_{k-1}\boldsymbol{Y}_{k-1}\boldsymbol{Y}'_{k-1}\boldsymbol{X}_{k-1}\,\boldsymbol{w}_k = \lambda\,\boldsymbol{w}_k,$$

with the PLS score vector given as $\boldsymbol{t}_k = \boldsymbol{X}_{k-1}\boldsymbol{w}_k$. This optimization is done for each dimension k and only the first singular values are retained, see Höskuldsson (1988). Note, finally, that the PLS regression makes a compromise between PC and LS regression. This follows from expression (3.6) and

$$\mathsf{cov}(X, Y) \propto \mathsf{var}(X)^{1/2}\ \mathsf{corr}(X, Y).$$

Indeed the variance maximization is related to PC, and the maximization of the correlation characterizes the LS regression.

3.2.3 Algorithmic implementations of PLS regression

Starting from the NIPALS algorithm various implementations of PLS regression has been proposed in the literature. More details are given in the following paragraphs. Here we just outline these algorithms:

- Orthogonal scores PLS regression [Wold et al. (1983, 1984)],
- Orthogonal loadings PLS regression [Martens (1985)],
- Helland Algorithm [Helland (1988)],
- SIMPLS Algorithm [de Jong (1993)],

3.3 Univariate PLS regression

The present thesis treats PLS regression problems on a single response vector. Therefore the univariate PLS regression algorithms are explicitly analyzed. Whenever it is practical the multivariate version of the quantities of interest are also given. Univariate PLS regression has gained the interest of statisticians and PLS regression practitioners mainly due to the following reasons: Univariate PLS regression is easier to understand, it should be well understood in order to go to the multivariate PLS regression setting. Univariate PLS regression is competitive to similar regression techniques outperforming the multivariate PLS. Often, univariate PLS regression of each column of $\boldsymbol{Y}$ is preferred to multivariate PLS (see Frank and Friedman, 1993; Breiman and Friedman, 1997). Finally, univariate PLS regression is easier in order to illustrate and understand shrinkage properties of PLS and its connections to the Conjugate Gradients and the Lanczos method.

3.3.1 Univariate PLS regression on orthogonal scores

The univariate Orthogonal Scores PLS regression algorithm (PLS1) given in Wold et al. (1983) is sketched in Algorithm 3.2. Algorithm 3.2 is equally expressed in statistical terms; this is done in Algorithm 3.3 later on this paragraph. We refer to Algorithm 3.3 simply as PLS regression algorithm on orthogonal scores.

Algorithm 3.2 PLS1 Algorithm

Input: $\boldsymbol{X}_0 = \boldsymbol{X}$, $\boldsymbol{y}_0 = \boldsymbol{y}$;
$k \leftarrow 1$;
while *a certain model selection criterion is not fullfiled* **do**

$$\boldsymbol{w}_k = \frac{\boldsymbol{X}_{k-1}^T \boldsymbol{y}_{k-1}}{\|\boldsymbol{X}_{k-1}^T \boldsymbol{y}_{k-1}\|}\,; \tag{3.8}$$

$$\boldsymbol{t}_k = \boldsymbol{X}_{k-1}\, \boldsymbol{w}_k; \tag{3.9}$$

$$\boldsymbol{X}_k = \boldsymbol{X}_{k-1} - \boldsymbol{t}_k\, \boldsymbol{p}_k^T, \text{ where } \boldsymbol{p}_k = \frac{\boldsymbol{X}_{k-1}^T \boldsymbol{t}_k}{\boldsymbol{t}_k^T \boldsymbol{t}_k}; \tag{3.10}$$

$$\boldsymbol{y}_k = \boldsymbol{y}_{k-1} - \boldsymbol{q}_k\, \boldsymbol{t}_k, \text{ where } \boldsymbol{q}_k = \frac{\boldsymbol{y}_{k-1}^T \boldsymbol{t}_k}{\boldsymbol{t}_k^T \boldsymbol{t}_k}; \tag{3.11}$$

$k \leftarrow k + 1$;
end while
Output: Give the final PLS1 regression model.

The univariate PLS regression algorithm (either in Algorithm 3.2 or Algorithm 3.3) extracts weights $\boldsymbol{w}_k$ and scores $\boldsymbol{t}_k$, and then computes the X-loading vectors

$\boldsymbol{p}_k \propto \boldsymbol{X}'_{k-1}\boldsymbol{t}_k$ and the Y-loadings $q_k \propto \boldsymbol{y}'\boldsymbol{t}_k$. These allow to perform the bilinear decomposition given by expressions (3.2) and (3.3). Note that q_k is the regression coefficient resulting from the regression of $\boldsymbol{t}_k$ on $\boldsymbol{y}$. We set $\widehat{\boldsymbol{q}}_k = (q_1, \ldots, q_k)$ for the loading vector after k iterations. This is the regression coefficient vector resulting from the final PLS regression of $\boldsymbol{T}_k$ on the response vector using least squares regression. The $k \times 1$ PLS regression coefficient vector $\widehat{\boldsymbol{q}}_k$, when it is transformed back in terms of the original variables, it *implies* the $p \times 1$ regression coefficient vector, denoted as $\widehat{\boldsymbol{\beta}}_k^{pls}$ and usually called the *implied regression coefficient*. It turns out that $\widehat{\boldsymbol{\beta}}_k^{pls}$ is a highly nonlinear function of the response $\boldsymbol{y}$. This can be seen by the following relation

$$\widehat{\boldsymbol{\beta}}_k^{pls} = \widetilde{\boldsymbol{W}}_k \widehat{\boldsymbol{q}}_k = \boldsymbol{W}_k (\boldsymbol{P}'_k \boldsymbol{W}_k)^{-1} \widehat{\boldsymbol{q}}_k \,, \tag{3.12}$$

for $\boldsymbol{W}_k = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k)$, $\boldsymbol{P}_k = (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_k)$, and $\widehat{\boldsymbol{q}}_k$ strongly depending on the response vector $\boldsymbol{y}$.

Algorithm 3.3 PLS regression on orthogonal scores

Input: $\boldsymbol{X}_0 \leftarrow \boldsymbol{X}$; $\boldsymbol{y}_0 \leftarrow \boldsymbol{y}$;
For $k = 1, ..., p$

1. Compute $\boldsymbol{w}_k$ according to:

$$\arg\max_{\boldsymbol{w}_k} \{ \operatorname{cov}(\boldsymbol{X}_{k-1}\boldsymbol{w}_k,\, \boldsymbol{y}_{k-1}) \} \quad \text{s.t.} \quad \boldsymbol{w}'_k \boldsymbol{w}_k = 1;$$

2. Derive component $\boldsymbol{t}_k = \boldsymbol{X}_{k-1}\boldsymbol{w}_k$ with $\boldsymbol{w}_k = (w_{1,k}, ..., w_{p,k})$;
3. Orthogonalize each predictor $\boldsymbol{x}_{j,k-1}$ with respect to $\boldsymbol{t}_k$;
*4. Orthogonalize the response $\boldsymbol{y}_{k-1}$ with respect to $\boldsymbol{t}_k$;

Output: Give the resulting sequence of the fitted vectors $\widehat{\boldsymbol{y}}_k = \boldsymbol{T}_k \widehat{\boldsymbol{q}}_k$, where $\boldsymbol{T}_k = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_k)$ and $\widehat{\boldsymbol{q}}_k = (\boldsymbol{T}'_k \boldsymbol{T}_k)^{-1} \boldsymbol{T}'_k \boldsymbol{y}$.

The complexity of the PLS regression model is controlled by the number of $\boldsymbol{t}_k$ in the final regression model. The choice for k affects the complexity of the final predictor and consequently its prediction performance. The model selection criterion on k is based on the prediction ability of the constructed model on a new set of observations. Finally, we note that the star exponent in step 4 is used to indicate that this step is optional since the projection of $\boldsymbol{y}_{k-1}$ on $\boldsymbol{t}_k$ is the same as the projection of $\boldsymbol{y}$ on $\boldsymbol{t}_k$, for a proof see Höskuldsson (1988).

Proposition 3.1 *For the PLS regression weight and score vectors as well as for the (deflated) data the following properties hold:*
1. $\boldsymbol{t}_k \perp \boldsymbol{t}_{k'} \quad for \quad k \neq k'$.
2. $\boldsymbol{w}_k \perp \boldsymbol{w}_{k'} \quad for \quad k \neq k'$.
3. $\boldsymbol{x}_{j,\ell} \perp \boldsymbol{t}_k$ $(\boldsymbol{t}'_k \boldsymbol{X}_\ell = 0)$ *for* $\ell > k$.
4. $\boldsymbol{y}'_{k-1}\boldsymbol{t}_k = \boldsymbol{y}'_0 \boldsymbol{t}_k$.

Proof: *Proofs and more properties can be found in Höskuldsson (1988) and Tenenhaus (1998).*

The properties in Proposition 3.1 confirm orthogonality of the score and the weight vectors $\boldsymbol{t}_k$ and $\boldsymbol{w}_k$, respectively. Property 3 confirms that residual data are orthogonal to previous extracted score vectors, while the last property essentially confirms that there is no need for data deflation in univariate PLS regression.

3.3.2 Predictions and implied regression coefficients

The PLS regression fitted values for a model base on k components is given by

$$\widehat{\boldsymbol{y}}_k = \boldsymbol{T}_k \widehat{\boldsymbol{q}}_k = \boldsymbol{X} \widetilde{\boldsymbol{W}} \widehat{\boldsymbol{q}}_k = \boldsymbol{X} \widehat{\boldsymbol{\beta}}_k^{pls}, \tag{3.13}$$

with the coefficient vector obtained from the expression (3.12) above.
Note that for multivariate PLS regression the implied coefficients transforms to

$$\widehat{\mathbf{B}}_k^{pls} = \widetilde{\boldsymbol{W}}_k \widehat{\boldsymbol{Q}}_k = \boldsymbol{W}_k (\boldsymbol{P}_k' \boldsymbol{W}_k)^{-1} \widehat{\boldsymbol{Q}}_k', \tag{3.14}$$

with the coefficient vector being now the coefficients matrix $\widehat{\mathbf{B}}_k^{pls}$. Predictions are then based on

$$\widehat{\boldsymbol{Y}}_k = \boldsymbol{X} \widehat{\mathbf{B}}_k^{pls}. \tag{3.15}$$

3.3.3 Univariate PLS regression on orthogonal loadings

The algorithm given in Martens (1985) is based on orthogonal loadings rather than orthogonal score vectors. It is reproduced here in Algorithm 3.4.

Orthogonality on the loadings is provided given equations (3.17) and (3.19) in Algorithm 3.4. In contrast to the NIPALS algorithm, the algorithm given by Martens (1985) uses directly multiple linear regression to get the loadings $\boldsymbol{q}_k$, and then deflates as in (3.19).
The star superscript in $\boldsymbol{t}_k^\star$, $\boldsymbol{y}_k^\star$, and in $\widehat{\boldsymbol{y}}_k^\star$ is here used only for illustrative purposes and comparisons to OS-PLSR. Indeed, Helland (1988) has proved the equivalence between the two PLS regression algorithms for prediction purposes, and has given the following proposition:

Proposition 3.2 *For the two PLS regression implementations in Algorithms 3.3 and 3.4, and for $k \leq p$ the following statements hold:*
1. $\boldsymbol{p}_k^\star = \boldsymbol{w}_k$.
2. $\mathsf{span}(\boldsymbol{t}_1, \ldots, \boldsymbol{t}_k) = \mathsf{span}(\boldsymbol{t}_1^\star, \ldots, \boldsymbol{t}_k^\star)$.
3. $\boldsymbol{y}_k^\star = \boldsymbol{y}_k$.
4. $\widehat{\boldsymbol{y}}_k^\star = \widehat{\boldsymbol{y}}_k$.

Proof: *For the proof see Helland (1988).*

Algorithm 3.4 PLS regression on orthogonal loadings

Input: $\boldsymbol{X}_0 \leftarrow \boldsymbol{X}$ and $\boldsymbol{y}_0 \leftarrow \boldsymbol{y}$;
For $k = 1, ..., p$

1. Compute the loading vector $\boldsymbol{p}_k^\star$ according to

$$\boldsymbol{p}_k = \boldsymbol{X}_{k-1}' \boldsymbol{y}_{k-1} \text{ and } \boldsymbol{p}_k^\star = \frac{\boldsymbol{p}_k}{\|\boldsymbol{p}_k\|_2}; \tag{3.16}$$

2. Compute the score vector

$$\boldsymbol{t}_k^\star = \boldsymbol{X}_{k-1} \boldsymbol{p}_k^\star / \boldsymbol{p}_k^{\star\prime} \boldsymbol{p}_k^\star; \tag{3.17}$$

   with $\boldsymbol{p}_k^\star = (p_{1,k}^\star, ..., p_{p,k}^\star)$ and store it in matrix $\boldsymbol{T}_k^\star$;
3. Recover the PLS regression coefficient according to

$$\boldsymbol{q}_k^\star = (\boldsymbol{T}_k^{\star,\prime} \boldsymbol{T}_k^\star)^{-1} \boldsymbol{T}_k^{\star\prime} \, \boldsymbol{y}_{k-1}; \tag{3.18}$$

4. Deflate data $\boldsymbol{X}_{k-1}$ and $\boldsymbol{y}_{k-1}$ according to

$$\boldsymbol{X}_k = \boldsymbol{X}_{k-1} - \boldsymbol{X}_{k-1} \boldsymbol{p}_k^\star \boldsymbol{p}_k^{\star\prime} \text{ and } \boldsymbol{y}_k^\star = \boldsymbol{y}_{k-1}^\star - \sum_{\ell=1}^{k} \boldsymbol{t}_\ell^\star \boldsymbol{q}_\ell^\star \, . \tag{3.19}$$

Output: Give the resulting sequence of the fitted vectors $\widehat{\boldsymbol{y}}_k^\star = \boldsymbol{T}_k^\star \widehat{\boldsymbol{q}}_k^\star$.

3.3.4 The Helland's approach

Helland (1988) provided a much more convenient way to express the PLS regression weight vector $\boldsymbol{w}_k$ given by the following recurrence:

$$\boldsymbol{w}_k \propto \begin{cases} \mathsf{b} & \text{for } k = 1, \\ \mathsf{b} - \mathsf{A}\,\boldsymbol{W}_{k-1}(\boldsymbol{W}'_{k-1}\,\mathsf{A}\,\boldsymbol{W}_{k-1})^{-1}\,\boldsymbol{W}'_{k-1}\,\mathsf{b}, & \text{for } k > 1, \end{cases} \tag{3.20}$$

where $\boldsymbol{W}_k = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k)$, $\mathsf{b} = \boldsymbol{X}'\boldsymbol{y}$ and $\mathsf{A} = \boldsymbol{X}'\boldsymbol{X}$. The PLS regression coefficient vector is then directly computed according to the following expression

$$\widehat{\boldsymbol{\beta}}_k^{pls} = \boldsymbol{W}_k(\boldsymbol{W}'_k\,\mathsf{A}\,\boldsymbol{W}_k)^{-1}\,\boldsymbol{W}'_k\,\mathsf{b}. \tag{3.21}$$

So the regression coefficient vectors given in (3.12) and (3.21) are equivalent. We will come back to this point in a while. For the moment we should notice the fact that the Helland algorithm for PLS regression establishes a direct way to obtain the PLS regression estimate $\widehat{\boldsymbol{\beta}}_k^{pls}$ from the basis vectors $\boldsymbol{w}_k$ given by a simple recurrence. While Algorithm 3.3 forms bases vectors for the components, the algorithm given by Helland constructs bases for the regression coefficient vector. Therefore, there is no need to deflate data nor to extract score vectors. The Helland's algorithm simplifies both interpretation and computation of the PLS regression vector. It is sketched in Algorithm 3.5.

Algorithm 3.5 PLS regression (Helland's implementation)

Input: $\boldsymbol{X}$ and $\boldsymbol{y}$;
$k \leftarrow 1$;
while *a certain model selection criterion is not fulfilled* **do**
  Step 1. Compute the loadings $\boldsymbol{w}_k$ according to expression (3.20);
  Step 3. Store loading vectors $\boldsymbol{w}_k$ into matrix $\boldsymbol{W}_k$.
  Step 4. Recover the implied regression coefficients as in (3.21);
  $k \leftarrow k + 1$;
end while
Output: Give the final regression model $\widehat{\boldsymbol{y}}_k^{pls} = \boldsymbol{X}\,\widehat{\boldsymbol{\beta}}_k^{pls}$.

3.4 Numerical aspects in PLS regression

3.4.1 PLS regression and Krylov spaces

We know come back to the point concerning the equivalence between expressions (3.12) and (3.21) for the PLS regression coefficient vector. Borrowing notation from Helland (1988), let

$$\widetilde{\mathbf{H}}_k = \boldsymbol{W}_k(\boldsymbol{W}'_k\,\mathsf{A}\,\boldsymbol{W}_k)^{-1}\,\boldsymbol{W}'_k.$$

Using the expression (3.21), the PLS regression vector is equal to $\widehat{\boldsymbol{\beta}}_k^{pls} = \widetilde{\mathbf{H}}_k\, \mathsf{b}$. According to Helland any matrix $\boldsymbol{V}_k$ of the form $\boldsymbol{W}_k\, \boldsymbol{C}$ can replace $\boldsymbol{W}_k$ in $\widetilde{\mathbf{H}}_k$ as long as its columns form a basis for the space spanned by the weight vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$. The following proposition then holds:

Proposition 3.3 *As long as $\boldsymbol{w}_k \neq 0$, the space spanned by the the weight vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$ is given by the sequence $(\,\mathsf{b},\, \mathsf{A}\,\mathsf{b}, \ldots,\, \mathsf{A}^{k-1}\,\mathsf{b})$, that is:*

$$\mathsf{span}(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k) = \mathcal{K}_k(\,\mathsf{b},\, \mathsf{A}).$$

The PLS regression coefficient vector $\widehat{\boldsymbol{\beta}}_k^{pls}$ can be therefore expressed in terms of the Krylov matrix K_k as

$$\widehat{\boldsymbol{\beta}}_k^{pls} = \mathsf{K}_k(\mathsf{K}_k'\, \mathsf{A}\, \mathsf{K}_k)^{-1}\, \mathsf{K}_k'\, \mathsf{b}. \tag{3.22}$$

Proof: *See Helland (1988) and Phatak and de Hoog (2001).*

The above proposition creates the link between PLS regression methods with approximations from Krylov spaces and related methods. For definitions and properties of Krylov spaces see the Mathematical Appendix I. The link between PLS regression with Krylov spaces and related methods are discussed in the following paragraphs. The first and direct result is the following remark:

Remark 3.1 *The PLS regression solution is a Krylov solution in the sense that*

$$\widehat{\boldsymbol{\beta}}_k^{pls} = \arg \min_{\boldsymbol{\beta}\, \in\, \mathcal{K}_k(\,\mathsf{b},\, \mathsf{A})} \|\boldsymbol{y} - \boldsymbol{X}\, \boldsymbol{\beta}\|, \quad \textit{for} \quad k \leq p. \tag{3.23}$$

The NIPALS implementation for PLS regression has made a certain choice for $\boldsymbol{V}_k$. Indeed the weight vectors $\boldsymbol{w}_k$ in the NIPALS algorithm (PLS regression on orthogonal scores) and the resulting basis $\boldsymbol{W}_k$ is formed by a Gram-Schmidt orthonormalization of the columns of K_k in the above expressions (see Phatak and de Hoog, 2001). The weight vectors $\boldsymbol{w}_k$ form an orthonormal basis of $\mathcal{K}_k(\,\mathsf{b},\, \mathsf{A})$ and therefore the matrix $\boldsymbol{W}_k'\, \mathsf{A}\, \boldsymbol{W}_k$ corresponds to an unreduced tridiagonal matrix. Let $\boldsymbol{H}^{(k)}$ denote the unreduced tridiagonal matrix with the superscript (k) emphasizing the link to $\mathcal{K}_k(\,\mathsf{b},\, \mathsf{A})$. The matrix $\boldsymbol{H}^{(k)}$ establishes the well known connection between PLS regression and the Lanczos method (see Manne, 1987; Phatak, 1993) briefly given in the following paragraph.

3.4.2 PLS regression and the Lanczos method

The Lanczos method and its connection to Krylov spaces is described in the Mathematical Appendix I. There it is shown that the Lanczos method is nothing more than the Raleigh Ritz approximation extracted from a Krylov subspace. Here we emphasize on the importance of the Lanczos method and its relation to PLS regression. This is established via the *ritz* pairs $(\theta^{(k)}, \boldsymbol{\phi}^{(k)})$ by the following two points:

1. The extremal eigenvalues of the unreduced tridiagonal matrix $\boldsymbol{H}^{(k)}$ converge to the extremal eigenvalues of A well before $k = p$.

2. The *ritz* values and the *ritz* vectors $(\theta^{(k)}, \boldsymbol{\phi}^{(k)})$ consist the best set of approximations to the eigen pairs of A derived from the subspace $\mathcal{K}_k(\mathsf{b}, \mathsf{A})$.

The points above are necessary to understand the importance of the Lanczos methods. They can be found in Parlett (1980). The goodness of the approximations to the eigen pairs of A, which are derived from the subspace $\mathcal{K}_k(\mathsf{b}, \mathsf{A})$, depends on the starting vector b and the spread of the eigenvalues of A (see Parlett, 1980). To see that note that for a starting vector b, orthogonal to some eigenvectors of A, these eigenvectors not only will not be well approximated, but they will not be detected at all. This provides therefore some limitations of the Lanczos method. However, Lanczos method and ritz approximations to eigenpairs via tridiagonal matrices benefit especially when matrices are large and sparse providing computationally efficient and fast approximations.

The ritz pairs allow to rewrite the PLS regression coefficient vector in terms of ritz expansions according to the expression

$$\widehat{\boldsymbol{\beta}}_k^{pls} = \sum_{j=1}^{k} \left(\theta_j^{(k)}\right)^{-1} \boldsymbol{\phi}_j^{(k)} \, \boldsymbol{\phi}_j^{(k)\prime} \, \mathsf{b} \quad \text{for } k \leq p, \tag{3.24}$$

which reveals the strong connection between PC and PLS regression coefficient vectors. The expression above together with the eigenvalue approximation given by the ritz pairs will be further discussed in the following paragraphs. In particular on the shrinkage aspects of PLS regression as well as in the paragraph concerning the theoretical aspects of PLS regression and its relation to PC regression.

3.4.3 PLS regression and orthogonal decomposition

A question that naturally arises is which is the connection between generalized inverses and PLS regression. In which sense PLS regression gives its own generalized inverse for the solution of ill conditioned linear systems (see, for example Wold et al., 1984). Calculating a generalized inverse implies an orthogonal decomposition of matrix $\boldsymbol{X}$, i.e.

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{R}\boldsymbol{W}', \tag{3.25}$$

for orthonormal matrices $\boldsymbol{U}$ and $\boldsymbol{W}$ of appropriate dimensions, and $\boldsymbol{R}$ an easily inverted square matrix. The singular value decomposition of the matrix $\boldsymbol{X}$ (see Mathematical Appendix I) respects the decomposition form in (3.25) with $\boldsymbol{R}$ being a diagonal matrix with elements the singular values s_j, while the vectors composeed in $\boldsymbol{U}$ and $\boldsymbol{W}$ correspond to the orthonormal set of singular vectors.

PLS regression is based on a different form for the decomposition in (3.25). PLS regression is based on the bidiagonalization of matrix $\boldsymbol{X}$ given in Golub and

Kahan (1965) and the matrix $\boldsymbol{R}$ is now a right bidiagonal matrix. Consequently, the matrix $(\boldsymbol{R}'\boldsymbol{R})$ is an easily inverted unreduced tridiagonal matrix. The latter serves as a basis from where eigenvalues are easily approximated from spaces of dimension smaller than p, and the connection to the Lanczos method of the previous paragraph is straightforward.

The NIPALS algorithm can be seen as making a certain choice for $\boldsymbol{U}$ and $\boldsymbol{W}$ in expression (3.25). The latter is constructed by puting inside the columns of $\boldsymbol{W}_k$ the weight vectors $\boldsymbol{w}_k$, with $k \leq p$ denoting the number of such vectors. The vectors $\boldsymbol{w}_k$ are computed together with the PLS regression scores $\boldsymbol{t}_k$ according to

$$\boldsymbol{w}_k \propto \boldsymbol{X}'_{k-1}\boldsymbol{y} \quad \text{and} \quad \boldsymbol{t}_k \propto \boldsymbol{X}_{k-1}\boldsymbol{w}_k, \tag{3.26}$$

while data are then deflated according to

$$\boldsymbol{X}_k = \boldsymbol{X}_{k-1} - \boldsymbol{t}_k\boldsymbol{p}'_k \quad \text{for} \quad \boldsymbol{p}_k \propto \boldsymbol{X}'_{k-1}\boldsymbol{t}_k, \tag{3.27}$$

with the latter denoting the X-loading vectors which are composed inside the columns of the loading matrix $\boldsymbol{P}_k = (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_k)$. This is just the NIPALS or the PLS regression algorithm on orthogonal scores.

3.4.4 PLS regression and Conjugate Gradients

Conjugate Gradient (CG) methods are very effective to solve systems of linear equations given by

$$\mathsf{A}\,\mathsf{z} = \mathsf{b}, \tag{3.28}$$

especially for large and sparse matrices A (see Golub and Van Loan, 1996). This is the case for the normal equations in modern applications. For an initial guess z_0, the solution to (3.28) is approximated as

$$\mathsf{z}_{k+1} = \mathsf{z}_k + \alpha_k\,\boldsymbol{d}_k,$$

for α_k minimizing $\eta(\mathsf{z}_{k+1}) = \eta(\mathsf{z}_k + \alpha_k\,\boldsymbol{d}_k)$ with η being the quadratic function $\eta(\mathsf{x}) = \frac{1}{2}\mathsf{x}'\,\mathsf{A}\,\mathsf{x} - \mathsf{x}'\,\mathsf{b}$, and where $\boldsymbol{d}_k$ denote direction vector where the solution is searched for during the k^{th} step. The minimization takes place over a sequence of spaces of increasing dimension; these are Krylov spaces.

For the CG method and its algorithm one can see the Mathematical Appendix I and the Bibliographic notes therein. For $\mathsf{A} = \boldsymbol{X}'\boldsymbol{X}$ and $\mathsf{b} = \boldsymbol{X}'\boldsymbol{y}$ the connection between PLS regression and CG is the following : given an initial guess $\mathsf{z}_0 = 0$, the CG iterate solutions z_k are the PLS regression estimates $\widehat{\boldsymbol{\beta}}_k^{pls}$ for $k \leq p$ (see Phatak, 1993). By letting $\boldsymbol{r}_k = \mathsf{b} - \mathsf{A}\,\mathsf{z}_k$ denote the residual vectors in the CG, the following remark concerns vectors $\boldsymbol{d}_k$ and $\boldsymbol{r}_k$.

Remark 3.2 *For the direction vectors $\boldsymbol{d}_k$ and the residual vectors $\boldsymbol{r}_k$ the following hold:*

1. $\mathsf{span}(\boldsymbol{r}_0, \ldots, \boldsymbol{r}_{k-1}) = \mathsf{span}(\boldsymbol{d}_0, \ldots, \boldsymbol{d}_{k-1}) = \mathcal{K}_k(\,\mathsf{b},\, \mathsf{A}\,)$.
2. $\boldsymbol{d}_i' \,\mathsf{A}\, \boldsymbol{d}_j = 0$ *for* $i \neq j$, *so* $\boldsymbol{d}_i \perp_{\mathsf{A}} \boldsymbol{d}_j$.
3. $\boldsymbol{r}_i' \boldsymbol{r}_j = 0$ *for* $i \neq j$, *so* $\boldsymbol{r}_i \perp \boldsymbol{r}_j$.

Proof: For the proof one can see Phatak (1993) and the original paper on CG by Hestens and Stiefel (1952).

3.5 PLS statistical models

PLS regression has received a lot of criticism from the statistical community concerning whether it is a statistical model or simply an algorithm. The latter is better understood by the fact that the bilinear decomposition is rather a decomposition than a concrete statistical model. Another point which has been difficult to interpret in statistical terms is data deflation and measuring association between deflated predictors and response. Yet, it turns out that PLS regression has some more serious drawbacks which mainly concern the shrinkage aspects of PLS compared to other shrinkage regression methods (see Frank and Friedman, 1993).

It is mainly I.S. Helland who has spent a lot of time and effort trying to explore the statistical aspects of PLS methods in regression and who has tried to give a population version for the PLS regression model (see Helland, 1990, 2001, 2004). He insisted on the fact that if one wants to get a statistical model behind PLS regression then one should search for it as n goes to infinity since all models are approximations to an ideal situation where we have infinite information. In such cases the main point is to observe that

$$\frac{1}{n}\,\mathsf{A} \to \boldsymbol{\Sigma}_{XX},\ \frac{1}{n}\,\mathsf{b} \to \boldsymbol{\Sigma}_{XY},\ \text{and then}\ \boldsymbol{\beta} = \boldsymbol{\Sigma}_{XX}^{-1}\boldsymbol{\Sigma}_{XY},$$

with $\boldsymbol{\Sigma}$ being the true covariance in the population, and $\boldsymbol{\beta}$ the true regression parameter. Then the population version of the space spanned by the PLS regression algorithm is also spanned by

$$\boldsymbol{\Sigma}_{XY}, \boldsymbol{\Sigma}_{XX}\boldsymbol{\Sigma}_{XY}, \ldots, \tag{3.29}$$

with $\boldsymbol{\Sigma}_{XY}$ reducing to $\boldsymbol{\sigma}$ for a single response, and the population matrix $\boldsymbol{\Sigma}_{XX}$ may be expressed in terms of the population eigenvalues and population eigenvectors denoted here by $\widetilde{\lambda}_j$ and $\widetilde{\mathsf{w}}_j$.

Definition 3.1 *(see Helland, 1990) The eigenvectors* $\widetilde{\mathsf{w}}_j$ *for which* $\widetilde{\mathsf{w}}_j'\boldsymbol{\sigma} \neq 0$ *are called relevant and the corresponding scores* $\boldsymbol{t}_j$ *are called relevant components.*

If by K we denote the maximum dimension of the sequence in (3.29) and S_K the space spanned by this sequence in the population, then the coefficient $\boldsymbol{\beta}$ belongs

to this space (it is a PLS regression coefficient), and $(\mathbf{\Sigma}\, S_K)$ spans the same space as S_K (that is S_K is approximatively invariant).

Remark finally that the LS coefficient belongs to $\mathcal{K}_{k^\star} = \mathcal{K}_{k^\star+1} = \ldots = \mathcal{K}_p$, for $k^\star$ being the lowest integer for which $\dim(\mathcal{K}_{k^\star}) = \dim(\mathcal{K}_{k^\star+1})$ (see also Mathematical Appendix I). PLS regression, similarly to PC regression, implies different constraints on the regression vector for $k < k^\star$. Indeed the two regression methods give very similar results, the former however will often require less k (see Helland, 1988). This may be understood by the fact that in certain cases better approximations to the eigen pairs are given by the ritz pairs approximated from Krylov spaces. This is however not always true.

3.5.1 PLS and biased regression

PLS regression similarly to PC regression imposes constraints on the regression coefficient vector. The PC regression vector, for example, given its expansion along the eigenvectors which is provided below

$$\widehat{\boldsymbol{\beta}}_\ell^{pcr} = \sum_{j=1}^{\ell} \lambda_j^{-1}\, \mathsf{w}_j\, \mathsf{w}_j'\, \mathsf{b}, \tag{3.30}$$

truncates the terms $j > \ell$ for which $\lambda_j \approx 0$. The PLS regression coefficient vector is similarly expressed as the expansion of ritz pairs given by the expression (3.24). The use of the superscript (k) emphasizes that the ritz values and vectors are extracted from a Krylov space of dimension k and approximate the eigenvalues of the tridiagonal matrix in the Lanczos procedure (see Mathematical Appendix I). Therefore, PLS imposes constraints on the final solution similar to PC regression.

Both regression methods belong to the class of biased regression methods. The PLS regression coefficient vector is a biased estimate of the true coefficient, that is

$$\mathsf{E}(\widehat{\boldsymbol{\beta}}_k^{pls}) \neq \boldsymbol{\beta},$$

for $k < k^\star$ with $k^\star$ defined in the previous paragraph. By introducing bias PLS regression reduces the variance of the regression estimates reaching in certain cases lower MSE than unbiased estimates (see paragraph 2.3). The shrinkage properties of biased estimates are normally in the center of interest for statisticians. For PLS regression, shrinkage properties are discussed in the following paragraph.

3.5.2 Shrinkage in PLS regression

Recall from paragraph 2.3 that a shrinkage estimator is given generally by

$$\widehat{\boldsymbol{\beta}}_{shrink} = \sum_{j=1}^{p} f(\lambda_j)\, \boldsymbol{\alpha}_j\,,$$

where $\boldsymbol{\alpha}_j = \lambda_j{}^{-1}\ \mathsf{w}_j\ \mathsf{w}_j'\,\mathsf{b}$, and that the term $f(\lambda_j)$ is the shrinkage factor. The extent of the shrinkage effect in the MSE of estimation and prediction has already been seen in Proposition 2.1 in paragraph 2.3. Recall also that the variance of the j-th component decreases when $f(\lambda_j) < 1$, while it increases when $f(\lambda_j) > 1$. The LS shrinkage factor has been shown to be equal to $f(\lambda_j)_{\text{ols}} = 1$ for all j, the PC shrinkage factor equals $f(\lambda_j)_{\text{pcr}} = 1$ if component j is in the model and $f(\lambda_j)_{\text{pcr}} = 0$ otherwise, while ridge regression gives $f(\lambda_j)_{rr} = \lambda_j/(\lambda_j + \delta)$.

PLS regression yield shrinkage estimates (see de Jong, 1995; Goutis, 1996). Yet, their properties have been shown to be rather peculiar (see Frank and Friedman, 1993; Butler and Denham, 2000). For an overview on PLS regression shrinkage properties one can see also Krämer (2006). Phatak and de Hoog (2001) give the shrinkage factor for PLS regression as follows

$$f_k(\lambda_j) = 1 - \frac{\chi_k(\lambda_j)}{\chi_{k,0}}, \tag{3.31}$$

where $\chi_{k,0} = (-1)^k \det(\boldsymbol{H}^{(k)})$ and $\chi_k(\lambda_j) = \det(\lambda_j\, \boldsymbol{I}_k - \boldsymbol{H}^{(k)})$. The last expressions stand for the characteristic polynomial for the tridiagonal matrices $\boldsymbol{H}^{(k)}$ from where eigenvalues and eigenvectors of A are approximated by the ritz values and vectors. Their connection to PLS regression and their relation to Krylov spaces and the Lanczos method have already been discussed (see paragraphs 3.4.1 and 3.4.2). It is also possible to express the PLS shrinkage in terms of Conjugate Gradients direction vectors. That is, PLS regression shrinks towards the smallest conjugate gradient directions $\boldsymbol{d}_j$ (see paragraph 3.4.4).

A serious drawback in PLS regression modelling is the fact that $f^k(\lambda_j)$ takes values larger than 1 expanding therefore the regression coefficient vector. This is in full contrast to the shrinkage principle that $f(\lambda_j) < 1$. The latter is due to the polynomial expression in (3.31). The first to note the fluctuating behavior for the PLS regression shrinkage factor were Frank and Friedman (1993) and Butler and Denham (2000). Frank and Friedman (1993) proposed to truncate the shrinkage factors of the PLS estimator by 1 when PLS estimator expands. Nevertheless, the PLS regression estimate is a nonlinear function of $\boldsymbol{y}$, and the PLS shrinkage factors are stochastic. This is in contrast with most shrinkage methods mentioned. Krämer (2006) gives an extensive simulation study on the effect of bounding the PLS shrinkage factors on the MSE of the regression estimator.

3.6 Model selection and assessment

Model selection and assessment has been already discussed for a general class of regression methods in Chapter 2. Here the general principles are specified for PLS regression and some parts may seem redundant given the discussion in Chapter 2. PLS regression models are based on the number of components retained in the final PLS regression model. This corresponds to the dimension of the Krylov space, following the Helland approach. In both cases we use the subscript k

to indicate either the component's number in Orthogonal Scores PLS regression or the Krylov expansion for the Helland implementation of PLS regression. The parameter k corresponds to the regularization hyperparameter which controls the PLS regression model complexity and consequently the bias-variance tradeoff.

A serious drawback in PLS regression model selection is the fact that the regression vector is a nonlinear function of the response vector and therefore many model selection techniques can not be applied. An extensive study of different model selection techniques to select the number of components are compared and analyzed in Butler and Denham (2000). We consider below three such methods: the cross validation, the bootstrap, and linearization techniques which have been proposed by Phatak (1993).

3.6.1 Cross validation

Cross validation computes the *out-of-sample* prediction error by splitting data in a training set $\mathcal{D}_{train}$ (model construction) and a test set $\mathcal{D}_{test} = \mathcal{D}^{\star}$ (model validation), where $\mathcal{D}_{train} \cap \mathcal{D}_{test} = \emptyset$ and $\mathcal{D}_{train} \cup \mathcal{D}_{test} = \mathcal{D}$. The cross validated MSEP is given by

$$\text{MSEP}_{\text{k}}^{\text{cv}} = \mathsf{E}_M \left[\mathsf{E}_m \left(\mathcal{L} \left(\boldsymbol{y}^{\star}, \widehat{\boldsymbol{y}}_k^{\star(-m)} \right) \right) \right]. \tag{3.32}$$

In expression (3.32) we have used the following notation:

- $\widehat{\boldsymbol{y}}_k$ denotes the PLS fitted values for a model including k components,
- The superscript * indicates observations in $\mathcal{D}^{\star}$,
- $m = 1, \ldots, M$ denotes the group of the data which is left out ($M = 1$: one-random-split, $M = n$: leave-one-out),
- E_M denotes average over the M different splits,
- E_m denotes average over the number of observations inside the m^{th} test set,
- $^{(-m)}$ indicates that the fits are given by PLS regression models on the data set excluding the m^{th} part.

3.6.2 Bootstrap

The bootstrap estimates the *optimism* (ω) induced in the empirical risk, by generating B bootstrap samples with replacement from the initial data (they are denoted as $\mathcal{D}^{\star,b}$ with $b = 1, \ldots, B$). The bootstrap MSEP is given by

$$\text{MSEP}_{\text{k}}^{\text{boot}} = \mathsf{E}_n \left[\mathcal{L}(\boldsymbol{y}, \widehat{\boldsymbol{y}}_k) \right] + \widehat{\omega}. \tag{3.33}$$

The bootstrap estimate of the optimism $\widehat{\omega}$ is calculated according to

$$\widehat{\omega} = \mathsf{E}_n \, \mathsf{E}_B \left[\mathcal{L}(\boldsymbol{y}, \boldsymbol{X}\boldsymbol{\beta}_k^{\star b}) - \mathcal{L}(\boldsymbol{y}^{\star,b}, \boldsymbol{X}^{\star b}\boldsymbol{\beta}_k^{\star b}) \right], \tag{3.34}$$

with

- $\widehat{\boldsymbol{y}}_k$ denotes the PLS fitted values for a model including k components,
- $\mathcal{L}(\boldsymbol{y}^{\star,b},\, \boldsymbol{X}^{\star\, b}\boldsymbol{\beta}_k^{\star\, b})$ denoting resampling error,
- $\mathcal{L}(\boldsymbol{y},\, \boldsymbol{X}\boldsymbol{\beta}_k^{\star\, b})$ the loss induced by testing predictions for models constructed on the bootstrap samples $\left((\boldsymbol{X}^{\star,b}, \boldsymbol{y}^{\star,b}) \rightarrow \widehat{\boldsymbol{\beta}}^{\star\, b}\right)$ on the original data $(\boldsymbol{X}, \boldsymbol{y})$,
- E_n denotes the expectation over the n sample values,
- E_B denotes the expectation over the B bootstrap replicates.

3.6.3 Linearization techniques

In this approach the idea is to provide an expansion of $\widehat{\boldsymbol{\beta}}_k^{pls}$ over a point $\boldsymbol{y}_0$, as follows

$$\widehat{\boldsymbol{\beta}}_k^{pls}(\boldsymbol{y}) \approx \widehat{\boldsymbol{\beta}}_k^{pls}(\boldsymbol{y}_0) + \frac{\partial \widehat{\boldsymbol{\beta}}_k^{pls}}{\partial \boldsymbol{y}} \mid_{\boldsymbol{y}_0} (\boldsymbol{y} - \boldsymbol{y}_0) \,.$$

Taking expectations it is then shown that

$$\mathsf{E}_n\left(\widehat{\boldsymbol{\beta}}_k^{pls}(\boldsymbol{y})\right) \approx \widehat{\boldsymbol{\beta}}_k^{pls}(\boldsymbol{y}_0) + \frac{\partial \widehat{\boldsymbol{\beta}}_k^{pls}}{\partial \boldsymbol{y}} \mid_{\boldsymbol{y}_0} (\,\mathsf{E}_n(\boldsymbol{y}) - \boldsymbol{y}_0) \,,$$

which gives

$$\mathrm{MSEP}_{\mathrm{k}}^{\mathrm{lin}} \approx \left(1 + \frac{1}{n}\right)\sigma^2 + \frac{1}{n}\left(\left[\mathsf{E}\left(\widehat{\boldsymbol{\beta}}_k^{pls}(\boldsymbol{y})\right) - \boldsymbol{\beta}\right]' \boldsymbol{X}'\boldsymbol{X} \left[\mathsf{E}\left(\widehat{\boldsymbol{\beta}}_k^{pls}(\boldsymbol{y})\right) - \boldsymbol{\beta}\right]\right)$$

$$+\frac{\sigma^2}{n}\mathsf{trace}\left[\widehat{\boldsymbol{\beta}}_k^{*}(\boldsymbol{y})'\, \boldsymbol{X}'\boldsymbol{X}\, \widehat{\boldsymbol{\beta}}_k^{*}(\boldsymbol{y})\right],$$

where

$$\widehat{\boldsymbol{\beta}}_k^{*}(\boldsymbol{y}_0) = \frac{\partial \widehat{\boldsymbol{\beta}}_k^{pls}}{\partial \boldsymbol{y}} \mid_{\boldsymbol{y}_0}$$

Yet, the choice for $\boldsymbol{y}_0$ as well as of an estimate for σ^2 still remain open.

3.7 Illustrative examples

3.7.1 Example I: an orthogonal design

In cases where no dependence exists among the predictors and data are scaled if they are measured on different scales, the PLS regression reaches the least squares solution after one step ($k = 1$). This is given in the proposition below and it is illustrated by an example.

Proposition 3.4 PLS regression takes one step for $\boldsymbol{X}'\boldsymbol{X} \propto \boldsymbol{I}_p$.

Proof: *For a geometric proof see in the Mathematical Appendix II.*

PLS regression method is a generalization of Multiple Linear Regression (MLR). The latter is justified using a regression problem where no correlation between the predictors occurs. Table 3.7.1 gives an artificial data set with $\mathsf{corr}(\boldsymbol{x}_1, \boldsymbol{x}_2) = 0$.

Table 3.7.1: Example I: An orthogonal design

Y	X_1	X_2
18	-2	4
12	1	3
10	0	-6
16	-1	-5
11	2	3
9	0	-3
11	-1	6
8	1	-1
7	1	0
12	-1	-1

The regression analysis for the data in Table 3.7.1 lead to the following least squares estimates:

$$\widehat{\boldsymbol{y}}^{ls} = 11.40 - 1.8571\boldsymbol{x}_1 + 0.1408\boldsymbol{x}_2.$$

Using PLS regression, after centering and scaling the data, we reach exactly the same solution with just one component used, that is, by $\widehat{\boldsymbol{y}}_1^{pls} = \widehat{q}_1\, \boldsymbol{t}_1$. We finally get

$$\widehat{\boldsymbol{y}}_1^{pls} = \boldsymbol{T}_1\, \widehat{q}_1 = \boldsymbol{X}\boldsymbol{w}_0\widehat{q}_1 = \boldsymbol{X}\widehat{\boldsymbol{\beta}}_1^{pls}.$$

Thus for standardized data the implied regression coefficients are

$$\widehat{\boldsymbol{\beta}}_1^{pls} = (-0.68, 0.164\,),$$

which, transformed back, equal to $(11, 40, -1.8571, 0.1408)$. The latter are the least squares estimates in the MLR.

3.7.2 Example II: an example from chemometrics

The Biscuit or the Cookies data have a long history in the Near Infra-Red (NIR) experiments. This is a high dimensional data set for which dimension reduction methods such as PLS and PC regression are commonly used. The Cookies data have been already analyzed in Goutis (1996), Brown et al. (2001), and West (2003). Forty samples of cookies have been used in this experiment and the NIR

spectrum for each cookie has been recorded throughout the wavelengths range 1100 - 2498 nanometers (p = 700). Thirty two new samples have been then used to test the constructed models. For the biscuit data the response is a matrix containing fat, sucrose, dry flour, and water. The data sets are available through the web link *http://www.stat.tamu.edu/ mvannucci/webpages/codes.html*. For more details on the Biscuit data one can see Brown et al. (2001) and the references therein. The spectra for all training samples are given in Figure 3.2.

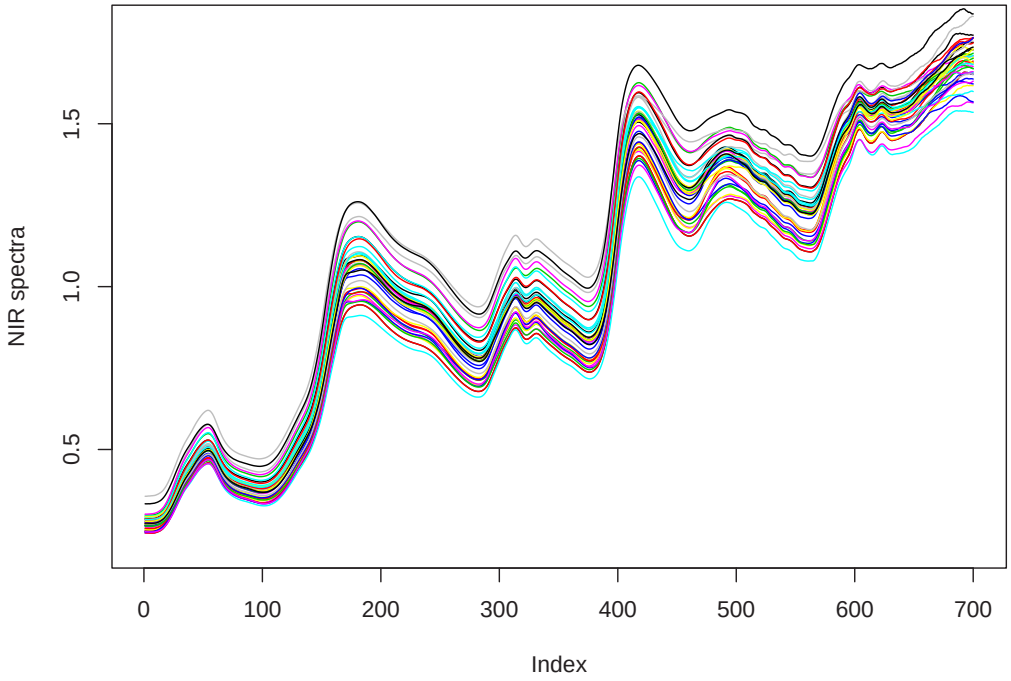


Figure 3.2: Biscuit data. NIR calibration spectra on 40 cookies samples measured throughout the wavelengths range 1100 - 2498 nanometers.

This data set may be used for multivariate PLS regression modelling with $\boldsymbol{Y}$ including 4 columns corresponding to Y_1: fat, Y_2: sucrose, Y_3: flour, and Y_4: water. Most of the analysis on the Biscuit data suggest that it is better to build a different univariate PLS regression model for each response instead of one multivariate PLS model. We construct here a univariate PLS regression model with the fat content taken as response. The dimension reduction aspect of this model is very important since no analyst wants to work on a 700-dimensional space. The use of a small number of PLS score vectors is certainly our goal. The data are initially centered by removing column means $\bar{\boldsymbol{x}}_j$ and $\bar{\boldsymbol{y}}$. The training set is used to construct the PLS regression model. Observation 23 has been previously

removed as it is suggested by most of the authors above. The constructed models are then validated on the test set of the 32 samples. We compute in the test set the cross validated mean squared error of prediction for one split as in paragraph 3.6.

The values of the prediction loss for $k = 1, \ldots, 7$ are:

$$33.7892,\ 26.8230,\ 8.9947,\ 4.0641,\ 23.2270,\ 35.1906,\ 49.2022.$$

Inclusion of more components increases the prediction loss and indicates overfitting. The minimum value for the prediction loss is reached for $k = 4$. The PLS regression model retains finally only 4 components and it largely reduces the dimension of the regression problem. The regression coefficients in terms of the derived feature $\boldsymbol{T}_4$ is

$$\widehat{\boldsymbol{q}}_4 = (0.949, 11.625, 7.888, 10.312),$$

which when it is transformed back in terms of the original feature $\boldsymbol{X}$ it results in the PLS implied regression coefficient vector $\widehat{\boldsymbol{\beta}}_4^{pls}$. The latter has 700 elements as this is the number of the original predictors. The transformation from $\boldsymbol{q}_4$ to $\widehat{\boldsymbol{\beta}}_4^{pls}$ is done using the expression (3.12). The resulting implied regression coefficient vector is illustrated in Figure 3.3.

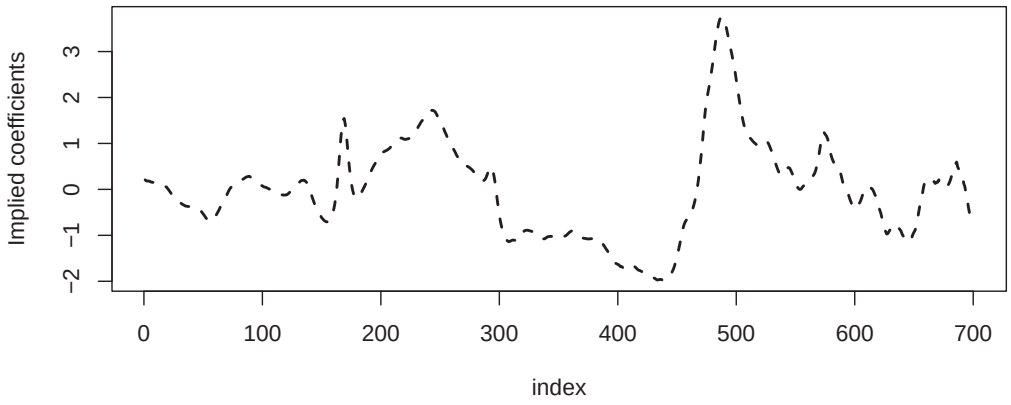


Figure 3.3: Biscuit data. Implied regression coefficients for the PLS regression model including 4 components.

Finally, in order to make predictions on new samples one should add the means on the training data ($\bar{\boldsymbol{x}_j}$ and $\bar{\boldsymbol{y}}$) which have been initially removed. Thus for a new set of data $\boldsymbol{X}^\star = (\boldsymbol{x}_1^\star, \ldots, \boldsymbol{x}_p^\star)$ the fitted values $\widehat{\boldsymbol{y}}^\star$ are given according

to

$$\widehat{\boldsymbol{y}}^{\star} = \bar{\boldsymbol{y}} - \sum_{i=1}^{n} \dot{\boldsymbol{x}}_i^{\star\prime} \widehat{\boldsymbol{\beta}}_4^{pls}, \tag{3.35}$$

where $\dot{\boldsymbol{x}}_i^{\star} = \Big((x_{i1}^{\star} - \bar{x}_1), \ldots, (x_{ip}^{\star} - \bar{x}_p) \Big)$ with the means $\bar{x}_j$ corresponding to the training data means.

3.8 Bibliographic notes

For general handbooks and tutorials on PLS methods one can see Geladi and Kowalski (1986); Martens and Naes (1989), as well as Tenenhaus (1998), and Wold et al. (2001). For a discussion on PLS statistical models see Helland (2001, 1990), while for an illustration of PLS regression statistical properties and its connection to numerical analysis see Manne (1987); Helland (1988); Höskuldsson (1988); Phatak and de Hoog (2001), Goutis (1996); Butler and Denham (2000). For PLS methods in chemometrics there is a huge amount of literature; one can see Martens and Naes (1989) and the references given therein. For the use for PLS in brain image and microarray experiments see McIntosh et al. (1996) and Nguyen and Rocke (2002a,b).

Chapter 4

The Partial LAD regression

4.1 The PLAD regression method

The Partial LAD (PLAD) regression uses the L_1 norm associated with least absolute deviations regression instead of the L_2 norm of PLS regression. It takes the structure of the PLS algorithm for univariate partial least squares regression, (see Tenenhaus, 1998), and similarly extracts components $\boldsymbol{t}$, in directions that depend upon the response variable. In PLAD these directions are determined by a Gnanadesikan-Ketterning (GK) covariance estimate (see Gnanadesikan and Kettenring, 1972) that replaces the usual variance based on the L_2 norm with mad, the median absolute deviation based on L_1:

$$\mathsf{mad}_n(\boldsymbol{x}) = \mathsf{med}_n\,|\boldsymbol{x} - \mathsf{med}_n(\boldsymbol{x})|.$$

We use the superscript *mad* to emphasize that covariance is calculated from mad instead of from the more common variance. We also use the suffix *lad* in the final estimates in order to emphasize that these depend upon the L_1 norm criteria in the Partial LAD algorithm.

4.1.1 The PLAD regression algorithm

With $i = 1, \ldots, n$ and $j = 1, \ldots, p$, denoting observation units and predictors, the PLAD regression algorithm is given in Algorithm 4.6.

[1]This chapter is based on the following papers:
1. Dodge, Y. and Kondylis, A. and Whittaker, J. (2004), Extending PLS1 to PLAD regression and the use of the L1-norm in soft modelling. COMPSTAT 2004, Eds. J. Antoch, pages 935–942, Physica/Verlag, Springer.
2. Kondylis, A. and Whittaker, J. (2006), Bootstrapping Partial LAD regression, to appear in the Special Issue on PLS methods in *Computational Statistics*.

Algorithm 4.6 Partial LAD regression

1. The original centered data $\mathcal{D} = (\boldsymbol{X}, \boldsymbol{y})$; $\boldsymbol{X}_0 \leftarrow \boldsymbol{X}$;

2. For $k = 1, \ldots, p$

2a. compute $\boldsymbol{w}_k^{mad}$ from:

$$w_{jk}^{mad} = \frac{1}{4}\left(\mathsf{mad}_n^2(\boldsymbol{x}_{jk-1} + \boldsymbol{y}) - \mathsf{mad}_n^2(\boldsymbol{x}_{jk-1} - \boldsymbol{y})\right); \tag{4.1}$$

2b. scale $\boldsymbol{w}_k^{mad}$ to 1;
2c. build the component

$$\boldsymbol{t}_k = \boldsymbol{X}_{k-1}\boldsymbol{w}_k^{mad}; \tag{4.2}$$

2d. orthogonalise each $\boldsymbol{x}_{jk-1}$, $j = 1, \ldots, p$ with respect to $\boldsymbol{t}_k$, to give $\boldsymbol{X}_k$;

3. Compute the LAD regression to give the fitted vectors $\widehat{\boldsymbol{y}}_k = \boldsymbol{T}_k\, \widehat{\boldsymbol{q}}_k^{\,lad}$, where $\boldsymbol{T}_k = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_k)$ is the score matrix, and $\widehat{\boldsymbol{q}}_k^{lad} = (\widehat{q}_1^{lad}, \ldots, \widehat{q}_k^{lad})'$ is the LAD regression coefficient.

4. Recover the implied partial LAD regression coefficients from $\widehat{\boldsymbol{\beta}}_k = \widetilde{\boldsymbol{W}}_k\, \widehat{\boldsymbol{q}}_k^{lad}$, where the matrix $\widetilde{\boldsymbol{W}}_k$ with columns $\boldsymbol{w}_k$ is expressed in terms of the original $\boldsymbol{x}_j$.

4.1.2 PLAD regression properties, coefficients and prediction

In contrast to principal components regression, which extracts the same components whatever the response, PLS and PLAD regression share similar properties, we note

1. $\boldsymbol{y} = q_1\,\boldsymbol{t}_1 + \ldots + q_k\,\boldsymbol{t}_k + \boldsymbol{\epsilon}$,
2. $\boldsymbol{X} = \boldsymbol{p}_1\,\boldsymbol{t}_1 + \ldots + \boldsymbol{p}_k\,\boldsymbol{t}_k + \boldsymbol{f}$,
3. $\mathsf{cov}(\boldsymbol{t}_i, \boldsymbol{t}_j) = 0 \quad \text{for} \quad i \neq j$,

where $\boldsymbol{e}$ and $\boldsymbol{f}$ correspond to residual terms, and $\boldsymbol{p}_k$ are the $\boldsymbol{X}$-loadings. PLAD builds a regression model at each iteration k that relates the predictors to the response according to

$$\widehat{\boldsymbol{y}}_k = \sum_{j=1}^{p} \widehat{\beta}_{jk}\,\boldsymbol{x}_j. \tag{4.3}$$

The implied regression coefficients $\widehat{\beta}_{jk}$ are determined by the derived components retained in the final regression model. In fact,

$$\widehat{\boldsymbol{\beta}}_k^{plad} = \widetilde{\boldsymbol{W}}_k\,\widehat{\boldsymbol{q}}_k^{lad}, \tag{4.4}$$

with $\widetilde{\boldsymbol{W}}_k = \boldsymbol{W}_k(\boldsymbol{P}_k'\,\boldsymbol{W}_k)^{-1}$ being the X-weights expressed in terms of the original predictors. Expression (4.4) shows clearly the very similar structure of PLAD to PLS regression, while it reveals a main difference. PLAD regression deflates the X-data similar to PLS regression. It retains therefore orthogonal components, while it recovers the weight vectors $\boldsymbol{w}_k$ pooled in matrix $\boldsymbol{W}_k$ in terms of the original predictors (instead of deflated data) according to $\widetilde{\boldsymbol{W}}_k = \boldsymbol{W}_k(\boldsymbol{P}_k'\,\boldsymbol{W}_k)^{-1}$. Yet, the PLAD coefficients $\boldsymbol{q}_k$ are obtained using LAD regression of the scores $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_k$ on the response $\boldsymbol{y}$. While for PLS regression at dimension k the coefficients $q_1, \ldots, q_{k-1}$ remain the same, for PLAD regression this is not the case. Therefore we use the suffix *plad* to indicate that the coefficients q_k depend on LAD regression. In the same sense we use the suffix *plad* in $\boldsymbol{w}_k^{plad}$ to emphasize that the direction vectors depend upon the GK-type covariance as it is given in expression (4.1).

The overall prediction error omitted by the partial LAD regression model in (4.3) for the training data (data at hand) $\mathcal{D}$ using the standard root mean squared error (RMSE) loss function is given by

$$\mathcal{EL}(\mathcal{D}) = \sqrt{\mathsf{E}_n[\mathcal{L}(\boldsymbol{y}, \widehat{\boldsymbol{y}}_k)]} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_{ik})^2}. \tag{4.5}$$

Recall the notation $\mathsf{E}_n[\cdot]$ indicates averaging over the n observations in $\mathcal{D}$; the latter are the training data $(\boldsymbol{X}, \boldsymbol{y})$. An alternative loss function which could be used is the *absolute error* in which case

$$\mathsf{E}_n[\mathcal{L}(\boldsymbol{y}, \widehat{\boldsymbol{y}}_k)] = \frac{1}{n}\sum_{i=1}^{n} |\boldsymbol{y} - \widehat{\boldsymbol{y}}_k\,|.$$

The latter may be used to extend the use of the L_1 norm on assessing the prediction error by calculating the absolute deviations of each y_i from $\widehat{y}_{ik}$, where $\widehat{y}_{ik}$ corresponds to the predicted value for observation i from a PLAD regression model containing k components. The idea behind the use of the absolute error is that it may render the prediction error assessment less sensitive to outliers.

The essential motivation for PLAD regression is to model the median of the response, instead of the mean as with PLS, and so it employs the L_1 instead of the L_2 norm. Additionally, as LAD regression is less sensitive to outliers than least squares regression PLAD may profit in the same way. However, PLAD is not a robust alternative to PLS for two main reasons: firstly, because the partial LAD algorithm does not bound the influence arising from high leverages in the predictor space. This is verified by the scores expression in (4.2). Secondly, because of the GK type of covariance, with the mad replacing the variance, has unstable robust properties depending on the scales of $\boldsymbol{X}$ and $\boldsymbol{y}$, (Huber, 1981). A simple alternative is to set

$$w_{jk} = \frac{1}{4\,\alpha_j\beta}(\,\mathsf{mad}_n^2(\alpha_j \boldsymbol{x}_{jk-1} + \beta \boldsymbol{y}) - \,\mathsf{mad}_n^2(\alpha_j \boldsymbol{x}_{jk-1} - \beta \boldsymbol{y})),$$

with $\alpha_j = 1/\,\mathsf{mad}_n(\mathbf{x}_j)$ and $\beta = 1/\,\mathsf{mad}_n(\boldsymbol{y})$. This is the definition we use. The rationale behind the idea of replacing formula (4.1) by the formula given above is that the last one is much less sensitive on the choice of scale for the predictor variables. Indeed, this becomes essential in PLS regression which is not a scale invariant method.

4.2 Assessing the PLAD regression method using the bootstrap

4.2.1 PLS, PLAD and the bootstrap

The bootstrap (Efron and Tibshirani, 1993) is a resampling method which provides an assessment of uncertainty when theoretical solutions are not available, as is the case with PLS and PLAD regression. The general bootstrap algorithm is sketched in Algorithm 4.7.

The subscript R in $\mathsf{E}_R(\cdot)$, $\mathsf{var}_R(\cdot)$, and $\mathsf{sd}_R(\cdot)$ indicates averaging over the R bootstrap replicates.

Bootstrapping data or residuals?

In principle may base our analysis on bootstrapping the raw data $\mathcal{D}$ or on bootstrapping the residuals having fitted a regression. For several reasons we employ the former here. Bootstrapping residuals requires fixing the number of components k in order to define the fitted values and so determine the residuals. This may conflict with our interest in comparison of how different regression methods might choose k. Furthermore our interest is to compare the manner in which PLS

Algorithm 4.7 Bootstrap

Let $(y_1, \ldots, y_n)$ be an observed sample of size n, and θ the statistic of interest. For $b = 1, \ldots, R$ the number of repetitions:

1. Generate a random sample $(y_1^b, \ldots, y_n^b) \sim F$, with replacement, from the distribution F which is either given or estimated by the empirical distribution $\widehat{F}$.
2. Compute $\widehat{\theta}_b$ from the simulated data $(y_1^b, \ldots, y_n^b)$.

Use $(\widehat{\theta}_1, \widehat{\theta}_2, \ldots \widehat{\theta}_R)$ to estimate the sampling distribution of $\widehat{\theta}$ and summaries of interest such as $\mathsf{E}_R(\widehat{\theta})$, $\mathsf{var}_R(\widehat{\theta})$, and $\mathsf{sd}_R(\widehat{\theta})$.

and PLAD regression generalise performance to new observations rather than to new samples of residuals. More generally, bootstrapping data is less sensitive to assumptions made on the regression model (see Efron and Tibshirani, 1993).
We generate R bootstrap samples $\mathcal{D}^{\star\, b},\ b = 1, 2, \ldots, R$, each consisting of $(\boldsymbol{X}^{\star\, b}, \boldsymbol{y}^{\star\, b})$ obtained by resampling the rows of $(\boldsymbol{X}, \boldsymbol{y})$. We compare the bootstrap results from PLS and PLAD regression. When necessary we use a suffix *pls* or *plad* to denote quantities computed from the corresponding methods.
Let $s = s(\mathcal{D})$ be the statistic of interest calculated from the original data, and assumed to be invariant to permutation of the rows of the data $\mathcal{D}$. Its mean and standard deviation calculated over the bootstrap samples are given as

$$\mathsf{E}_R(s^\star) = \frac{1}{R}\sum_{b=1}^{R} s(\mathcal{D}^{\star\, b}) \text{ and } \mathsf{sd}_R(s^\star) = \sqrt{\frac{1}{R-1}\sum_{b=1}^{R}[s(\mathcal{D}^{\star\, b}) - \mathsf{E}_R(s^\star)]^2}. \quad (4.6)$$

Constructing confidence intervals and hypothesis testing based on the statistic s is then straightforward (see Efron and Tibshirani, 1993, chapters 12,13,14,16).

4.2.2 Bootstrapping the regression coefficients

Consider the implied coefficients $\widehat{\beta}_{jk},\ j = 1, 2, \ldots, p$ using PLAD from (4.4) and a similar expression for PLS. These statistics are invariant to permuting the rows of $\mathcal{D}$. From the bootstrap we compute $\mathsf{E}_R(\widehat{\beta}^\star_{jk})$ and $\mathsf{sd}_R(\widehat{\beta}^\star_{jk})$ as at (4.6) above, that is

$$\mathsf{E}_R(\widehat{\beta}^\star_{jk}) = \frac{1}{R}\sum_{b=1}^{R} \widehat{\beta}^{\star\, b}_{jk} \text{ and } \mathsf{sd}_R(\widehat{\beta}^\star_{jk}) = \sqrt{\frac{1}{R-1}\sum_{b=1}^{R}[\widehat{\beta}^{\star\, b}_{jk} - \mathsf{E}_R(\widehat{\beta}^\star_{jk})]^2}. \quad (4.7)$$

The $(1-\alpha)\%$ percentile bootstrap confidence limits for $\widehat{\beta}_{jk}$ are

$$[\mathrm{q}^\star_{\alpha/2},\ \mathrm{q}^\star_{1-\alpha/2}] \quad (4.8)$$

corresponding to the $\alpha/2$ and $1-\alpha/2$ empirical quantiles of the distribution of the bootstrap replicates for $\widehat{\beta}^{\star\, b}_{jk}$. We set α equal to 0.05. Bootstrap confidence

intervals may be also obtained by using the ABC or the BCa methods (see Efron, 1987). We use the percentile approach here because it is simpler to interpret and it is not more computationally expensive.

4.2.3 Bootstrapping the prediction error

For any value of k and by means of the RMSE, the *apparent* prediction error $\mathcal{EL}(\mathcal{D})$ at (4.5) is invariant to permutation, and its bootstrap standard deviation is

$$\mathsf{sd}_R\left(\mathcal{EL}^\star\right) = \sqrt{\frac{1}{R-1}\sum_{b=1}^{R}\Big(\mathcal{EL}(\mathcal{D}^{\star\,b}) - \mathsf{E}_R(\mathcal{EL}^\star)\Big)^2}, \tag{4.9}$$

from (4.6). These quantities are computed for PLS and PLAD, giving $\mathsf{sd}_R(\mathcal{EL}^{\star\,pls})$ and $\mathsf{sd}_R(\mathcal{EL}^{\star\,plad})$.
The mean bootstrapped RMSE

$$\mathsf{E}_R\left(\mathcal{EL}^\star\right) = \frac{1}{R}\sum_{b=1}^{R}\mathcal{EL}(\mathcal{D}^{\star\,b}), \tag{4.10}$$

is the *resampling* prediction error rate. It is commonly used for model selection as the magnitude of the errors decrease with the number of the components k retained in the final regression model.

The bootstrap estimate of the apparent prediction error can be improved by subtracting the bias induced in using E_R instead of E_n. That is, by using $\widehat{\widehat{F}}$ instead of F (see Algorithm 4.7, step 1). This is the *optimism*. The bootstrap estimate of the expected optimism $\widehat{\omega}(\widehat{F})$ is obtained by

$$\widehat{\omega}(\widehat{F}) = \mathsf{E}_n\left[\mathcal{EL}(\mathcal{D}^{\star,b}, F) - \mathcal{EL}(\mathcal{D}^{\star,b}, \widehat{F})\right], \tag{4.11}$$

with $\mathcal{EL}(\mathcal{D}^{\star,b}, \widehat{F})$ equally denoting the resampling error rate $\mathcal{EL}(\mathcal{D}^{\star\,b})$. This is often called type II error. The first term in the right hand side of expression (4.11) corresponds to the loss induced by testing predictions for models constructed on the bootstrap sample on the original response, and it is often called type I error.

The averaged optimism above is approximated for the R bootstrap samples by

$$\widehat{\omega}(\widehat{F}) = \frac{1}{\sqrt{n}\,R}\left\{\sum_{b=1}^{R}\sqrt{\sum_{i=1}^{n}(y_i - \boldsymbol{x}_i'\widehat{\boldsymbol{\beta}}_k^{\star\,b})^2} - \sum_{b=1}^{R}\sqrt{\sum_{i=1}^{n}(y_i^{\star\,b} - \boldsymbol{x}_i^{\star\,b\,\prime}\widehat{\boldsymbol{\beta}}_k^{\star\,b})^2}\right\}. \tag{4.12}$$

The final estimate for the prediction error (FPE) of the partial LAD regression model including k components is given by

$$FPE_k^{plad} = \mathcal{EL}(\mathcal{D}, \widehat{F}) + \widehat{\omega}(\widehat{F}), \tag{4.13}$$

with a similar expression for the PLS regression model.

Further improvement in assessing the prediction error can be obtained by the use of the 0.632 bootstrap estimator (see Efron and Tibshirani, 1997). In that case the optimism is estimated in the apparent prediction error for observation i by using bootstrap samples that do not include the latter observation. The 0.632 estimate is computationally more expensive. We therefore base our estimation for prediction error on the expression (4.13).

4.3 Experience based on real data sets

4.3.1 The Diabetes data

The Diabetes data set (see Efron et al., 2004) consists of 442 diabetes patients measured on 10 baseline variables. These variables are: age, sex, body mass index, average blood pressure, and six blood serum measurements. The response of interest is a quantitative measure of disease progression one year after. The correlations between the 10 baseline variables are relatively large, especially between the six blood serum measurements.

The data are first standardized since the variables are recorded on different measurement scales. We obtain and we present below estimates for $k = 1, \ldots, 5$, for a number of the bootstrap replicates set to $R = 500$. Table 4.1 gives the values of the apparent and the resampling prediction errors for the diabetes data set together with the bootstrap estimates for the optimism and the final prediction error. The results are also illustrated in Figure 4.1.

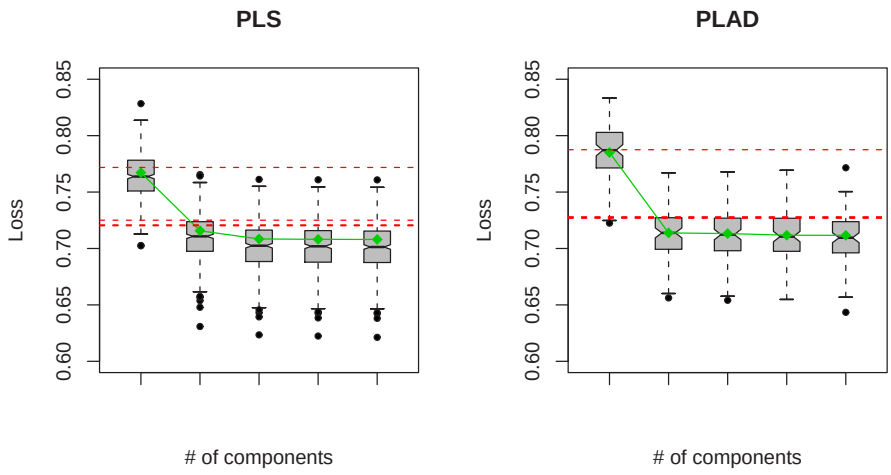


Figure 4.1: Diabetes data. *Apparent* and *resampling prediction error* for PLS (left panel) and PLAD (right panel) regression models. The *apparent* error follows the solid line (green) while the *resampling* prediction error is summarized by boxplots. The horizontal dashed lines (red) correspond to the final prediction error.

Table 4.1: Diabetes data: The *apparent* and the *resampling* prediction error together with the *optimism* and the *FPE* estimate for PLS and PLAD methods. The standard deviations for the *resampling* error are given in the parenthesis.

	PLS			
k	AE	RE (sd)	$\widehat{\omega}(\widehat{F})$	*FPE*
1	0.7672	0.76 (0.0199)	0.00462	0.7718
2	0.7156	0.71 (0.0204)	0.00936	0.7250
3	0.7084	0.70 (0.0208)	0.01185	0.7203
4	0.7081	0.70 (0.0209)	0.01219	0.7203
5	0.7080	0.70 (0.0209)	0.01288	0.7208
	PLAD			
k	AE	RE (sd)	$\widehat{\omega}(\widehat{F})$	*FPE*
1	0.7850	0.78 (0.0235)	0.0024	0.7875
2	0.7137	0.71 (0.0227)	0.0136	0.7274
3	0.7132	0.71 (0.0228)	0.0145	0.7277
4	0.7117	0.71 (0.0226)	0.0150	0.7267
5	0.7116	0.71 (0.0222)	0.0164	0.7279

The results for the Diabetes data set show an equivalence between PLS and PLAD regression. Both methods agree on the number of the components for the final model and provide stable estimates. Note that the resulting estimates for the bootstrap standard error as well as for the optimism are nearly equal especially for $k \geq 2$. The PLAD regression performs here very well, equally to the ordinary PLS regression model.

It is very interesting to make comparisons between the two methods with PC regression. The latter extract components which do not depend on the response; this being in full contrast to PLS and PLAD which take into account $\boldsymbol{y}$. Table 4.2 gives the correlations between the response vector $\boldsymbol{y}$ and the PC, PLS, and PLAD regression score vectors $\boldsymbol{t}_k$ for $k = 1, \ldots, 5$. Firstly, it is verified that PLS extracts components in order of association with $\boldsymbol{y}$. It turns out from Table 4.2 that PC regression analysis discards the fourth component by constructing a model on 2 or 3 components. Yet, the fourth PC score vector has a non negligible correlation to the response and therefore its inclusion in the regression model may increase the predictive ability of the latter.

Finally, we should notice that the Diabetes example show an equivalence between PLS and PLAD regression for a data set with collinearity problems arising on the predictors space, and a relatively large number of observations. The fitted mean of PLS regression and the fitted median of PLAD regression in this case are very close since n is quite large and no heavy tails appear on the distribution of the response.

Table 4.2: Diabetes data: Correlation between the response $\boldsymbol{y}$ and the PC, PLS, and PLAD regression score vectors.

k	PC	PLS	PLAD
1	-0.5671	0.6403	0.6203
2	-0.3253	0.2766	0.3263
3	0.0902	0.1016	0.0335
4	0.1848	0.0223	0.0558
5	0.0739	0.0118	-0.0086

4.3.2 NIR data sets

The field of near infra-red (NIR) experimentations is a principal application of PLS methods (Martens and Naes, 1989). We use here three NIR data sets: the Wheat data (Fearn, 1983), the Fish data (Naes, 1985) and the Octane data, (Tenenhaus, 1998). The predictors in each data set are spectra at different number of wavelengths, and are highly collinear. The reflectance of the NIR radiation by the sample units at different wavelengths are used to model chemical concentrations. The Wheat and the Fish data have a relatively small number of regressors, while the Octane data count more than 200 regressors for 39 observations. Further details on these data sets are given in the references.

Results

The analyzed data are initially centered but not rescaled as they are measured on similar physical scales. The number k_{max} is known for each data set due to previous analysis. Nevertheless, in our results we obtain and we present estimates for $k = 1, \ldots, 4$. The number of the bootstrap replicates is set to $R = 500$ for all data sets. Table 4.3 gives the values of the apparent and the resampling prediction errors for the NIR data sets together with the bootstrap estimates for the optimism.

The prediction errors from PLS and PLAD are close for both the Wheat and the Fish data sets. The PLAD estimates of the resampling prediction error are slightly more variable in comparison to the PLS. For the Octane data, PLAD regression reduces the apparent error on the second component rather more than PLS does (though including the third component brings PLS and PLAD together). However this reduction is accompanied by a relatively large variation in the second PLAD component, which is illustrated in Figure 4.2, where the solid line (green) represents the apparent prediction error and the boxplots are constructed according to the prediction error values on the bootstrap samples. Finally, the horizontal dashed lines (red) correspond to the final prediction error

Table 4.3: NIR data: RMSE: apparent error (AE), resampling error (RE), with standard errors (in parentheses), and the expected optimism $\widehat{\omega}(\widehat{F})$ for the three NIR data sets, and for $k = 1, \ldots, 4$.

$k=1$	Data	Method	AE	RE (sd)	$\widehat{\omega}(\widehat{F})$
	Wheat	PLS	1.2326	1.1690 (0.1282)	0.0882
		PLAD	1.2181	1.1919 (0.1528)	0.1101
	Fish	PLS	3.0161	2.8758 (0.3484)	0.2445
		PLAD	3.0220	2.9429 (0.3601)	0.2612
	Octane	PLS	1.7395	1.6041 (0.2691)	0.0901
		PLAD	1.8308	1.8325 (0.2443)	0.1448
$k=2$	Data	Method	AE	RE (sd)	$\widehat{\omega}(\widehat{F})$
	Wheat	PLS	0.9790	0.8050 (0.1979)	0.1114
		PLAD	0.9688	0.9697 (0.1957)	0.1352
	Fish	PLS	1.7914	1.6728 (0.1998)	0.1748
		PLAD	1.7234	1.6881 (0.2530)	0.1570
	Octane	PLS	0.6973	0.7364 (0.1152)	0.0408
		PLAD	0.5878	0.5783 (0.1740)	0.0472
$k=3$	Data	Method	AE	RE (sd)	$\widehat{\omega}(\widehat{F})$
	Wheat	PLS	0.2413	0.2127 (0.0546)	0.0545
		PLAD	0.3352	0.2901 (0.1200)	0.0792
	Fish	PLS	1.2777	1.1170 (0.2094)	0.3396
		PLAD	1.2900	1.2796 (0.2771)	0.3063
	Octane	PLS	0.2574	0.3801 (0.1469)	0.0260
		PLAD	0.3664	0.3878 (0.1515)	0.0434
$k=4$	Data	Method	AE	RE (sd)	$\widehat{\omega}(\widehat{F})$
	Wheat	PLS	0.1968	0.1673 (0.1968)	0.0569
		PLAD	0.2112	0.2049 (0.0557)	0.0709
	Fish	PLS	1.2266	1.0616 (0.2042)	0.3457
		PLAD	1.2713	1.1907 (0.2566)	0.3102
	Octane	PLS	0.2409	0.3555 (0.1533)	0.0482
		PLAD	0.2740	0.2511 (0.0478)	0.0505

estimate given by equation (4.13).

Table 4.3 leads to the conclusion that PLS generally reaches slightly lower levels of prediction error than PLAD regression. PLAD regression in certain cases has lower apparent prediction error for the data at hand in comparison to PLS. For the Octane data set, the apparent PLAD error provides some strong evidence that two components (instead of three for PLS) could be retained in

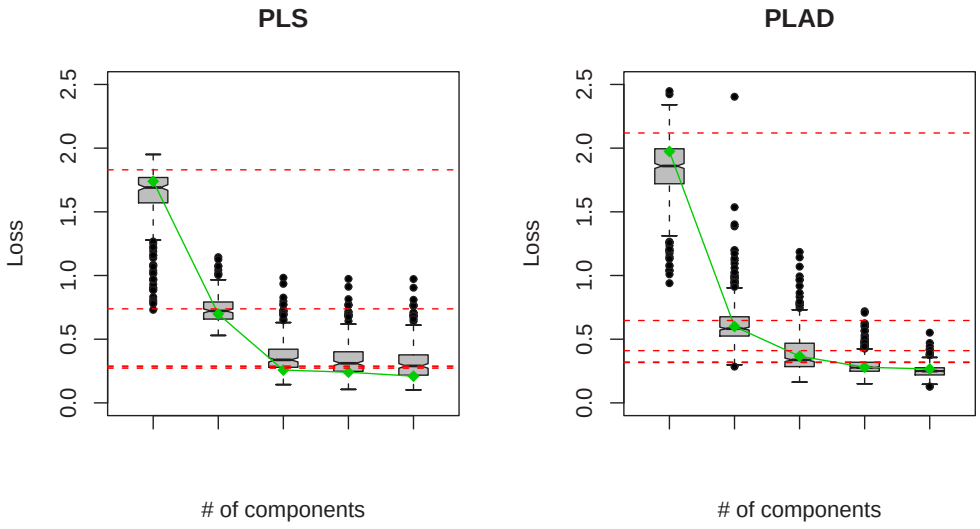


Figure 4.2: Octane data. *Apparent* and *resampling prediction error* for PLS (left panel) and PLAD (right panel) regression models. The *apparent* error follows the solid line (green) while the *resampling* prediction error is summarized by boxplots. The horizontal dashed lines (red) correspond to the final prediction error.

the final model. This is probably due to six outlying observations in the Octane data, (Dodge et al., 2004). Outlying observations and high leverages are also found in the Fish data set, where observations 43, 44, 45 are high leverages while observations 1 and 43 are outliers. For PLAD regression the difference in the prediction loss for $k = 2$ and $k = 3$ is much less than for PLS. This is in line with our findings for the Octane data. However for both data sets the resampling procedure does not indicate that this performance generalizes (note the right hand panel of Figure 4.2 for $k = 2$).
Figures 4.3 and 4.4 display the percentile bootstrap confidence limits for the implied regression coefficients from the Octane and the Wheat data. The PLAD implied regression coefficients in the right panel of Figure 4.3 are more variable than PLS coefficients given in the left panel.

This is also apparent for the Fish and Wheat data sets as well. We illustrate the latter in Figure 4.4 where the estimated regression coefficients for PLAD and PLS are displayed together. The larger intervals for PLAD regression coefficients show them to be more variable than the PLS coefficients.

4.4 Experimental data

We consider two sets of data constructed from simulation. The first from a switching regression model is an example where the results of PLS and PLAD

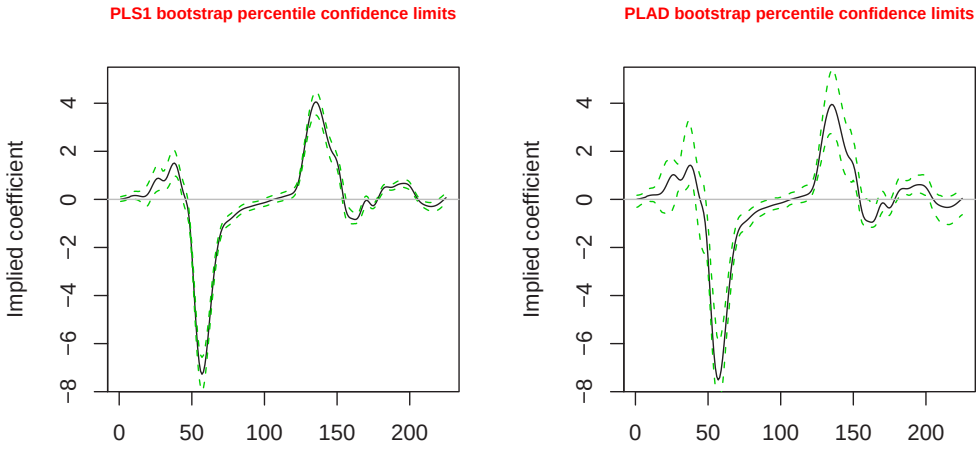


Figure 4.3: Octane data. Regression coefficients for PLS (left panel) and PLAD (right panel) regressions using on three components. Percentile bootstrap confidence limits for percentiles 0.025 and 0.975 are displayed as dashed lines (green).

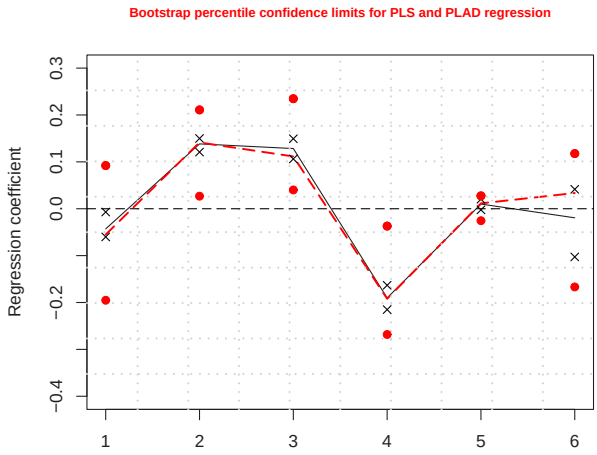


Figure 4.4: Fearn's Wheat data. Regression coefficients for PLS (black solid line) and PLAD (red dashed line) regression models on three components. The percentile bootstrap confidence limits for percentiles 0.025 and 0.975 are displayed as points, with bullets for PLAD and crosses for PLS.

differ because they are estimating different features of the data. The second illustrates the effect of outlier contamination on a standard factor model often

employed to illustrate PLS regression (see Martens and Naes, 1989).

4.4.1 Data from a switching regression model

There are underlying latent variables here with distributions $M \sim$ Uniform$(0, 1)$, $Z \sim$ N$(0, 1)$, and, to induce a switch, $S \sim$ Bernoulli(0.65) on $\{-1, 1\}$. The covariates are partitioned into $X = (X_1, X_2)$, and are related to the latent variables by

$$X_1|M \sim \text{N}(m1, I), \;\; X_2|Z \sim \text{N}(z1, I), \text{ and } Y|(S, M) \sim \text{N}(sm + z, 1).$$

The non linear model generated shows that scatter plots of elements of X_1 with Y are somewhat triangular with the lower quantile decreasing with X_1 but the upper quartile increasing.

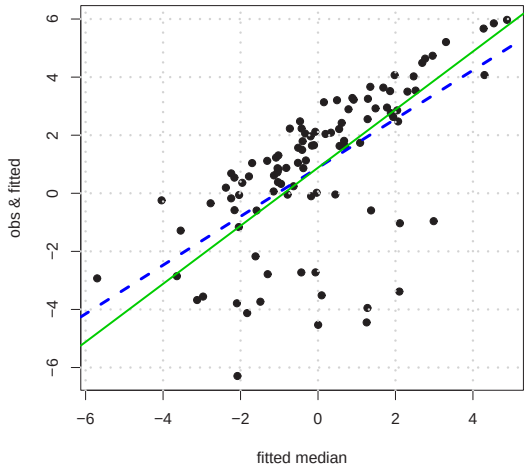


Figure 4.5: Data from a switching regression model. The PLS and PLAD fitted values plotted versus the fitted median. The fitted median and the fitted mean are illustrated by the dashed and solid line, respectively

A sample of $n = 100$ observations based on $p = 20$ explanatory variables is generated, and the results are displayed in Figure 4.5. The fitted median and the fitted mean resulting from the PLAD and PLS regression models are illustrated in Figure 4.5 by the dashed and solid line, respectively. In this figure the fitted PLS and PLAD lines are plotted versus the fitted median. Differences between the two methods are here detected on the two fits and the PLS and PLAD lines are now well distinguished.

4.4.2 Data from the bilinear factor model

Both PLS and PLAD methods are based on the bilinear model (see also paragraph 3.2) given by

$$\boldsymbol{y} = q_1\,\boldsymbol{t}_1 + \ldots + q_k\,\boldsymbol{t}_k + \boldsymbol{\epsilon}, \quad \text{and} \quad \boldsymbol{X} = \boldsymbol{p}_1\,\boldsymbol{t}_1 + \ldots + \boldsymbol{p}_k\,\boldsymbol{t}_k + \boldsymbol{f}. \tag{4.14}$$

We simulate here the components $\boldsymbol{T}$ by n independent realizations of

$$\boldsymbol{T} \sim \mathcal{N}(\mathbf{0}_k,\, \Sigma_{kk}).$$

The k-variate normal distribution has parameters $\mathbf{0}_k$ (as the data are centered) and the variance-covariance matrix Σ_{kk} is diagonal with specified variances

$$\Sigma_{kk} = \begin{pmatrix} \mathsf{var}(\boldsymbol{t}_1) & 0 & \ldots & 0 \\ 0 & \mathsf{var}(\boldsymbol{t}_2) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \mathsf{var}(\boldsymbol{t}_k) \end{pmatrix}.$$

The data set $\boldsymbol{X}$ and $\boldsymbol{y}$ is obtained according to equation (4.14) for a specified choice of residual structure in $\boldsymbol{f}$ and $\boldsymbol{\epsilon}$. We fix $k_{max} = 3$, $\boldsymbol{P} = \boldsymbol{I}_{3,p}$, and $\boldsymbol{q} = (1,1,1)'$. Finally the matrix Σ_{kk} is set to $\mathrm{diag}\,(10,5,1)$, that is

$$\Sigma_{kk} = \begin{pmatrix} 10 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

We generate the artificial data set as follows: the elements of the error term $\boldsymbol{\epsilon}$ in expression (4.14) are independent normals, that is $\epsilon_i \sim \mathcal{N}(0, 0.01)$ where $i = 1, \ldots, n$. We then contaminate by replacing a small fraction, 10%, of the data set with outliers. The contaminated error vector is denoted as $\boldsymbol{\epsilon}_{cont}$ and its elements are generated according to $\epsilon_{i',cont} \sim \mathcal{N}(\mu, 1)$ with $\mu = 5$ and $i' = 1, \ldots, \ell$ for $\ell = 0.10 \cdot n$. The residual term $\boldsymbol{f}$ in expression (4.14) remains a random normal variate centered to zero, and no contamination on the X-space has been used. The dimensions of the data sets are set to $n = 100$ and $p = 50$. For the simulated data we apply bootstrap methods with R equal to 500.

Results

Table 4.4 gives the *apparent* and the *resampling* prediction error together with the *optimism* and the *FPE* estimate for both PLS and PLAD methods. For both regression models the simulation setting is verified since three components are retained. The PLAD resampling prediction error is slightly more variable than the PLS resampling prediction error for all the components.

The available information on prediction error assessment for the contaminated case is displayed in Figure 4.6. Figure 4.6 is similar to Figure 4.2 with the

Table 4.4: Data from the bilinear factor model: The *apparent* and the *resampling* prediction error together with the *optimism* and the *FPE* estimate for PLS and PLAD methods. The standard deviations for the *resampling* error is given in the parenthesis.

	PLS			
k	AE	RE (sd)	$\widehat{\omega}(\widehat{F})$	*FPE*
1	1.5066	1.48 (0.0906)	0.0203	1.5269
2	1.2406	1.22 (0.0508)	0.0258	1.2665
3	1.0479	1.01 (0.0465)	0.0478	1.0958
4	0.9557	0.87 (0.0467)	0.1755	1.1312
5	0.9397	0.83 (0.0497)	0.2300	1.1697
	PLAD			
k	AE	RE (sd)	$\widehat{\omega}(\widehat{F})$	*FPE*
1	1.3667	1.42 (0.1265)	0.0392	1.4060
2	1.1951	1.26 (0.1160)	0.0401	1.2353
3	1.0902	1.15 (0.1000)	0.0476	1.1379
4	1.0873	1.11 (0.0719)	0.0462	1.1335
5	1.0810	1.09 (0.0630)	0.0499	1.1309

horizontal dashed lines (red) indicating the final prediction error estimate (*FPE*) given by equation (4.13).

Both regression methods retain three components. In this sense contamination has no effect on model dimension neither for PLS nor for PLAD. Yet PLAD is accompanied by a larger variability of the resampling prediction error. This is seen on the first three boxplots (that correspond to $k = 1, 2, 3$) in the right panel of Figure 4.6. Note also that PLAD seems to be much less vulnerable to overfitting in comparison to PLS. Especially for $k \geq 3$ the PLAD *apparent error* is much closer to the *FPE* in comparison to PLS regression for which the *apparent error* decreases constantly with the number of the components.

4.5 Conclusions

The PLAD regression has been tested and compared to PLS regression using the bootstrap. In the limited examples and experiments considered, we have established that PLAD and PLS estimate different features, but PLS is superior to PLAD in the sense that the implied regression coefficient estimates have smaller bootstrap confidence intervals; so that when the features are the same PLS is to be preferred. The magnitude of the difference is in line with the well known ratio of the standard deviation of the sample mean compared to that of the sample median when sampling from a Normal distribution. This gives some confidence

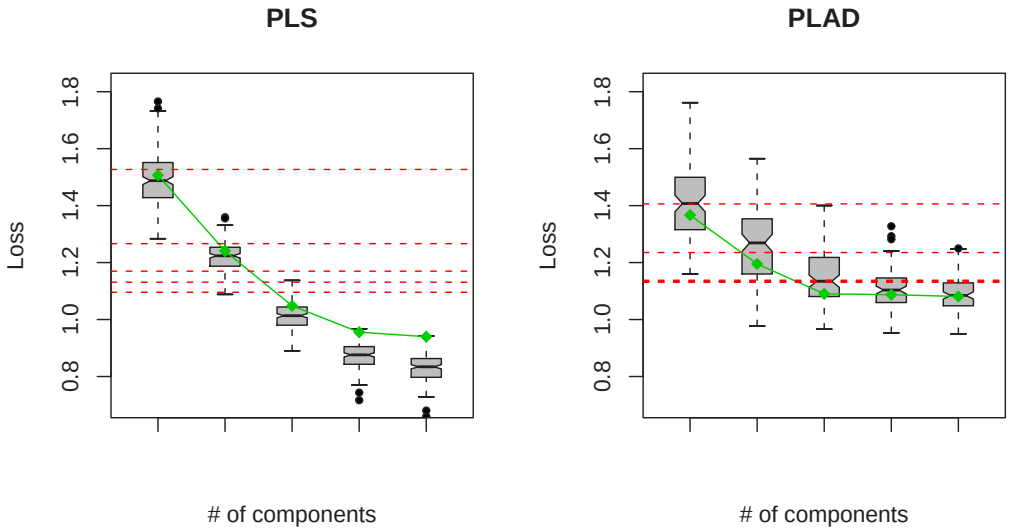


Figure 4.6: Data from the bilinear factor model. *Apparent* and *resampling prediction error* for PLS (left panel) and PLAD (right panel) regression models. The *apparent* error is displayed by solid lines (green) while the *resampling* prediction error is summarized in boxplots. The horizontal dashed lines (red) correspond to the final prediction error.

in the basic structure of the PLAD algorithm when the response distribution is far from Normal. Furthermore, we have established that PLAD performs as well as PLS in model selection and prediction error assessment with final prediction error estimates nearly equal.

Using the bootstrap reveals two main drawbacks for PLAD regression. Firstly, PLAD estimates of prediction error are more variable than PLS estimates, so that achieving a small number of retained components might be more hazardous. Secondly, the PLAD regression method is computationally more expensive than PLS regression. This is due to the LAD algorithm as well as the GK-type covariance which demands the computation of $\boldsymbol{w}_k^{mad}$ for all the columns of matrix $\boldsymbol{X}$. A nice feature observed in the experimental studies is that PLAD is more resistant to overfitting in comparison to PLS. The apparent error from PLAD is always closer to the FPE, especially for $k \geq k_{max}$. In these cases the variability of PLAD resampling error reaches smaller levels which are comparable to those obtained from PLS. Finally, as seen in the Diabetes example, PLAD and PLS provide nearly equal results. This happens when the number of the recorded observations is relatively large.

Chapter 5

PLS regression using the BACON algorithm

5.1 The BACON algorithm for outlier detection

The BACON algorithm was presented in Billor et al. (2000). It is generally based on the methods proposed by Hadi (1992b, 1994). The BACON algorithm starts by forming an initial *basic* subset of m observations that is presumably free from multivariate outliers, where m is specified by the data analyst. Then observations that are consistent with the basic subset are added to the initial basic subset. If all the observations are added to the basic subset, the data set is declared to be free from outliers, otherwise the observations that are not consistent with the basic subset are declared as multivariate outliers.

The BACON algorithm can be applied to multivariate data as well as to regression problems. In the former case the algorithm identifies outliers and returns robust estimates of the variance-covariance of the predictor variables $\boldsymbol{X}$. In the regression setting, BACON algorithm uses robust estimates obtained from the multivariate case, and for a response vector $\boldsymbol{y}$, it robustly estimates the regression parameter vector $\boldsymbol{\beta}$.

We use here the multivariate BACON algorithm. We treat two cases which appear in statistical practice and which are of our special interest. Firstly, the case where $\boldsymbol{X}$ is of full-rank; hence $(\boldsymbol{X}'\boldsymbol{X})^{-1}$ exists, yet it is near singular. This is investigated in paragraph 5.1.1. Secondly, the case where $\boldsymbol{X}$ is rank-deficient which is treated in paragraph 5.1.2. Both cases result from collinear data where PLS regression is used as regularization regression. The latter case is further adopted for high dimensional data, where the number of the recorded variables exceeds the available number of observations.

[2]This chapter is based on the following papers:
1. Kondylis, A. and Ali S. Hadi (2006), Derived Components Regression using the BACON algorithm, In Press in *Computational Statistics & Data Analysis*.
2. Kondylis, A. and Ali S. Hadi (2006), The BACON approach for rank deficient data, *working paper*.

Finally, we make some assumptions on the population model. In particular, we assume without loss of generality that the data $(\boldsymbol{X}, \boldsymbol{y})$ are centered observations corresponding to sample realizations arising from a joint elliptically symmetric probability distribution family $\mathcal{F}_{(X,Y)}$. The population variance-covariance matrix $\boldsymbol{\Sigma}$ may be decomposed as

$$\begin{pmatrix} \boldsymbol{\Sigma}_{XX} & \boldsymbol{\Sigma}_{Xy} \\ \boldsymbol{\Sigma}_{yX} & \boldsymbol{\Sigma}_{y} \end{pmatrix}. \tag{5.1}$$

Given that sample means are equal to zero, the sample variance-covariance matrix is then proportional to

$$\begin{pmatrix} \boldsymbol{X}'\boldsymbol{X} & \boldsymbol{X}'\boldsymbol{y} \\ \boldsymbol{y}'\boldsymbol{X} & \boldsymbol{y}'\boldsymbol{y} \end{pmatrix}. \tag{5.2}$$

For the rank deficient case the assumptions above are hard to verify and further details are given in paragraph 5.1.2.

5.1.1 The BACON approach - the ill-conditioned case

The BACON algorithm assumes that the matrix $\boldsymbol{X}$ is of full-column rank and also come from an elliptically symmetric distribution. The algorithm starts by selecting an initial basic subset, $\boldsymbol{X}_b$, of size $r > p$. There are two versions for selecting $\boldsymbol{X}_b$. In Version 1, the initial basic subset $\boldsymbol{X}_b$ consists of the r observations with the smallest values of the *Mahalanobis distances*

$$d_i(\bar{\boldsymbol{x}}, \mathbf{S}) = \sqrt{(\boldsymbol{x}_i - \bar{\boldsymbol{x}})' \, \mathbf{S}^{-1} \, (\boldsymbol{x}_i - \bar{\boldsymbol{x}})}, \quad i = 1, \ldots, n, \tag{5.3}$$

where $\boldsymbol{x}_i'$ is the ith row of $\boldsymbol{X}$, and $\bar{\boldsymbol{x}}$ and $\mathbf{S}$ are the mean and variance-covariance matrix of the variables in $\boldsymbol{X}$, respectively. In Version 2, $\boldsymbol{X}_b$ consists of the r observations with the smallest values of the Euclidean distances from the median

$$d_i(\boldsymbol{x}_i, \mathbf{m}) = \|\boldsymbol{x}_i - \mathbf{m}\|, \quad i = 1, \ldots, n, \tag{5.4}$$

where $\|\,.\,\|$ denotes the euclidian vector norm. In the original BACON algorithm, $\mathbf{m}$ is taken to be the vector containing the coordinatewise medians. But, of course, other forms of medians could also be used here. For example, the multivariate L_1 median or the spatial median (see Hössjer and Croux, 1995, for theoretical properties and computation). The latter is the L_1 location estimator given by the solution of the following problem

$$\mathbf{m} = \max_{\boldsymbol{u}} \sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{u}\|, \text{ for } i = 1, \ldots, n. \tag{5.5}$$

The distances from the median provide a robust initial subset with comparison to the Mahalanobis distances, but the Mahalanobis distance is affine equivariant and has low computational cost.

The initial basic subset $\boldsymbol{X}_b$ includes the m observations with the smallest distances d_i. The size of the initial basic subset is $r = cp$, where c is a multiplier, that is, at least c observations per parameter are used. Billor et al. (2000) suggest setting c to 3, 4, or 5. Let $\bar{\boldsymbol{x}}_b$ and $\mathbf{S}_b$ be the mean and covariance matrix of the observations in the current basic subset $\boldsymbol{X}_b$. Compute the robust distances

$$d_i(\bar{\boldsymbol{x}}_b, \mathbf{S}_b) = \sqrt{(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_b)' \, \mathbf{S}_b^{-1} \, (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_b)}, \quad i = 1, \ldots, n. \tag{5.6}$$

The BACON algorithm lets the current basic subset $\boldsymbol{X}_b$ to increase until it no longer changes. At each iteration, the basic subset includes the observations with

$$d_i(\bar{\boldsymbol{x}}_b, \, \mathbf{S}_b) < c_{npr} \, \chi_{p,\alpha/n}, \tag{5.7}$$

where $\chi^2_{p,\alpha}$ is the $1-\alpha$ percentile of the χ^2 distribution with p degrees of freedom, and

$$c_{npr} = c_1 + c_2 \tag{5.8}$$

is a correction factor, where

$$c_1 = 1 + \frac{p+1}{n-p} + \frac{2}{n-1-3p}, \tag{5.9}$$

$$c_2 = \max\left\{0, \frac{n+p+1-2r}{n+p+1+2r}\right\}, \tag{5.10}$$

and r is the size of the current subset. The BACON algorithm for identifying outliers in full-rank data is given in Algorithm 5.8.

The observations excluded from the final subset are nominated as outliers. The distances $d_i(\bar{\boldsymbol{x}}_b, \, \mathbf{S}_b)$ at the final step can be used as robust distances. Furthermore, the mean and covariance matrix of the final basic subset, $\bar{\boldsymbol{x}}_b$ and $\mathbf{S}_b$, can be viewed as robust estimators of location and scale, respectively.

Algorithm 5.8 BACON algorithm for Full-Rank Data

Input: A full-rank matrix $\boldsymbol{X}_{n\times p}$ of multivariate data.

Step 1. Select the initial basic subset $\boldsymbol{X}_b$ of size $r > p$ using (5.3) or (5.4).
Step 2. Compute the distances $d_i(\bar{\boldsymbol{x}}_b, \mathbf{S}_b)$ according to (5.6).
Step 3. Set the new basic subset to all points satisfying (5.7).
Step 4. Iterate Steps 2 and 3 until the size of the basic subset no longer changes.
Step 5. Nominate the observations excluded from the basic subset as outliers.

Output: A set of observations nominated as outliers, if any.
Robust estimates of the location and scale $\bar{\boldsymbol{x}}_b$ and $\mathbf{S}_b$, respectively.

As can be seen from (5.6), the BACON distances assume that the covariance matrix $\mathbf{S}_b$ is non-singular, or equivalently, that the subset $\boldsymbol{X}_b$ is of rank p, a

condition which will not hold when $p > n$. Note that the matrix $\mathbf{S}_b$ could be singular even when $n > p$. Of course, if $p > n$, then all subsets of the data are rank deficient and hence the corresponding $\mathbf{S}_b$ matrix is singular. Thus, when the matrix $\mathbf{S}_b$ is singular, the BACON algorithm can not be used. Paragraph 5.1.2 extends the BACON approach to cases where $\mathbf{S}_b$ is singular.

5.1.2 The BACON Approach for Rank-Deficient Data

In the rank deficient case the matrix $\mathbf{S}$ as well as all subset matrices $\mathbf{S}_b$ are not invertible. Therefore, the initial distances in (5.3) as well as the distances in (5.6) can not be computed. Moreover, the critical values in (5.7) are not appropriate because the distribution for the distances in (5.6) whenever they are computable is unknown and a χ^2 cut-off value should be reviewed. The BACON algorithm can be extended to deal with rank-deficient data in two ways. In the first case, which we refer to as the RD1-BACON, is based on applying the BACON algorithm on a subset of robust scores, denoted by $\widetilde{\boldsymbol{x}}_1, \ldots, \widetilde{\boldsymbol{x}}_k$ with number of score vectors k being much less than p. This is done using the eigen decomposition of a fast to compute initial robust scatter estimate. The second case, which we refer to as the RD2-BACON, is based on shrinkage or ridge-type regularization of the matrices $\mathbf{S}$ and $\mathbf{S}_b$ by adding a positive constant δ to its diagonal in order to solve singularities and to recover the robust distances. The details of these two alternatives are given below.

RD1-BACON

The RD1-BACON algorithm looks for robust directions on which the observations are then projected. These robust directions are derived from the spatial sign covariance matrix given by

$$\mathbf{C} = \frac{1}{n} \sum_{i=1}^{n} S(\boldsymbol{x}_i - \mathbf{m})\, S(\boldsymbol{x}_i - \mathbf{m})', \tag{5.11}$$

with $\mathbf{m}$ denoting the multivariate L_1 median or the spatial median (see Hössjer and Croux, 1995, for theoretical properties and computation). Let in general $\mathbf{z} = \boldsymbol{x} - \mathbf{m}$, then $S(\mathbf{z})$ denotes the spatial sign vector computed according to

$$S(\mathbf{z}) = \begin{cases} \frac{\mathbf{z}}{\|\mathbf{z}\|} & \text{for } \mathbf{z} \neq \mathbf{0}, \\ \mathbf{0} & \text{for } \mathbf{z} = \mathbf{0}, \end{cases} \tag{5.12}$$

with $\mathbf{0}$ denoting the zero vector. For the spatial sign covariance matrix one can see Visuri et al. (2000) and Locantore et al. (1999).

Let $\widetilde{\lambda}_1 \geq \widetilde{\lambda}_2 \geq \cdots \geq \widetilde{\lambda}_p$ be the eigenvalues of $\mathbf{C}$ and let $\widetilde{\mathbf{V}}$ be the corresponding matrix of eigenvectors. Then, we have $\mathbf{C} = \widetilde{\mathbf{V}}\widetilde{\boldsymbol{\Lambda}}\widetilde{\mathbf{V}}'$, where $\widetilde{\boldsymbol{\Lambda}}$ is a diagonal matrix containing the eigenvalues on its diagonal. We compute the $n \times k$ robust scores matrix as follows

$$\widetilde{\boldsymbol{X}}_k = (\boldsymbol{X} - \mathbf{m})\, \widetilde{\mathbf{V}}_k, \tag{5.13}$$

with $\mathbf{m}$ denoting the robust location estimate in equation (5.11), and $\widetilde{\mathbf{V}}_k$ denoting the matrix containing the first k vectors of $\widetilde{\mathbf{V}}$. Note that $k << p$ and $\widetilde{\boldsymbol{X}}_k$ contains orthogonal columns.

The ordinary BACON Algorithm 5.8 is then applied to $\widetilde{\boldsymbol{X}}_k$ in (5.13). The RD1-BACON initially removes the directions with zero robust scales after the inspection of the eigenvalues of the positive semi-definite matrix $\mathbf{C}$. In this way the value for k is defined as the number for which $(\widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_k)$ sufficiently reduce the dimension of the problem, with $\widetilde{\lambda}_j$ being a non zero eigenvalue. A scree plot of the eigenvalues $\widetilde{\lambda}_j$ may be equally used to detect the number for k. The RD1-BACON retains the iterative scheme of the ordinary BACON algorithm, while it uses the orthogonal matrix $\widetilde{\boldsymbol{X}}_k$ to run iterations. Note that the squared distances computed by the BACON algorithm inside the score subsets are now given as

$$d_i = \sqrt{(\boldsymbol{x}_i - \mathbf{m})' \, \widetilde{\mathbf{V}}_k \widetilde{\boldsymbol{\Lambda}}_k^{-1} \widetilde{\mathbf{V}}_k' \, (\boldsymbol{x}_i - \mathbf{m})}, \tag{5.14}$$

with the subscript k indicating eigenvector and eigenvalues matrices of order k. Because the scores vectors are approximately orthogonal and since their variance is equal to $\widetilde{\lambda}_j$ for $j = 1, \ldots, k$, the distances in (5.14) are easily computed according to

$$d_i = d(\widetilde{\boldsymbol{x}}_i) \approx \sqrt{\sum_{j=1}^{k} \frac{\widetilde{x}_{ij}^2}{\widetilde{\lambda}_j}}. \tag{5.15}$$

The subscript b should be added in the notation above since the distances are computed for each subset b in the BACON algorithm.

In order to classify the observations on outliers and clean observations, the BACON algorithm uses the cut-off value given by the expression in (5.7). Note that now k replaces p. The use of a chi-square cut-off value follows directly by the approximately normal distribution of the score vectors. Yet, the dimension of p being very large, and often much larger than n, it modifies the decision rule as follows

$$d_i \; < c_{nkr} \, \chi_{k,\, \alpha/\, \max(p,n)}. \tag{5.16}$$

By $\chi^2_{k,\,\alpha}$ we denote the $(1-\alpha)$ quantile of a chi-square distribution on k degrees of freedom. The use of $\max(p, n)$ instead of n (which is used for the full rank BACON) can be seen as penalizing the dimensionality of the data, and the critical value will be larger for large p, especially for $p >> n$. Note also that the smaller k, the retained columns of the scores $\widetilde{\boldsymbol{X}}$, the larger the initial selection parameter c should be set in the ordinary BACON. Small values for both k and c will restrict the initial basic subset to a very small number of r observations. We recall that in the full rank case the BACON algorithm sets $r = c \cdot p$. The RD1-BACON replaces p by k. In such cases, in order to make the algorithm more effective one should make a good choice for k and c.

Finally, it is worth to notice that the RD1-BACON has additionally a projection aspect. This is due to the fact that $S(\boldsymbol{x}_i - \mathbf{m})$ is the projection of the

p-dimensional vector $\boldsymbol{x}_i$ onto the unit sphere centered at $\mathbf{m}$ (see Locantore et al., 1999). The matrix $\mathbf{C}$ is robust. Yet, it is not very efficient (see discussion in Locantore et al., 1999). The RD1-BACON is normally expected to improve the efficiency of the final estimate for a relatively small computational effort. The RD1-BACON algorithm is given in Algorithm 5.9.

Algorithm 5.9 RD1-BACON for Rank-Deficient Data

Input: A rank-deficient matrix $\boldsymbol{X}_{n\times p}$ of multivariate data.

Step 1. Compute the robust scores matrix $\widetilde{\boldsymbol{X}}_k$ according to (5.13).
Step 2. Apply the ordinary BACON Algorithm 5.8 to $\widetilde{\boldsymbol{X}}_k$ by modifying the critical value to the one given in expression (5.16).

Output: A set of observations nominated as outliers, if any.
Robust estimates of the location and scale $\bar{\boldsymbol{x}}_b$ and $\mathbf{S}_b$, respectively.

RD2-BACON

The RD2-BACON algorithm is essentially based on a ridge type regularization in the covariance matrix. The spectral decomposition of the covariance matrix $\mathbf{S} = \boldsymbol{X}'\boldsymbol{X}$ is computed as

$$\mathbf{S} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}', \tag{5.17}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigen values of $\mathbf{S}$ and $\mathbf{V}$ is the corresponding orthonormal matrix of eigenvectors. Then the matrix in (5.17) is replaced by

$$\mathbf{S}^\star = \mathbf{V}\boldsymbol{\Lambda}^\star\mathbf{V}', \tag{5.18}$$

where $\boldsymbol{\Lambda}^\star = \boldsymbol{\Lambda} + \delta\mathbf{I}_p$ with δ a positive constant and $\mathbf{I}_p$ is the identity matrix of order p. Accordingly, the Mahalanobis distances in (5.3) can be replaced by

$$d_i(\bar{\boldsymbol{x}}, \mathbf{S}^\star) = \sqrt{(\boldsymbol{x}_i - \bar{\boldsymbol{x}})' \left(\mathbf{S}^\star\right)^{-1} (\boldsymbol{x}_i - \bar{\boldsymbol{x}})}, \quad i = 1, \ldots, n, \tag{5.19}$$

The RD2-BACON algorithm starts by ordering the observations according to their distances in (5.4) or (5.19). The subset of $r << p$ with the smallest distances forms the initial basic subset. The spectral decomposition of $\mathbf{S}_b$ is computed and the distances in (5.6) is replaced by

$$d_i(\bar{\boldsymbol{x}}_b, \mathbf{S}_b^\star) = \sqrt{(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_b)' \left(\mathbf{S}_b^\star\right)^{-1} (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_b)}, \quad i = 1, \ldots, n, \tag{5.20}$$

where $\mathbf{S}_b^\star = \mathbf{V}_b\boldsymbol{\Lambda}_b^\star\mathbf{V}_b'$ and $\boldsymbol{\Lambda}_b^\star = \boldsymbol{\Lambda}_b + \delta\mathbf{I}_p$. The iterations of the ordinary BACON algorithm continues until the basic subset no longer changes. However, since the critical values in (5.7) are no longer suitable, we select the basic subset to include all observations satisfying

$$d_i(\mathbf{S}^\star) \leq \mathsf{med}(d(\mathbf{S}^\star)) + t_\alpha \, \mathsf{IQR}\left(d(\mathbf{S}^\star)\right), \tag{5.21}$$

where t_α is a suitable constant chosen to control the desired null size α (see paragraph 5.1.2), and $\mathsf{med}(d(\mathbf{S}^\star))$ and $\mathsf{IQR}(d(\mathbf{S}^\star))$ are the median and the interquartile range, respectively, of the distances in $d(\mathbf{S}^\star) = \{d_1(\mathbf{S}^\star), d_2(\mathbf{S}^\star), \ldots, d_n(\mathbf{S}^\star)\}$. This nonparametric cutoff value requires no knowledge of the distribution of $d_i(\mathbf{S}^\star)$, which is unknown and difficult to derive. This makes RD2-BACON suitable when no assumptions on multivariate normality or elliptical symmetry of the data can be made. This is the case in many applications of high dimensional data sets. The subsets of RD2-BACON algorithm grow rapidly and according to our experience just a few iterations are sufficient to reach the final subset.

Note that when $p > n$, the computational burden due to the large p is solved by working on the observations space. That is, instead of computing the eigen decomposition of the matrix $\boldsymbol{X}'\boldsymbol{X}$, which is $p \times p$, one can compute the eigen decomposition of the much smaller (in dimension) matrix

$$\boldsymbol{X}\boldsymbol{X}' = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}', \tag{5.22}$$

which is of order $n \times n$, where the matrix $\boldsymbol{\Lambda}$ in (5.17) and (5.22) have identical diagonal elements up to the n^{th} position, and $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_p)$ is an $n \times p$ orthonormal matrix with columns corresponding to the eigenvectors of $\boldsymbol{X}\boldsymbol{X}'$. The eigenvectors $\mathbf{V}$ are then obtained according to

$$\mathbf{V} = \boldsymbol{X}'\mathbf{U}\,\boldsymbol{\Lambda}^{-1/2}. \tag{5.23}$$

The RD2-BACON algorithm is given in Algorithm 5.10.

Algorithm 5.10 RD2-BACON for Rank-Deficient Data

Input: A rank-deficient matrix $\boldsymbol{X}_{n\times p}$ of multivariate data.

Step 1. Select the initial basic subset $\boldsymbol{X}_b$ of size $r << p$ using (5.4) or (5.19).
Step 2. Compute the distances $d_i(\bar{\boldsymbol{x}}_b, \mathbf{S}_b^\star)$ according to (5.20).
Step 3. Set the new basic subset to all points satisfying (5.21).
Step 4. Iterate Steps 2 and 3 until the size of the basic subset no longer changes.
Step 5. Nominate the observations excluded from the basic subset as outliers.

Output: A set of observations nominated as outliers, if any.
Robust estimates of the location and scale $\bar{\mathbf{x}}_b$ and $\mathbf{S}_b$, respectively.

Note that the matrix $\mathbf{S}^\star$ in (5.18) corresponds to $(\mathbf{S} + \delta\mathbf{I}_p)$ and hence the distances in (5.19) can be thought of as the leverage of the ith observation and can be equally written in terms of eigenvalues λ_j as

$$h_i^\delta = \sum_{j=1}^{p} \frac{\lambda_j}{\lambda_j + \delta}\,\mathbf{u}_{ij}, \tag{5.24}$$

with the superscript δ emphasizing regularization, where u_{ij} is the ij-th element of the matrix $\mathbf{U}$ in (5.22). Indeed for $\delta = 0$ the common leverage point is obtained (the least squares leverage). For $\delta > 0$, h_i^{δ} decreases monotonically. Yet, the rate of the decrement depends on the position of observation i on a subspace which is commonly expressed in terms of score vectors (see Walker and Birch, 1988). The value for δ is related to the strength of the collinearity in the data set as follows. Let the jth root condition index of the matrix $\boldsymbol{X}\boldsymbol{X}'$ for $n < p$ or for the matrix $\boldsymbol{X}'\boldsymbol{X}$ for $n > p$ be defined as

$$\kappa_j = \sqrt{\frac{\lambda_1}{\lambda_j}}, \quad j = 1, 2, \ldots, p. \tag{5.25}$$

The values for κ_j grow constantly for $j > 1$. The rate of the growth depends on the collinearity induced in the data set. The higher the collinearity in the data set the faster the index κ_j grows. No collinearity results in a very slow increment for κ_j throughout the whole range of j except the very last values of the latter, that is for $j \approx p$. Intermediate situations naturally demand careful inspection for the index κ_j. When no strong collinearity appears we choose δ to be the value λ_j for which $1 < \kappa_j < 10$. For collinear data we choose δ to be the value λ_j for which κ_j does not exceed large values such as 50 or 100. The latter is especially used for high dimensional data with very strong collinearity. This is the case for example for the Near Infra-Red data sets where variables represent digitized spectra or other functional data applications. The above choice for δ are justified in Belsley (1991). In his study of collinearity he noted that for a maximum for the condition index between 5, 10, and 30 collinearity is not a very serious problem, while a maximum larger than 1000 indicates severe collinearity problems. Given that κ_j in expression (5.25) is the root condition index we detect δ around the spectrum where κ_j does not exceed values such as 50 or 100.

Note, finally, that the RD2-BACON algorithm is a natural generalization of the ordinary multivariate BACON algorithm. To understand this one should take into account the case where $n \leq p$ and no collinearity appears on the predictors. The ordinary multivariate BACON will not be able to give an inverse for matrix $\mathbf{S}$ and the final distances d_i. Therefore the RD2-BACON adds a constant δ (as described above) in order to get $\mathbf{S}^{\star}$ and $d_i(\mathbf{S}^{\star})$. When no collinearity is present the $d_i(\mathbf{S}^{\star})$ are just proportional to a constant $\times$ $d_i(\mathbf{S})$ and almost any value of λ_j can play the role of δ.

Defining t_α

The value of t_α is a suitably chosen constant that controls the desired null size α. This corresponds to the non contaminated data cases and controls the probability of falsely detected outliers. In order to settle its value we run simulations on non contaminated data sets. The simulation setting is the one that will be described in 5.4.1 with zero contamination.

We store the number of times in which the RD2-BACON algorithms falsely detected outliers on clean data sets. These values are then divided by $M = 1000$ which is the total number of the generated data, and they are given in Table 5.1. We slightly tune the parameter t_α which obviously depends on the dimensionality of the data. It turns out that t_α strongly depends on p, which makes it very important in rank deficient case. If the data were normally distributed and the number of observations n was sufficiently larger than p the value for t_α would be around 1.65. Departures from normality and for large p, the value for t_α increases. For high dimensional data it may be set equal to 2.

Table 5.1: Simulation Results: Null size results for the RD2-BACON (RD2).

p	t_α	RD2
50	1.8	0.971
50	2.0	0.989
100	1.8	0.937
100	2.0	0.951
250	2.0	0.961
250	2.1	0.976
500	2.0	0.960
500	2.2	0.978

5.2 The BACON PLS regression

The BACON PLS regression (BPLSR) uses a robust scatter estimate for the variance-covariance matrix $\boldsymbol{\Sigma}$ and runs the Helland implementation for PLS regression. For the ill-conditioned case the robust scatter estimate is directly obtained by the multivariate BACON algorithm presented in paragraph 5.1.1. For the rank deficient case, the BACON algorithm detects the outliers as described in paragraph 5.1.2. The BACON PLS regression then runs the PLS regression without the detected outliers. Note that for rank deficient data the BACON is essentially used as an outlier detection tool.

In both cases the proposed algorithm follows the Helland (1988) algorithm for PLS regression, see Algorithm 3.5. It computes weights $\boldsymbol{w}_k$ and regression coefficients, denoted by $\widehat{\boldsymbol{\beta}}_k^{\text{BPLS}}$, according to

$$\boldsymbol{w}_k \propto \begin{cases} \boldsymbol{X}_b'\boldsymbol{y}_b, & \text{for } k = 1, \\ \boldsymbol{X}_b'\boldsymbol{y}_b - \boldsymbol{X}_b'\boldsymbol{X}_b\,\boldsymbol{W}_{k-1}(\boldsymbol{W}_{k-1}'\,\boldsymbol{X}_b'\boldsymbol{X}_b\,\boldsymbol{W}_{k-1})^{-1}\,\boldsymbol{W}_{k-1}'\,\boldsymbol{X}_b'\boldsymbol{y}_b, & \text{for } k > 1. \end{cases} \tag{5.26}$$

and

$$\widehat{\boldsymbol{\beta}}_k^{\text{BPLS}} = \boldsymbol{W}_k\,(\boldsymbol{W}_k'\,\boldsymbol{X}_b'\boldsymbol{X}_b\,\boldsymbol{W}_k)^{-1}\,\boldsymbol{W}_k'\,\boldsymbol{X}_b'\boldsymbol{y}_b. \tag{5.27}$$

By the subscript b we denote as usual the final subset in the multivariate BACON algorithm. The BACON PLS regression algorithm is given in Algorithm 5.11.

Algorithm 5.11 The BACON PLS regression algorithm

Input: $\boldsymbol{X}; \boldsymbol{y}$;
  $k \leftarrow 1$;
  while *a certain model selection criterion is not fulfilled* **do**
    Step 1. Run the multivariate BACON algorithm on the data set $(\boldsymbol{X}, \boldsymbol{y})$ and obtain the robust covariance estimate $\mathbf{S}_b$.
    Step 2. Compute the loadings $\boldsymbol{w}_k$ according to expression (5.26).
    Step 3. Store loading vectors $\boldsymbol{w}_k$ into matrix $\boldsymbol{W}_k$.
    Step 4. Recover the implied regression coefficients according to expression (5.27).
    $k \leftarrow k+1$;
  end while
Output: Give the final regression model $\widehat{\boldsymbol{y}}_k^{\mathrm{BPLS}} = \boldsymbol{X}\,\widehat{\boldsymbol{\beta}}_k^{\mathrm{BPLS}}$.

The BACON PLS regression follows the Helland algorithm for PLS rather than the NIPALS. It avoids extracting score vectors by constructing robust orthogonal bases directly for the final regression coefficient vector. These are built by $\boldsymbol{w}_k$ according to the recurrence

$$\boldsymbol{w}_k \propto \begin{cases} \mathsf{b}^\star, & \text{for } k = 1, \\ \mathsf{b}^\star - \mathsf{A}^\star \boldsymbol{W}_{k-1} (\boldsymbol{W}_{k-1}' \, \mathsf{A}^\star \boldsymbol{W}_{k-1})^{-1} \boldsymbol{W}_{k-1}' \, \mathsf{b}^\star, & \text{for } k > 1. \end{cases}$$

for a robust $\mathsf{b}^\star = \boldsymbol{X}_b' \boldsymbol{y}_b$ and $\mathsf{A}^\star = \boldsymbol{X}_b' \boldsymbol{X}_b$. Finally the interpretation of the BPLS regression vector is simple and straightforward. This is given in the following remark.

Remark 5.1

The Bacon PLS regression coefficient vector solves the following optimization problem:

$$\widehat{\boldsymbol{\beta}}_k^{\mathrm{BPLS}} = \arg \min_{\boldsymbol{\beta} \in \mathcal{K}_k(\mathsf{b}^\star, \mathsf{A}^\star)} \|\boldsymbol{y} - \boldsymbol{X}\,\boldsymbol{\beta}\|. \tag{5.28}$$

The BACON PC regression

The strong relation between PLS and PC regression has been elucidated throughout Chapters 2 and 3. Here we introduce a robust PC regression method based on the BACON algorithm. That is, we use the robust scatter estimate on the joint data $(\boldsymbol{X}, \boldsymbol{y})$ and run the ordinary PC regression on the latter. Note that in contrast to PLS regression the PC estimate is linear on $\boldsymbol{y}$, the response vector does not enter the maximization problem. Therefore, another approach for the BACON PC regression (BPCR) would be to run the multivariate BACON

only on $\boldsymbol{X}$ and then introduce robust BACON regression on $\boldsymbol{y}$. In the real and experimental data that follow we additionally provide results for the PC and the BACON PC regression. The latter is sketched in Algorithm 5.12.

Algorithm 5.12 The BACON PC regression algorithm

Input: $\boldsymbol{X}; \boldsymbol{y}$;

For $k = 1, ..., p$,

Step 1. Run the multivariate BACON algorithm on the matrix $(\boldsymbol{X}, \boldsymbol{y})$ and obtain the robust covariance matrix $\mathbf{S}_b$.

Step 2. Compute loading matrix $\boldsymbol{W}_k$ according to

$$\boldsymbol{W}_k = (\boldsymbol{w}_1, ..., \boldsymbol{w}_k),$$

where $\boldsymbol{w}_j$ is the eigenvector corresponding to the j^{th} eigenvalue of $\boldsymbol{X}_b'\boldsymbol{X}_b$.

Step 3. Recover the implied robust regression coefficients

$$\widehat{\boldsymbol{\beta}}_k^{\text{BPCR}} = (\boldsymbol{W}_k' \boldsymbol{X}_b' \boldsymbol{X}_b \boldsymbol{W}_k)^{-1} \boldsymbol{W}_k' \boldsymbol{X}_b' \boldsymbol{y}_b.$$

Output: Select k using model selection and give the final regression model $\widehat{\boldsymbol{y}}_k^{\text{BPCR}} = \boldsymbol{X}\, \widehat{\boldsymbol{\beta}}_k^{\text{BPCR}}$.

5.3 Illustrative examples

5.3.1 The NY Rivers data

The New York Rivers data result from a study relating water quality to four land-use variables for 20 rivers located in New York State. The response variable, $\boldsymbol{y}$, is the water quality measured as the mean nitrogen concentration (mg/liter). The four predictor variables are: the percentage of land area used in agriculture, X_1, the percentage of land area in commercial or industrial use, X_2, the percentage of forest land, X_3, and the percentage of land area in residential use, X_4. The data have been previously analyzed in Hadi (1992a).

The correlation matrix of the explanatory variables (not shown here) indicates high correlations among the variables which naturally causes collinearity problems. Furthermore, Hadi (1992a) reports that River 5 is a high leverage point (outlier in the X-space), River 7 is an outlier in the Y-space, while rivers 3 and 4 are a combination of both.

We run PC, PLS, BPC and BPLS regression algorithms. The left panel in Figure 5.1 illustrates the projection of the 20 observations on the space generated by the first two PLS regression and BPLS regression scores. The circles represent the PLS regression scores, the crosses represent the BPLS regression scores.

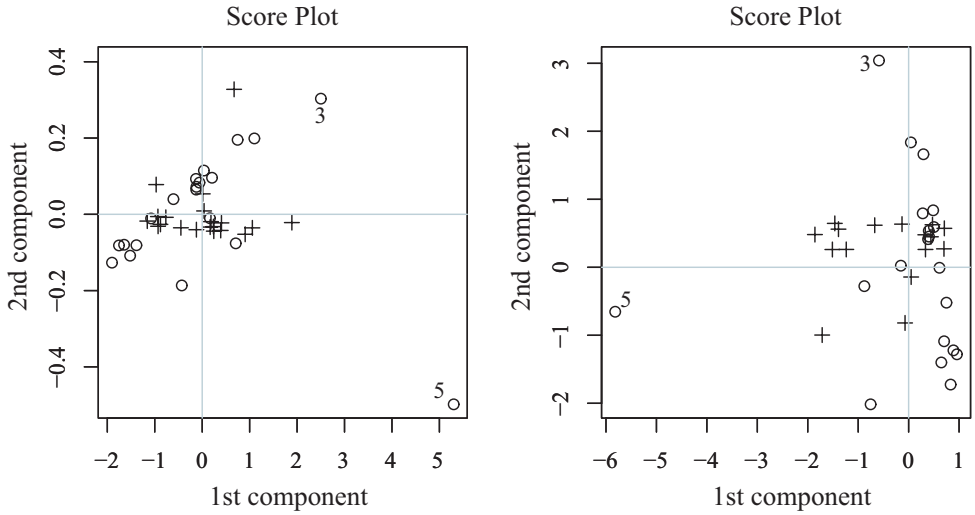


Figure 5.1: New York Rivers Data. (a) Left panel: Score plot for both PLSR and BPLSR methods. (b) Right panel: Score plot for both PCR and BPCR methods. Circles (∘) correspond to PLSR and PCR scores, while crosses (+) are used to indicate the BPCR and BPLSR scores

It can be seen from Figure 5.1 that the BPLS regression scores for cases 3, 4, 5 and 7 lie close to the scores for the rest of the data. In contrast the projection of PLS regression scores on the first two components clearly overscores observations 3 and 5 which clearly influence the extracted components. The same pattern holds for the PC and BPC regression score plot which is shown in the right panel in Figure 5.1.

The loadings $\boldsymbol{w}_1$ resulting from all four methods are recorded in Table 5.2. They reflect how strongly the predictors are represented in the derived components. Comparisons between PC and BPC regression loadings as well as between PLS and BPLS regression loadings indicate differences in both magnitudes and signs. Notably we mention the decrease for the loadings related to the third and the fourth variable for both PC and PLS regression methods. This is mainly due to the outlying position of observations 3 and 5 on the space spanned by X_3 and X_4.

Table 5.2: New York Rivers Data. PCR, BPCR, PLSR and BPLSR loadings

	PCR	BPCR	PLSR	BPLSR
X_1	0.0869	0.7528	0.3433	0.6949
X_2	0.4041	–0.6570	–0.6624	–0.7143
X_3	–0.6608	0.0323	0.4851	0.0701
X_4	–0.6264	–0.0215	0.4560	0.0435

Finally we compute, for all four regression models, the root mean squared error of prediction which is given by

$$\text{RMSE}(k) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_{ik})^2}, \tag{5.29}$$

with $\widehat{\boldsymbol{y}}_k$ corresponding to the fitted value for a model containing k components. Table 5.3 reports the values for the RMSE_k for PC, BPC, PLS, and BPLS regression, respectively. Both BPC and BPLS regression methods reduce the prediction loss in comparison to PLS and PC regression. This is the case for all $k = 1, \ldots, 4$.

Table 5.3: New York Rivers Data. RMSE_k for PCR, BPCR, PLSR, and BPLSR methods

	$k=1$	$k=2$	$k=3$	$k=4$
PCR	0.7221	0.5486	0.5277	0.5254
BPCR	0.2172	0.2182	0.2167	0.2108
PLSR	0.5533	0.5425	0.5269	0.5254
BPLSR	0.3452	0.4124	0.3937	0.4028

5.3.2 The Octane data

The Octane data set consists of 39 gasoline samples for which the octanes have been measured at 225 wavelengths, thus resulting in a 39×225 data matrix. The resulting spectra are illustrated in Figure 5.2. Around the end of the spectrum it is already observed an abnormal group of octane samples which contain a lot of alcohol.

The Octane data are suitable for illustrating the use of BPLS and BPC regression in the rank deficient case. Therefore, we run here the regression methods using the RD1-BACON algorithm given in Algorithm 5.9. The multivariate BACON is run on scores derived by the spectral decomposition of the scatter estimate in expression (5.11) with only six score vectors retained. Hence, BPLS and BPC regression algorithms are executed on the reduced set $\widetilde{\boldsymbol{X}}_{(39\times 6)}$ by using the zero weights for the detected outliers. This means that only the observations belonging to the final subset are taken into account.

We set $c = 3$ for the parameter c of the multivariate BACON resulting to an initial subset of $3 \times 6 = 18$ observations. The RD1-BACON algorithm converged in 4 iterations. Observations $25, 26$ and $36, 37, 38, 39$ are detected as outliers. The same results are obtained by taking $\widetilde{\boldsymbol{X}}_{(39\times 3)}$ and increasing c to 5; the algorithm converges faster, on 3 iterations.

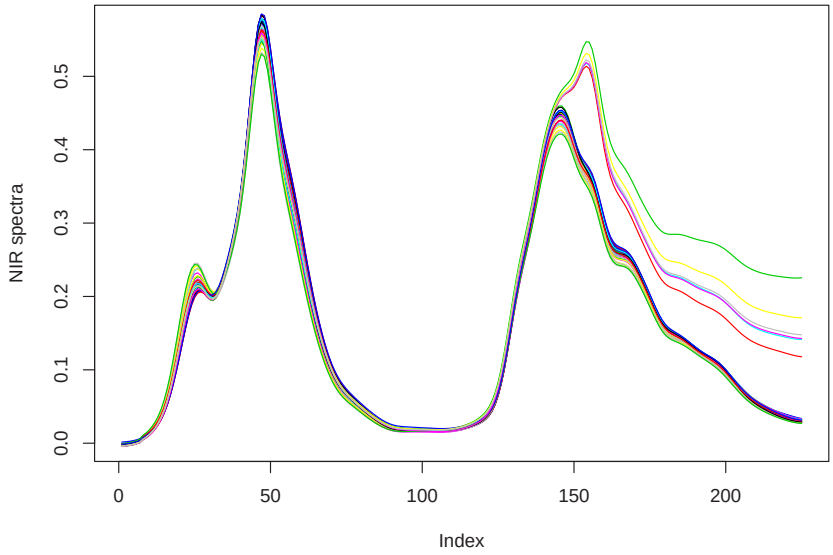


Figure 5.2: Octane Data. Near Infra-Red spectra for 39 gasoline samples for which the octanes have been measured at 225 wavelengths.

We are interested here in defining the dimension of the final regression model. Given the dimensionality of the problem we seek to reduce dimension and regularize the regression problem using PLS and PC regression. Fixing the number of components in the final regression model is done by means of cross validation (see Denham, 2000). In particular, we split the data into five equal parts and we run the regression methods five times. Each time we leave out one part (see Hastie et al., 2004, Chapter 7). The values for the RMSE statistic is then averaged over the five splits. The resulting values for all four regression methods are plotted in Figure 5.3.

Both panels in Figure 5.3 suggest two components for the BPLS and BPC regression final models. Ordinary PLS and PC regression need one more component since they are both influenced by the group of six outlying observations (25, 26, 36, 37, 38, and 39). These cases were detected as outliers by the BACON algorithm. Our results are in accordance with a previous study on the Octane data (see Engelen et al., 2004) and with what it has been already observed in Figure 5.2. Outliers can seriously affect dimension reduction and model selection. In paragraph 5.4.2 we give an extensive simulation study based on the bilinear factor model where we compare model selection for the ordinary and the BACON PLS regression.

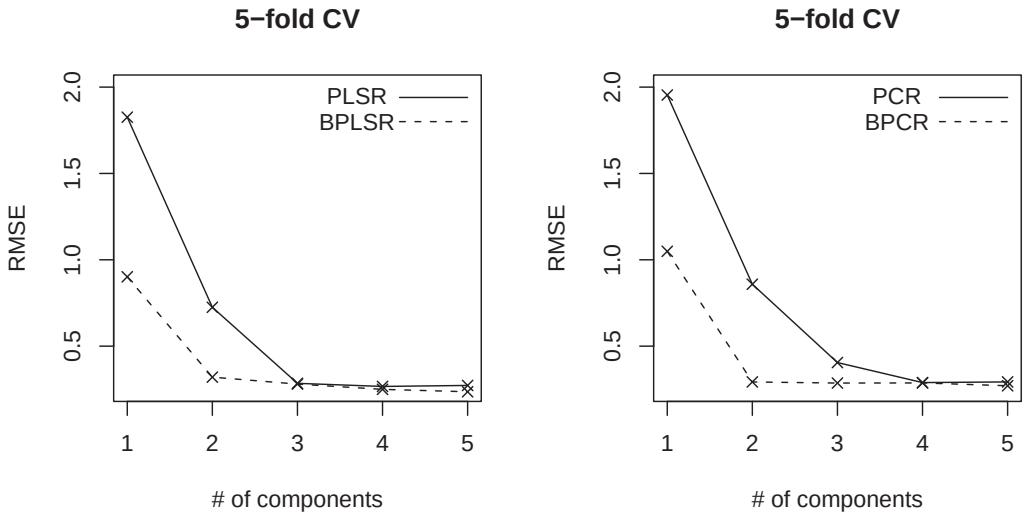


Figure 5.3: Octane Data. Left Panel: RMSE(k) for PLSR and BPLSR. Right Panel: RMSE(k) for PCR and BPCR methods. The value for k is taken over $1, \ldots, 5$.

5.4 Experimental data

Outliers have negative effects in estimating location, scatter, and regression parameters. In high dimensional data sets outliers still remain a problem. Yet, high dimensional data are themselves a serious drawback; the ordinary variance-covariance matrix for example is no more positive definite. As seen in the Octane data, outliers may have a negative effect in model selection and assessment. For PLS regression this corresponds to the reduction of the dimension given by the final number of components in the PLS regression model and its expected prediction loss.

In what follows we give two extensive simulation studies in paragraphs 5.4.1 and 5.4.2. These cover both the low and the high dimensional setting. Furthermore, in the simulation setting in paragraph 5.4.2 the real dimension of the PLS regression model is known in advance. Therefore it is suitable to explore dimension reduction and outliers effects on model selection. Throughout the simulation studies we use the mean squared error for the regression parameter $\widehat{\boldsymbol{\beta}}$ of a model including k components, denoted as $\mathrm{MSE}_k(\widehat{\boldsymbol{\beta}})$, in order to investigate the efficiency of the proposed algorithms. The mean squared error is given by

$$\mathrm{MSE}_k(\widehat{\boldsymbol{\beta}}) = \mathsf{E}_M[(\widehat{\boldsymbol{\beta}}_k^m - \boldsymbol{\beta})'(\widehat{\boldsymbol{\beta}}_k^m - \boldsymbol{\beta})], \tag{5.30}$$

with $\widehat{\boldsymbol{\beta}}_k^m$ denoting the estimated coefficient vector in the m^{th} simulation for a model including k components, and $\boldsymbol{\beta}$ the true coefficient vector. The subscript M in the expectation indicates that we average over the M simulated data. Fur-

thermore, in order to select the final regression model and to assess its predictive ability we use the root mean squared error of prediction (see expression (5.29)) using cross validation.

5.4.1 Simulating setting - Gunst and Mason

We generated data sets following the simulation study presented in Gunst and Mason (1977). We have taken the following steps:

1. For the matrix $\boldsymbol{X}_{(n\times p)}$ we generated $p^* < p$ columns of random observations according to $x_{ij^*} \sim U(0, 10)$, where $i = 1, \ldots, n$ and $j^* = 1, \ldots, p^*$.

2. For the remaining $d = (p-p^*)$ columns in the matrix $\boldsymbol{X}$ linear dependencies were introduced, inducing d-variable collinearity. This means that for $j = (p^* + 1), \ldots, p$ we generated $\boldsymbol{x}_j = g\,\mathbf{x}_{j^*} + \mathbf{e}$, where $g \sim U(0, 1)$, and error vector $\mathbf{e}$ with elements $e_i \sim N(0, 1)$.

3. We have chosen two directions for the real regression coefficient vector $\boldsymbol{\beta}$. The first lies on the direction of the first and dominant eigenvector $\mathbf{v}_1$ of $\boldsymbol{X}'\boldsymbol{X}$, that is, $\boldsymbol{\beta}_1 = \mathbf{v}_1$. The second vector is given by $\boldsymbol{\beta}_2 = 0.25\,(\mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_4 + \mathbf{v}_5)$, with $\mathbf{v}_j$ denoting the j^{th} eigenvector of $\boldsymbol{X}'\boldsymbol{X}$ as given in (5.17). For both vectors $\boldsymbol{\beta}_1'\boldsymbol{\beta}_1 = \boldsymbol{\beta}_2'\boldsymbol{\beta}_2 = 1$ holds.

4. The signal-to-noise ratio (SNR), generally given by $\boldsymbol{\beta}'\boldsymbol{\beta}/\sigma^2$, is set equal to 10 and 100. These values arise for σ^2 equal to 0.1 and 0.01 respectively.

5. We finally contaminated the data set by randomly replacing $10\%, 15\%$, and 20% of the observations. The contaminant observations are generated according to $x_{ij^*} \sim U(20, 30)$, with collinearity induced in exactly the same way with the non contaminated data. We also run the simulations for the 0% contamination rate in order to test the efficiency of the proposed algorithms on clean data.

Table 5.4: Parameter values in the simulations

n	p	$\boldsymbol{\beta}$	SNR	Contamination Rate
100	10	$\boldsymbol{\beta}_1$	10	0%
	20	$\boldsymbol{\beta}_2$	100	10%
	30			15%
				20%

Table 5.4 summarizes the settings of the experimental data. The value of p^* is set to 5; strong collinearities are therefore induced. For each set of simulation

parameters we generated $M = 1000$ data sets and we computed the corresponding values for RMSE_k and $\text{MSE}_k(\widehat{\boldsymbol{\beta}})$ according to equations (5.29) and (5.30), respectively. Mahalanobis distances were used in the BACON process instead of distances from the medians. Throughout the simulations, the detected outliers are given weights inversely proportional to the distances $d_i(\bar{\boldsymbol{x}}_b, \mathbf{S}_b)$ obtained from the final iteration of Algorithm 5.8. Therefore outliers are not thrown away from the analysis, yet, they are downweighted.

The real dimension of the m^{th} simulated model is not fixed a priori. Therefore we used one random split cross validation in order to fix it; this was done before data contamination. We computed the sample prediction error by means of the RMSE(k) given in expression (5.29), by replacing the total sample size n with the size of the test data, n_{test}. We chose the model for which the sample prediction error is essentially minimized. We denote by $k^\star$ the dimension of the selected model. Computation is now restricted to $k = k^\star$.

Simulation Results

We first compute the mean value of the final dimension ($\bar{k}^\star$) for the PLS and the PC regression models. We found $\bar{k}^\star_{\text{PLSR}} = 2.2$ and $\bar{k}^\star_{\text{PCR}} = 5.2$. These values confirm that PLS regression reduces the regression problem further than PC regression (this is true when no components selection technique is used for PC regression). Moreover, the value of $\bar{k}^\star_{\text{PCR}} = 5.2$ reflects the five-variable collinearity that was induced in the simulation setting.

The $\text{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}})$ and the $\text{RMSE}(k^\star)$ were computed for all parameter combinations. A few outlying values were firstly removed in order to avoid their influence on the simulation results. Alternatively a trimmed MSE can be used (see Serneels et al., 2005b).

Tables 5.5, 5.6, and 5.7 report the relative efficiency of the BPLS regression and the BPC regression estimates. Tables 5.5 and 5.6 report efficiency for $10\%, 15\%$, and 20% contamination levels, while Table 5.7 reports efficiency for clean data. The relative efficiency is computed as the ratio between the MSE of the BACON estimates divided by the MSE of the ordinary estimates. We denote by RE.BPLSR the relative efficiency of the BPLS regression estimate compared to the ordinary PLS regression estimate. Given that both PLS and PC regression estimates are highly sensitive to outliers, a RE.BPLSR value close to zero indicates high performance of the BPLS and the BPC regression estimates relative to PLS and PC regression. In contrast RE.BPLSR close to one shows no improvement by using the BACON algorithm in PC regression and PLS regression. These hold for contaminated data.

The results in Table 5.5 show that BPLS regression relative efficiency reaches low levels. In almost all cases BPLS regression works better than ordinary PLS regression. In more than 75% of the simulated cases the value of the RE.BPLSR is below 0.2. Yet, as contamination and data dimension increase the $\text{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}_{\text{BPLSR}})$ gets closer to the $\text{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}_{\text{PLSR}})$.

Table 5.5: Simulation Results: The relative efficiencies, RE.BPLSR, of the BPLSR

		Contamination Rate					
		10%		15%		20%	
p	SNR	$\boldsymbol{\beta}_1$	$\boldsymbol{\beta}_2$	$\boldsymbol{\beta}_1$	$\boldsymbol{\beta}_2$	$\boldsymbol{\beta}_1$	$\boldsymbol{\beta}_2$
10	10	0.1245	0.0431	0.1022	0.0900	0.1192	0.0555
	100	0.1200	0.0483	0.1192	0.1022	0.1221	0.0556
20	10	0.1787	0.0849	0.1936	0.0918	0.1769	0.0983
	100	0.1780	0.0854	0.1945	0.1114	0.1822	0.0992
30	10	0.2202	0.1126	0.2611	0.1940	0.6885	0.9097
	100	0.2168	0.1128	0.2649	0.1953	0.6846	0.9032

This is verified by means of a four-way analysis of variance on the results given in Table 5.5. The linear model yields $R^2 = 75\%$ with significant parameters the data dimension (p) and the contamination rate. Both estimates provided strong statistical evidence (p-value < 0.01).

Table 5.6: Simulation Results: The relative efficiencies, RE.BPCR, of the BPCR

		Contamination Rate					
		10%		15%		20%	
p	SNR	$\boldsymbol{\beta}_1$	$\boldsymbol{\beta}_2$	$\boldsymbol{\beta}_1$	$\boldsymbol{\beta}_2$	$\boldsymbol{\beta}_1$	$\boldsymbol{\beta}_2$
10	10	0.1096	0.0582	0.1123	0.1125	0.1132	0.0759
	100	0.1126	0.0649	0.1181	0.1008	0.1128	0.0725
20	10	0.2581	0.0972	0.2260	0.1172	0.3431	0.1368
	100	0.2716	0.0983	0.2944	0.1120	0.3395	0.1378
30	10	0.3109	0.1204	0.3240	0.1356	0.8132	0.9009
	100	0.3212	0.1214	0.3555	0.1300	0.8154	0.8913

Similar conclusions are drawn by the inspection of Table 5.6. In all trials BPC regression method works better than ordinary PC regression. The values for the $\mathrm{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}_{\mathrm{BPLSR}})$ are far below $\mathrm{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}_{\mathrm{BPLSR}})$ in most of the cases. Yet, the relative efficiency reaches levels close to one for 20% contamination in high dimension ($p = 30$). The four-way analysis of variance (done on the values of RE.BPCR in Table 5.6) provided highly significant estimates for the dimension p and the contamination rate. Finally, no statistical evidence is found between the relative efficiencies of BPLS and BPC regression. This means that both BPLS and BPC regression perform equally well in robustifying the ordinary PLS and PC regression, respectively.

An important aspect of a robust method is its efficiency in the case where the data are not contaminated. Therefore we included in our simulation study

the case of zero contamination rate. Table 5.7 gives the results for the relative efficiencies RE.BPLSR and RE.BPCR for 0% contamination. It can be seen that, for all different settings in our simulation, relative efficiencies are very close to one. Given the fact that data are clean (0% contamination) relative efficiency close to one proves that both BPLS and BPC regression are highly efficient. This is also the case for large dimensions p.

Table 5.7: Simulation Results: Relative efficiencies (RE.BPLSR and RE.BPCR) for 0% contamination

		RE.BPLSR		RE.BPCR	
p	SNR	$\boldsymbol{\beta}_1$	$\boldsymbol{\beta}_2$	$\boldsymbol{\beta}_1$	$\boldsymbol{\beta}_2$
10	10	0.9995	1.0014	1.0067	1.0239
	100	1.0031	0.9924	1.0194	1.0124
20	10	1.0005	1.0016	1.0106	1.0084
	100	0.9989	0.9978	1.0058	1.0194
30	10	0.9996	0.9997	1.0138	1.0052
	100	0.9965	0.9884	1.0140	1.0011

Finally, we investigated the results concerning the prediction loss; a complete table with our results is not given here, but Figure 5.4 shows the boxplots of the RMSE($k^\star$) for PLS and BPLS regression, on the left panel, and the RMSE($k^\star$) for PC and BPC regression on the right panel.

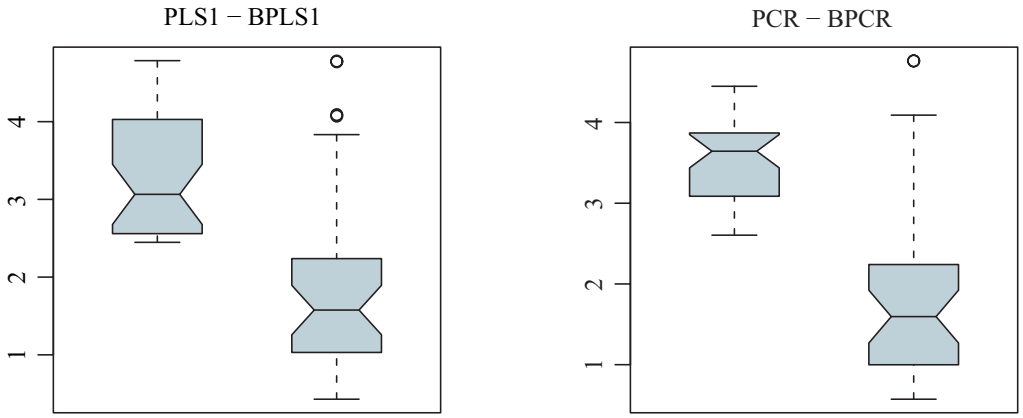


Figure 5.4: Simulated Data. Left Panel: RMSE($k^\star$) for PLSR and BPLSR. Right Panel: RMSE($k^\star$) for PCR and BPCR methods.

BPLS regression provides better predictions for all the simulated data sets. The improvement in the prediction by using BPLS regression instead of PLS regression is about 0.51; this is the mean of the relative prediction error between BPLS and PLS regression. The corresponding value for the PC regression case

is equal to 0.50. It indicates an important gain in prediction by using BPC regression instead of ordinary PC regression. Both panels of Figure 5.4 illustrate that BPLS and BPC regression almost always improve the prediction accuracy of the final model.

Comparison to other robust methods

The BPLS and BPC regression methods are compared here to other robust methods for PLS and PC regression. In particular we compare Algorithm 5.11 to RSIMPLS (Hubert and Branden, 2003) and PRM (Serneels et al., 2005a) regression methods for PLS regression. Algorithm 5.12 is compared to ROBPCR algorithm (see Hubert et al., 2005). The comparisons are based on the same simulation setting as in Table 5.4 (see paragraph 5.4.1), but for fixed signal-to-noise ratio and regression coefficient vector. We used $\boldsymbol{\beta}_2$ as the true coefficient vector and we set the SNR to 10. In fact we retained in our simulations the factors which have been statistically significant, that is, the data dimension and the contamination rate. For the PRM regression we set the threshold for convergence equal to 10^{-2}, while RSIMPLS and ROBPCR used the Minimum Covariance Determinant (MCD) scatter estimate due to the data dimension. We generated 500 simulated data sets for each setting and we computed the relative efficiencies according to the ratio of the MSE between the robust and the ordinary estimate. We denote by RE.BPLSR, RE.RSIMPLS, RE.PRM, RE.BPCR, and RE.ROBPCR the relative efficiencies for the BPLSR, the RSIMPLS, the PRM, the BPCR and the ROBPCR methods. Table 5.8 provides the simulation results for the PLS regression method and Table 5.9 shows the simulation results for the PC regression methods.

Table 5.8 demonstrates the high efficiency of the BPLS regression compared to both the RSIMPLS and PRM methods. Similar conclusions, concerning PC regression methods, are drawn by the inspection of Table 5.9. It is also worth noting that the BPLS regression and the PRM algorithm were much faster in computation than the RSIMPLS methods. The same holds for BPC regression in comparison to ROBPCR for method.

Finally, the RMSE($k^\star$) has been computed for $k^\star$ equal to the dimension indicated by random split cross validation on the PLS and PC regression methods before data contamination (more details are given in paragraph 5.4.1). The final dimensions were in average $\bar{k}^\star_{\text{PLSR}} = 2.7$ for PLS regression and $\bar{k}^\star_{\text{PCR}} = 5.4$ for PC regression. Inspection of the prediction results (not shown here) reveal important gain in prediction by using the BPLS regression and the RSIMPLS instead of the ordinary PLS regression, as well as the BPC regression and the ROBPCR algorithms instead of ordinary PC regression. PRM regression, on the other hand, did not succeed to reduce the prediction loss compared to ordinary PLS regression.

Table 5.8: Simulation Results: The relative efficiencies for all three regression methods, denoted by RE.BPLSR, RE.RSIMPLS, and RE.PRM

		Contamination Rate			
	p	0%	10%	15%	20%
RE.BPLS	10	0.99957	0.10316	0.11165	0.10820
RE.RSIMPLS		1.24180	0.17530	0.13359	0.12392
RE.PRM		1.41042	0.19470	0.24102	0.30033
RE.BPLS	20	1.00439	0.10143	0.09417	0.09261
RE.RSIMPLS		1.50350	0.14109	0.13353	0.11982
RE.PRM		1.93172	0.19397	0.20263	0.26460

Table 5.9: Simulation Results: The relative efficiencies for BPCR and ROBPCR

		Contamination Rate			
	p	0%	10%	15%	20%
RE.BPCR	10	0.99920	0.12126	0.07859	0.09317
RE.ROBPCR		1.16654	0.15677	0.10077	0.10719
RE.BPCR	20	1.01791	0.13813	0.15964	0.146792
RE.ROBPCR		1.64277	0.26650	0.25663	0.22056

5.4.2 Simulating from the bilinear factor model

The bilinear model is described in paragraph 4.4.2 where it has been already used in assessing the PLAD regression. The bilinear model proceeds in the bilinear decomposition given by

$$\boldsymbol{y} = q_1\,\boldsymbol{t}_1 + \ldots + q_k\,\boldsymbol{t}_k + \boldsymbol{\epsilon}, \quad \text{and} \quad \boldsymbol{X} = \boldsymbol{p}_1\,\boldsymbol{t}_1 + \ldots + \boldsymbol{p}_k\,\boldsymbol{t}_k + \boldsymbol{f}. \tag{5.31}$$

By simulating the components $\boldsymbol{T}$ by n independent realizations of

$$\boldsymbol{T} \sim \mathcal{N}_k(\mathbf{0}_k,\, \boldsymbol{\Sigma}_{kk}),$$

we can control the dimension of the PLS regression model as it is given by k. The k-variate normal distribution has mean $\mathbf{0}_k$, since the data are centered, and variance-covariance matrix $\boldsymbol{\Sigma}_{kk}$ given as

$$\boldsymbol{\Sigma}_{kk} = \begin{pmatrix} \mathsf{var}(\boldsymbol{t}_1) & 0 & \ldots & 0 \\ 0 & \mathsf{var}(\boldsymbol{t}_2) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \mathsf{var}(\boldsymbol{t}_k) \end{pmatrix}.$$

The data set $\boldsymbol{X}$ and $\boldsymbol{y}$ are directly obtained according to equation (5.31) for a specific choice for the residual structure in $\boldsymbol{f}$ and $\boldsymbol{\epsilon}$.

Simulation setting - A

In this simulation study we fix $k_{max} = 3$, $\boldsymbol{P} = \boldsymbol{I}_{3,p}$, and $\boldsymbol{q} = (1, 1, 1)'$. Finally the matrix Σ_{kk} is set to $\text{diag}\,(50, 10, 5)$.

$$\boldsymbol{\Sigma}_{kk} = \begin{pmatrix} 50 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 5 \end{pmatrix}.$$

For the generated data we set in the bilinear decomposition above

$$\epsilon_i \sim \mathcal{N}(0, 0.1), \quad \text{for} \quad i = 1, \ldots, n.$$

We then contaminate by replacing a small fraction of the data set with outliers; the contamination rate is 10%. Note that this rate becomes 20% after splitting the data on training and test set as it will be seen shortly. The contaminated error vector is denoted as $\boldsymbol{\epsilon}_{cont}$ and its elements are generated according to

$$\epsilon_{i',cont} \sim \mathcal{N}(\mu, 1) \quad \text{for} \quad \mu = 5 \quad \text{and} \quad i' = 1, \ldots, \ell \quad \text{for} \quad \ell = 0.10 \cdot n.$$

The residual term $\boldsymbol{f}$ similarly is generated by standard normals while it has then been contaminated by a random normal variate as in the Y case, that is

$$f_{i',cont} \sim \mathcal{N}(\mu, 1) \quad \text{for} \quad \mu = 5 \quad \text{and} \quad i' = 1, \ldots, \ell \quad \text{for} \quad \ell = 0.10 \cdot n.$$

The dimension of the generated data sets is set to $n = 100$ and $p = 100$.

We start by generating the non contaminated data as described above, and we compute the out-of-sample prediction loss by means of the cross validated RMSE of prediction with the first half of the data constructing the training set and the second half the test set. Hence, the real dimension of the data where the model is built corresponds to 50×100 and the contamination rate is 20% instead of 10%. The cross validated RMSE of prediction is computed for both PLS and BPLS regression models. The training set is then contaminated; the test set is not contaminated. Both PLS and BPLS regression models are once more constructed and the cross validated RMSE of prediction is computed for both methods. We repeat the process above $M = 1000$ times and we store in matrices of appropriate dimensions the loss for both methods and for both conditions (contaminated data and clean data). Given that the dimension of the model is known, that is $k = 3$, we use the RD1-BACON with 4 retained scores vectors and parameter c set to 5. The choice of 4 score vectors is reasonable since we know that 3 components are sufficient and the chosen subspace is very close to the true dimension. Moreover choosing 4 allows to construct a sufficiently large initial subset in the BACON algorithm for $c = 5$, that is $4 \times 5 = 20$ observations for the initial basic subset.

We repeat our experiment for two more settings described below (denoted as Set2 and Set3). Set1 corresponds to the initial simulation setting described above.

Set1: True contamination rate is equal to 20% in the training set, and $\mathbf{\Sigma}_{kk}$ is set to $\text{diag}\,(50, 10, 5)$. All the other parameters are described in detail above.

Set2: True contamination rate is equal to 10% in the training set, and $\mathbf{\Sigma}_{kk}$ is set to $\text{diag}\,(100, 30, 10)$. All the other parameters are the same with the initial setting.

Set3: True contamination rate is equal to 30% in the training set, and $\mathbf{\Sigma}_{kk}$ is set to $\text{diag}\,(50, 20, 10)$. All the other parameters are the same with the initial setting.

Simulation results - A

For the first simulated setting (Set1) the resulting averaged out-of-sample prediction loss resulting from PLS and BPLS regression on clean and contaminated data are given in Figure 5.5. The PLS regression results are given in a solid red line, while the BPLS regression results are given by the black dashed line. In the first simulation setting the two methods perform in average equally before (left panel) and after (right panel) contamination. Dimension reduction is not affected by contamination of the data. Yet, assessing the final prediction error shows a relatively small loss in prediction accuracy for both regression methods. For the BPLS regression this loss is smaller compared to ordinary PLS regression.

The results for the second simulation setting are given in Figure 5.6. Contamination of the data had a clear negative effect in dimension reduction for ordinary PLS regression. For the contaminated case (right panel) the ordinary PLS regression needs 4 components in the final model. This is not the case for BPLS regression which remains on 3 components for both clean and contaminated data. Yet, ordinary PLS regression by including one more component reduces the prediction loss to levels comparable with the prediction loss level of the non contaminated data.

Finally, we illustrate in Figure 5.7 the results for the third simulation setting (Set3). For this case the BPLS regression once more retains three components after data contamination, in contrast to ordinary PLS regression which is affected by outliers and adds one more component in its final model. Moreover, the level of prediction loss for the BPLS regression on 3 components seems comparable to the prediction loss of ordinary PLS regression even when the last includes 4 components (see the right panel of Figure 5.7). Note also that in the third setting the true rate of contamination was 30% indicating good performance for BPLS regression even in high contamination levels.

Finally, we should emphasize the fact the BPLSR performance on the non contaminated data for all the three settings is almost identical to the PLS regres-

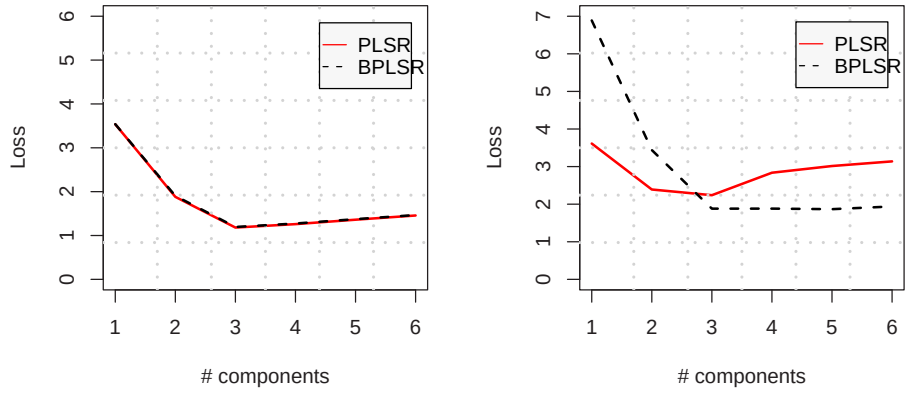


Figure 5.5: Simulating from the Bilinear factor model (Set1). Averaged out-of-sample prediction loss on $M = 1000$ simulated data sets. *Left Panel:* RMSE(k) with $k = 1, \ldots, 6$ for PLSR (red solid line) and BPLSR (black dashed line) before contamination. *Right Panel:* RMSE($k^\star$) with $k = 1, \ldots, 6$ for PLSR (red solid line) and BPLSR (black dashed line) after contamination.

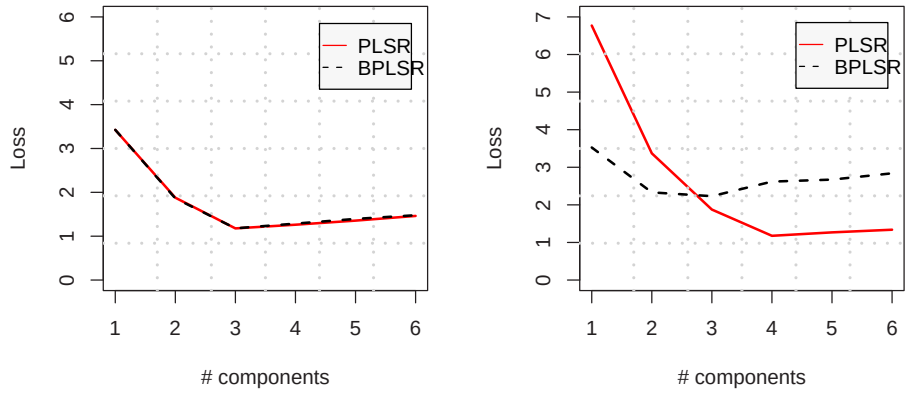


Figure 5.6: Simulating from the Bilinear factor model (Set2). Averaged out-of-sample prediction loss on $M = 1000$ simulated data sets. *Left Panel:* RMSE(k) with $k = 1, \ldots, 6$ for PLSR (red solid line) and BPLSR (black dashed line) before contamination. *Right Panel:* RMSE($k^\star$) with $k = 1, \ldots, 6$ for PLSR (red solid line) and BPLSR (black dashed line) after contamination.

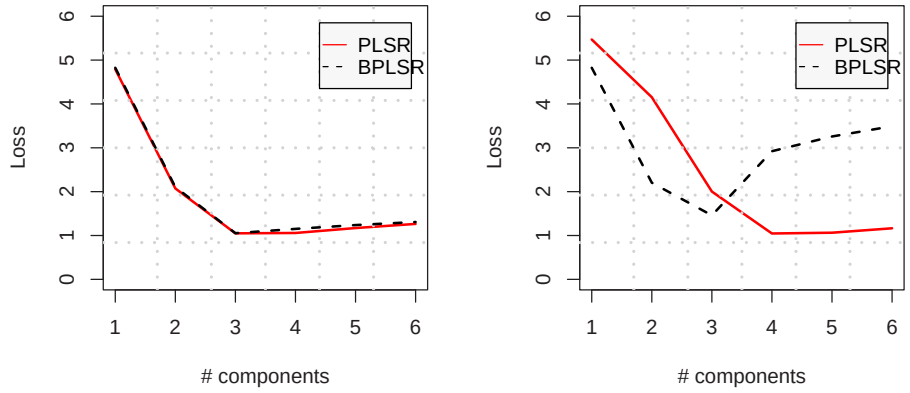


Figure 5.7: Simulating from the Bilinear factor model (Set3). Averaged out-of-sample prediction loss on $M = 1000$ simulated data sets. *Left Panel:* RMSE(k) with $k = 1, \ldots, 6$ for PLSR (red solid line) and BPLSR (black dashed line) before contamination. *Right Panel:* RMSE($k^\star$) with $k = 1, \ldots, 6$ for PLSR (red solid line) and BPLSR (black dashed line) after contamination.

sion. The two lines in the left panels for all three figures almost coincide. It is therefore once more verified the efficiency of the BPLSR method.

Simulation setting - B

In this simulation study we are interested on the loss of the regression coefficient. We fix, as in the previous simulation study, $k_{max} = 3$, $\boldsymbol{P} = \boldsymbol{I}_{3,p}$, and $\boldsymbol{q} = (1, 1, 1)'$. The covariance matrix for the components $\boldsymbol{\Sigma}_{kk}$ is now set to

$$\boldsymbol{\Sigma}_{kk} = \text{diag}\,(50, 10, 1) \quad \text{and} \quad \boldsymbol{\Sigma}_{kk} = \text{diag}\,(100, 50, 10).$$

Hence, we want to have a better idea on the coefficient's loss for different structures in Σ_{kk}. Similar to the preceding simulation study we retain

$$\epsilon_i \sim \mathcal{N}(0, 0.1), \quad \text{for} \quad i = 1, \ldots, n.$$

We fix $n = 100$ and $p = 100$ and we contaminate by replacing

$$5\%,\ 10\%,\ 15\%,\ 20\%,\ 25\%,\ \text{and}\ 30\%,$$

of the data set with outliers. The contaminated error vectors are generated by elements for which

$$\epsilon_{i',cont}, f_{i',cont} \sim \mathcal{N}(\mu, 1) \quad \text{for} \quad \mu = 5 \quad \text{and} \quad i' = 1, \ldots, \ell,$$

where ℓ equals n times the contamination rate. In contrast to the simulation setting - A, here we do not split the data since we do not focus on out-of-sample prediction error. We are interested on the loss of the regression coefficient vector. The latter is measured using the mean squared error (MSE) given by

$$\mathrm{MSE}_k(\widehat{\boldsymbol{\beta}}) = \mathsf{E}_M[(\widehat{\boldsymbol{\beta}}_k^m - \boldsymbol{\beta})'(\widehat{\boldsymbol{\beta}}_k^m - \boldsymbol{\beta})], \tag{5.32}$$

with $\widehat{\boldsymbol{\beta}}_k^m$ for $m = 1, \ldots, 1000$ denoting the estimated coefficient vector in the m^{th} simulation for a model including k components. In contrast to the simulations setting which followed the Gunst-Mason approach, in the bilinear factor model setting we know $k^\star$ in advance, so we use the RD1-BACON with 4 retained scores vectors and parameter c set to 10 in order to have a relatively reasonable initial basic subset ($4 \times 10 = 40$ units) in the BACON procedure. Furthermore, note that we do not really know the true coefficient $\boldsymbol{\beta}$. One possible solution for our simulation is to estimate it using LS and to set $\boldsymbol{\beta} = \widehat{\boldsymbol{\beta}}_p^{pls}$ which is true for $k = p$, yet this is computationally too expensive. Here we are mostly interested on the resistance of the BPLS regression estimate after contamination of the data, therefore an alternative approach is to take for $\boldsymbol{\beta}$ the PLS regression estimate on the clean data. That is we use expression (5.32), we fix $k = k^\star$, and we measure the distances between:
a. the PLS regression coefficient vector after contamination and the PLS vector on clean data, and
b. the BPLS regression coefficient vector after contamination and the PLS vector on clean data.

Simulation results - B

Given $\boldsymbol{\beta} = \widehat{\boldsymbol{\beta}}_{k^\star}^{pls}$ in expression (5.32) and $k^\star = 3$, we compute and compare the $\mathrm{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}^{\mathrm{BPLSR}})$ and the $\mathrm{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}^{\mathrm{PLSR}})$ for the different contamination levels. The resulting values are given in Table 5.10 and Table 5.11.

Table 5.10: Simulation Results: MSE for the regression coefficient vectors for different contamination levels. Σ_{kk} set to $\mathrm{diag}\,(50, 10, 1)$.

	Contamination Rate					
	5%	10%	15%	20%	25%	30%
$\mathrm{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}^{\mathrm{PLSR}})$	0.77819	0.77968	0.78853	0.79327	0.7920063	0.7997610
$\mathrm{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}^{\mathrm{BPLSR}})$	0.16928	0.22352	0.27657	0.31750	0.4168965	0.7694110

Tables 5.10 and 5.11 indicate that the BPLS regression coefficient limits its loss compared to the ordinary PLS regression coefficient vector when contamination

Table 5.11: Simulation Results: MSE for the regression coefficient vectors for different contamination levels. Σ_{kk} set to $\text{diag}\,(100, 50, 10)$.

	Contamination Rate					
	5%	10%	15%	20%	25%	30%
$\text{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}^{\text{PLSR}})$	0.56279	0.5793630	0.58753	0.57643	0.58359	0.59401
$\text{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}^{\text{BPLSR}})$	0.042032	0.055595	0.06664	0.07553	0.13228	0.512856

is present. Indeed the BPLS regression vector resists to outliers up to 25% contamination levels. Both Tables provide a similar picture for the coefficient's loss. The BPLS regression vector performs even better for the case where components are more variable. Note that in this case the PLS regression vectors is less sensitive to contamination compared to its performance for Σ_{kk} set to $\text{diag}\,(50, 10, 1)$. Nevertheless, the relative efficiency for the two coefficient vectors, that is the ratio $\text{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}^{\text{BPLSR}})/\text{MSE}_{k^\star}(\widehat{\boldsymbol{\beta}}^{\text{PLSR}})$ indicates better performance for the BPLS regression vector in both case.

5.5 Conclusions

The BACON algorithm has been used in order to robustly estimate the variance-covariance matrix and to run robust PLS regression. A robust alternative for PC regression has been also given. Following Helland (1988), robust methods for derived components regression methods were presented. These methods were tested on real and simulated data, on both low and high dimensional data. The former were generated within a concrete statistical framework which has already been used in the statistical literature. The latter followed the bilinear factor model commonly used in PLS regression.

The results on both real and simulated data demonstrate that the BACON algorithm is resistant to reasonable levels of outliers. Regression estimates were not much affected by the outlying values and consequently BPLS and BPC predictions and regression estimates restricted their loss only for high contamination levels and relatively high dimensions of the data sets. In the latter case the BACON algorithm was extended to provide solution for rank deficient data. Two approaches were presented, yet the first one (RD1-BACON) has been mainly used in our illustration due to structural similarities with PLS and PC regression methods. In the rank deficient case we simulated (a) in order to control robustness of the proposed method concerning model selection, and (b) in order to measure the relative loss for PLS and BPLS regression coefficient vector under contamination compared to the ordinary PLS regression vector on clean data. The results indicated robustness in model selection for the BPLS regression com-

pared to ordinary PLS regression. As it has been already seen in the real world examples the BPLS regression method avoids the inclusion of more components in the final model because of the outliers. Therefore it is much more efficient in dimension reduction than PLS regression. Concerning point (b) the simulation results demonstrated limited loss of the BPLS regression compared to ordinary PLS regression when contamination is present. A systematic comparison of the BPLS and the BPC regression with other robust proposals was done. Despite the arbitrariness inherent in simulation studies, the comparisons revealed two main points. Firstly that the BACON approach is much more efficient and easy to compute compared to its competitors while it is robust on reasonable levels of contamination. Secondly, the simulation indicated that the BACON algorithms are more effective than the other robust methods.

Chapter 6

Preconditioning methods applied in PLS regression

6.1 Notational preliminaries

Throughout this chapter we explore numerical methods for solving linear systems of equations of the form

$$\mathsf{A}\,\mathsf{z} = \mathsf{b}, \tag{6.1}$$

and we apply these methods in the framework of PLS regression. The relation of the above systems with regression analysis stems from the system of the normal equations

$$\frac{1}{n}\boldsymbol{X}'\boldsymbol{X}\,\boldsymbol{\beta} = \frac{1}{n}\boldsymbol{X}'\boldsymbol{y}. \tag{6.2}$$

Letting $\mathsf{A} = \boldsymbol{X}'\boldsymbol{X}$ and $\mathsf{b} = \boldsymbol{X}'\boldsymbol{y}$, and replacing $\boldsymbol{\beta}$ by z for notational ease, the normal equations in (6.2) are written according to the linear system in (6.1). This will be from now on the notation we use throughout this chapter, relating the normal equations and the regression coefficient vector to A, b and z, respectively. Furthermore, we use the star superscript * for the regression estimates in order to emphasize estimates on the preconditioned systems. For this chapter we use the subscript m instead of k to denote the dimension of PLS regression models, while we use ℓ for the dimension of the PC regression models. We also use the symbol $\widetilde{\cdot}$ for two different notations. When it is used on b and A it denotes the preconditioned vector $\widetilde{\mathsf{b}}$ and the preconditioned matrix $\widetilde{\mathsf{A}}$. The symbol $\widetilde{\cdot}$ above the regression vector $\boldsymbol{\beta}$, or equivalently z, is used to denote transformed coordinates. These will be better understood on the following paragraphs.

6.2 Preconditioning Krylov spaces

Preconditioning techniques have been mainly attracted the interest of researchers in numerical analysis and applied mathematics. They consist in premultiplying

and/or postmultiplying linear systems by a non-singular matrix Q; the latter stands for the preconditioning matrix.

Definition 6.1 *Premultiplying or postmultiplying a linear system by a non-singular matrix* Q *is called preconditioning.*

Krylov spaces (see Mathematical Appendix I for definition and basic properties) are very suitable to approximate solutions in large linear systems. By preconditioning Krylov spaces we mean solving the preconditioned linear systems of the form

$$\mathsf{Q}\ \mathsf{A}\ \mathsf{z} = \mathsf{Q}\ \mathsf{b}, \tag{6.3}$$

using approximations derived from Krylov spaces, that is from spaces of the following form

$$\mathsf{span}\left(\mathsf{Q}\,\mathsf{b}, (\,\mathsf{Q}\,\mathsf{A}\,)^1\ \mathsf{Q}\,\mathsf{b}, (\,\mathsf{Q}\,\mathsf{A}\,)^2\,\mathsf{Q}\,\mathsf{b}, \ldots, (\,\mathsf{Q}\,\mathsf{A}\,)^{m-1}\,\mathsf{Q}\,\mathsf{b}\right) = \mathcal{K}_m(\,\mathsf{Q}\,\mathsf{b},\ \mathsf{Q}\,\mathsf{A}\,).$$

The system in (6.3) requires that the matrix Q is symmetric and positive definite (see Golub and Van Loan, 1996, p.532). The method of Conjugate Gradient, as already discussed, is intimately connected to Krylov spaces. It may be equally applied in order to solve preconditioned linear systems. The basic idea behind preconditioned Conjugate Gradients is to solve the transformed system

$$\mathsf{Q}\ \mathsf{A}\ \mathsf{Q}\ \tilde{\mathsf{z}} = \mathsf{Q}\ \mathsf{b}. \tag{6.4}$$

For a subspace of dimension m, the solution $\tilde{\mathsf{z}}_m$ is an approximation of $\tilde{\mathsf{z}}$, while the final solution on the original coordinates z is recovered by $\mathsf{z} = \mathsf{Q}\tilde{\mathsf{z}}$. Let $\boldsymbol{r}^\star_m, \boldsymbol{d}^\star_m$ denote the residual and the direction vectors in the Conjugate Gradients for the preconditioned system. For these vectors it is straightforward to verify that

$$\mathsf{span}(\boldsymbol{r}^\star_1, \ldots, \boldsymbol{r}^\star_m) = \mathsf{span}(\boldsymbol{d}^\star_1, \ldots, \boldsymbol{d}^\star_m) = \mathcal{K}_m(\,\widetilde{\mathsf{b}},\, \widetilde{\mathsf{A}}\,),$$

with $\widetilde{\mathsf{b}} = \mathsf{Q}\ \mathsf{b}$ and $\widetilde{\mathsf{A}} = \mathsf{Q}\,\mathsf{A}$. To verify the last expression use induction and note that for $m = 1$ we have $\boldsymbol{d}^\star_m = \mathsf{Q}\ \mathsf{b}$, while for $m > 1$ we get $\boldsymbol{d}^\star_m = \mathsf{Q}\ \mathsf{b} - \mathsf{Q}\ \mathsf{A}\ \mathsf{Q}\tilde{\mathsf{z}}_{m-1} = \mathsf{Q}\ \mathsf{b} - \mathsf{Q}\ \mathsf{A}\,\mathsf{z}_{m-1}$. This also holds for the residual vector since both $\boldsymbol{d}^\star_m$ and $\boldsymbol{r}^\star_m$ span the same space. For more details on preconditioned Conjugate Gradients see Golub and Van Loan (1996), Chapter 10.3.

Preconditioning and PLS regression

Preconditioning techniques may be adequately used in the framework of PLS regression. This stems from the fact that the instruments of the PLS regression algorithm, that is the weight and the score vectors as well as the regression coefficient vector, lie in Krylov spaces. PLS regression can be therefore extended in order to approximate solutions for preconditioned systems through the sequence

of preconditioned Krylov spaces. The preconditioned PLS regression coefficient vector is then expressed according to

$$\widehat{\boldsymbol{\beta}}_m^{\star} = \mathsf{K}_m^{\star}(\mathsf{K}_m^{\star\prime}\,\mathsf{A}\,\mathsf{K}_m^{\star})^{-1}\,\mathsf{K}_m^{\star\prime}\,\mathsf{b}, \tag{6.5}$$

for $\mathsf{K}_m^{\star}$ a non-singular matrix $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)$. It is not hard to show that the columns of $\mathsf{K}_m^{\star}$ span the preconditioned Krylov space, that is

$$\mathsf{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) = \mathcal{K}_m(\,\mathsf{Q}\,\mathsf{b},\ \mathsf{Q}\,\mathsf{A}). \tag{6.6}$$

The link between PLS regression and preconditioning techniques for Krylov spaces stems from the following proposition.

Proposition 6.1 *Let* Q *be a* $p \times p$ *non-singular and symmetric matrix, and define the matrix* $\widetilde{\mathsf{A}} = \mathsf{Q}\,\mathsf{A}$ *and the vector* $\widetilde{\mathsf{b}} = \mathsf{Q}\,\mathsf{b}$. *An orthogonal basis to solve the PLS regression problem is given by* $\mathsf{K}_m^{\star} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)$ *for which*

$$\mathsf{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) = \mathcal{K}_m(\widetilde{\mathsf{A}}, \widetilde{\mathsf{b}}). \tag{6.7}$$

The proof is given in the Mathematical Appendix II.

Use and choice of the preconditioner

Amongst other reasons, preconditioning may be used in order to:

1. reduce the number of iterations in Krylov spaces and accelerate convergence,
2. in certain cases preconditioners may be used as smoothers,
3. preconditioning may also improve the interpretation of the final solution.

The choice of an appropriate preconditioner is not easy. One should compensate the computational cost that naturally arises when preconditioning by the relative gain of their use. For smooth preconditioniers one can see Calvetti et al. (2005), while in Calvetti and Somersalo (2005) a Bayesian perspective in preconditioning is given. Krämer et al. (2006) use preconditioning in order to further penalize the PLS regression method.

Preconditioning techniques are illustrated here with two main goals:

1. We iteratively precondition the Krylov spaces and we propose a method to improve the interpretation of the PLS regression coefficient vector. This is done by downweighting predictors intervals, rather than individual variables, with no statistical relevance. The latter is measured via a relative importance statistic. We proceed in the detection of the predictors' intervals with low statistical importance. This is very important in order to have a better knowledge on the redundant variables especially in high dimensional data. This is explored in paragraph 6.3.

2. We use preconditioning techniques for solving linear systems and we find PLS and PC regression estimates to be the limiting solutions of the proposed method. Using lower dimensional expansion either on eigen or Krylov spaces we provide a rich ensemble of bases for the regression coefficient vector. The link between the proposed method and Cyclic Subspace Regression (CSR) (see Kalivas, 1999) is also given. Two important points are further investigated. Firstly the shrinkage properties of the resulting regression coefficient vectors and secondly the dimension reduction obtained by such regression models. Finally, using the preconditioning approach the regression coefficients for PC and PLS regression are expressed in terms of polynomials inside the Krylov spaces. This is given in paragraph 6.4.

The two proposals are implemented using both the Helland and the NIPALS algorithm for PLS regression.

6.3 Detection of non relevant predictors in PLS. Interpretation *vs* Prediction

As in most PLS regression applications, high dimensional data sets consist of hundreds of highly collinear variables which make the data matrix $\boldsymbol{X}$ rank deficient. For such problems common statistical inference is impossible. PLS regression solutions are very hard to interpret since they form predictors' intervals rather than single covariates which affect the response vector. It is feasible to weight these groups in order to make solutions easier to interpret. It is not unrealistic to believe that a relatively small number of these predictors' intervals have a significant effect on the response, while the remaining predictors get small values which represents noise. The idea therefore is to use an algorithm which downweights the predictor intervals with no statistical relevance without excluding any variable. This is achieved by using iteratively preconditioning. Recall that PLS solutions lie in Krylov spaces. Preconditioning the latter may provide regression coefficient vectors with good predictive power and easier interpretation.

6.3.1 Iterative preconditioning

The use of preconditioning to improve the interpretation of the final solution can be better understood in the following context. Consider the system

$$\mathsf{Q}\ \mathsf{A}\ \mathsf{z} = \mathsf{Q}\ \mathsf{b},$$

and let $\mathsf{Q} = \mathsf{diag}(\mathsf{q}_1, \ldots, \mathsf{q}_p)$ with the scalar diagonal element q_j denoting the relative importance of the j^{th} predictor variable. These will be defined shortly. Solving iteratively the system above by downweighting the non relevant predictors may lead to solutions which are much easier to interpret. Using s to indicate the current iteration number, the system above after s iterations is the solution to

$$\mathsf{Q}^{(s)}\ \mathsf{A}\ \mathsf{z}^{(s)} = \mathsf{Q}^{(s)}\ \mathsf{b}\,, \tag{6.8}$$

The matrix $\mathsf{Q}^{(s)}\ \mathsf{A}$ is a square but not symmetric $p \times p$ matrix. Note also that $\mathsf{Q}^{(s)}$ is positive definite since its diagonal elements are never set to zero.

The preconditioned system above can be solved following the system in (6.4); that is $\mathsf{Q}\,\mathsf{A}\,\mathsf{Q}\,\tilde{\mathsf{z}} = \mathsf{Q}\,\mathsf{b}$. The matrix $\mathsf{Q}\,\mathsf{A}\mathsf{Q}$ is now symmetric and the system is solved in terms of the transformed coordinates $\widetilde{\mathsf{z}}$ with the original coefficient vector being recovered as $\mathsf{z} = \mathsf{Q}\,\widetilde{\mathsf{z}}$. Working in terms of the transformed coordinates for $\mathsf{Q}\,\mathsf{A}\mathsf{Q}$ has an interesting statistical interpretation as it will be seen in paragraph 6.3.5. Letting the iteration parameter s enter the system, we solve iteratively

$$\mathsf{Q}^{(s)}\ \mathsf{A}\ \mathsf{Q}^{(s)}\,\widetilde{\mathsf{z}}^{(s)} = \mathsf{Q}^{(s)}\ \mathsf{b}\,, \tag{6.9}$$

with $\widetilde{\mathsf{z}}^{(s)}$ the s^{th} iterative solution on the transformed coordinate system. The final coefficient vector is easily recovered as $\mathsf{z}^{(s)} = \mathsf{Q}^{(s)}\,\widetilde{\mathsf{z}}^{(s)}$.

Following the iterative scheme described above we seek to detect the non relevant predictors and restrict their value close to zero. The final regression coefficient vector may be therefore easier to interpret since it will include near zero components corresponding to predictors with no statistical relevance.

Choosing the relative importance factors q_j

In order to capture statistical relevance we should introduce the relative importance factors q_j for $j = 1, \ldots, p$. The choice of the relative importance factors is the most important point for constructing the algorithm. In the simplest case there exist two groups of predictors, the relevant and the non relevant predictors. We can then decompose the predictor's matrix $\boldsymbol{X}$ as

$$\boldsymbol{X} = (\boldsymbol{X}_A,\, \boldsymbol{X}_B\,)\,,$$

with A being the set of indexes $j\ :\ \boldsymbol{x}_j$ is a relevant predictor, and B the set of indexes $j\ :\ \boldsymbol{x}_j$ is not a relevant predictor. Setting

$$\mathsf{q}_j \neq 0 \ \text{ for } j\ \in\ A, \ \text{ and } \ \mathsf{q}_j = 0 \ \text{ for } j\ \in\ B,$$

is a relatively crude approach because in high dimensional data it is very hard to draw such discriminant rules. Take for instance an association measure such as the cov or the corr between each predictor $\boldsymbol{x}_j$ and the response $\boldsymbol{y}$ for the relative importance factors; a rather smooth function relating the relative importance with the response will normally appears. The idea therefore is to focus on predictors' intervals and to apply a function for the relative importance factors rather than discrete 0-1 values for each predictor.

Our approach is to run the PLS regression on preconditioned systems based on matrix Q which carries the relevance information on the predictors. The first step is to run the PLS regression and to obtain estimates for the regression coefficient vector from a subspace of dimension $m < p$; the latter is commonly determined by means of model selection methods such as the cross validation and

the bootstrap (see Hastie et al., 2004, Chapter 7). From this model we construct the preconditioning matrix Q according to

$$\mathsf{Q} = \mathsf{diag}\,(\,\mathsf{q}_1, \ldots,\, \mathsf{q}_p\,), \;\; \text{for} \;\; \mathsf{q}_j \propto \sum_{k=1}^{m} |\, w_{j,k}|\,, \tag{6.10}$$

where $|\, w_{j,m}|$ is the absolute value for the covariance $\mathsf{cov}(\boldsymbol{x}_{j,m-1},\, \boldsymbol{y})$ for dimension m. This essentially measures the association between the response $\boldsymbol{y}$ and the (residual) predictor variables $\boldsymbol{x}_{j,m-1}$ for all the extracted dimensions up to dimension m. Note that this is easy to obtain since it arises directly from the weight vectors in the PLS regression algorithm. For $s > 1$ preconditioning provides the sequence $(\boldsymbol{w}_1^\star, \ldots, \boldsymbol{w}_m^\star)$ which is easily expressed in terms of the initial sequence $(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m)$ as it will be seen shortly. The relative importance of each predictor is accumulated throughout the s iterations, and at each iteration all the relevant predictors will appear from the m first dimensions (this is further discussed in paragraph 6.3.6). The remaining predictors will be successively discarded. Note that by $\propto$ in expression (6.10) we denote that we normalize the importance factors. This is helpful for implementation aspects described in paragraph 6.3.4. This scaling is used only to compute the relative importance factors in the expression (6.10). Using the NIPALS instead of the Helland PLS regression algorithm, allows one to use the loadings

$$p_{mj} = \mathsf{corr}(\boldsymbol{x}_{j,m-1}, \boldsymbol{t}_m) \;\; \text{and} \;\; \boldsymbol{p}_m = \{p_{j,m}\} \;\; \text{for} \;\; j = 1, \ldots, p,$$

instead of the covariances in (6.10). The latter however reflects mostly the strength of the relation of each predictor variable with the derived score rather than the direct association between predictors and response. Therefore, we retain in our implementations the weight vectors in expression (6.10).

6.3.2 Using the NIPALS algorithm

The NIPALS algorithm is used in the framework described above in the following way. For each extracted dimension m in the NIPALS algorithm we extract the X-weight vector according to

$$\boldsymbol{w}_m^\star \propto \mathsf{Q}\, \boldsymbol{X}'_{m-1}\, \boldsymbol{y},$$

and the corresponding score vectors are then obtained as

$$\boldsymbol{t}_m^\star \propto \boldsymbol{X}_{m-1}\, \boldsymbol{w}_m^\star.$$

We run the usual PLS regression algorithm based on $\boldsymbol{w}_m^\star$ and $\boldsymbol{t}_m^\star$ which compose the matrices $\boldsymbol{W}_m^\star$ and $\boldsymbol{T}_m^\star$, respectively. Similarly to the non preconditioned case, the ordinary PLS regression, the matrix $\boldsymbol{R}^{\star(m)}$ has the following properties:

Lemma 6.1 *Dropping the superscript* $^{(m)}$ *to simplify notations, and using the superscript* * *to denote weights and scores derived in the preconditioned system, the following statements hold:*

1. *The matrix* $\boldsymbol{R}^\star = \boldsymbol{T}^{\star\prime}\,\boldsymbol{X}\,\boldsymbol{W}^\star$ *is right bidiagonal.*
2. *The matrix* $\boldsymbol{R}^{\star\prime}\boldsymbol{R}^\star$ *is tridiagonal.*

The proof is given in the Mathematical Appendix II. Corollary 6.1 guarantees that we can still use approximations from the sequence of the tridiagonal matrices $\boldsymbol{R}^{\star(m)\prime}\boldsymbol{R}^{\star(m)}$ for $m \leq p$. The NIPALS algorithm uses the weight vectors $\boldsymbol{w}_m^\star$ in order to create such a basis for the preconditioned Krylov spaces.

Corollary 6.1 *For* $\widetilde{\mathsf{A}} = \mathsf{Q}\,\mathsf{A}$ *and* $\widetilde{\mathsf{b}} = \mathsf{Q}\,\mathsf{b}$*, the space spanned by the weight vectors* $\boldsymbol{w}_m^\star$ *is given by* $\mathsf{span}(\boldsymbol{w}_1^\star, \ldots, \boldsymbol{w}_m^\star) = \mathcal{K}_m(\widetilde{\mathsf{b}},\ \widetilde{\mathsf{A}})$*, while the final coefficient vector* $\widehat{\boldsymbol{\beta}}_m^\star$ *lies in* $\mathcal{K}_m(\widetilde{\mathsf{b}},\ \widetilde{\mathsf{A}})$*.*

For a given dimension m, the NIPALS algorithm may be iteratively rerun s times for a suitable choice for the preconditioning matrix $\mathsf{Q}^{(s)}$; the suffix (s) indicates the current iteration. This leads to two sets of predictors. The first set contains non relevant predictors clustered very close to zero, and the second contains the relevant predictors. The former are naturally discarded since they are sequentially downweighted, yet they are not excluded. The resulting coefficient vector is much more easy to interpret.

Proposition 6.2 *At each iteration* s*, the space spanned by the weight vector denoted by* $\boldsymbol{w}_m^{\star s}$ *is the same as* $\mathcal{K}_m(\widetilde{\mathsf{b}},\ \widetilde{\mathsf{A}})$ *for*

$$\widetilde{\mathsf{b}} = \mathsf{Q}_s\ \mathsf{b} \quad \textit{and} \quad \widetilde{\mathsf{A}} = \mathsf{Q}_s\ \mathsf{A}, \quad \textit{where} \quad \mathsf{Q}_s = \prod_{i=1}^{s}\ \mathsf{Q}^{(i)}.$$

For the proof see the Mathematical Appendix II.

It is straightforward to extend Corollary 6.1 and to use Proposition 6.2 to get the following remark for the regression coefficient vector at iteration s.

Remark 6.1 *Following the notation in Proposition 6.2, for the regression coefficient vector at iteration* s*, denoted as* $\widehat{\boldsymbol{\beta}}_m^{(s)\star}$*, the following statement holds:*

$$\widehat{\boldsymbol{\beta}}_m^{(s)\star} \ \in\ \mathcal{K}_m(\,\mathsf{Q}_s\,\mathsf{b},\ \mathsf{Q}_s\,\mathsf{A}).$$

The NIPALS implementation is given in Algorithm 6.13. It is based on two loops; on s and m. The latter depends on a model selection criterion, while the former depends on a threshold value. The choice of the threshold is discussed in paragraph 6.3.4.

Algorithm 6.13 NIPALS implementation for detection of irrelevant predictors

Input: $\boldsymbol{X}_0 \leftarrow \boldsymbol{X}$; $\boldsymbol{y}_0 \leftarrow \boldsymbol{y}$;
$s \leftarrow 1$;
Initialize $\mathsf{Q}^{(1)} = \mathsf{diag}(1, \ldots, 1)$;
while *a certain variable selection criterion is not fulfilled* **do**
  $m \leftarrow 1$;
  while *a certain model selection criterion is not fulfilled* **do**
    Step 1. Extract direction $\boldsymbol{w}_m^{\star s} = \mathsf{Q}^{(s)}\, \boldsymbol{X}'_{m-1} \boldsymbol{y}_{m-1}$, normalize $\boldsymbol{w}_m^{\star s}$ and store it in $\boldsymbol{W}_m^{\star s}$;
    Step 2. Take score $\boldsymbol{t}_m^{\star s} = \boldsymbol{X}_{m-1}\, \boldsymbol{w}_m^{\star s}$ and store in $\boldsymbol{T}_m^{\star s}$;
    Step 3 Compute X-loading $\boldsymbol{p}_m^{\star s} \propto \boldsymbol{X}'_{m-1} \boldsymbol{t}_m^{\star s}$ and store in $\boldsymbol{P}_m^{\star s}$;
    Step 4 Compute Y-loading $\boldsymbol{q}_m^{\star s} \propto \boldsymbol{y}'_{m-1} \boldsymbol{t}_m^{\star s}$ and store in $\boldsymbol{Q}_m^{\star s}$;
    Step 5. Recover the transformed implied regression coefficient vector as

$$\widehat{\boldsymbol{\beta}}_m^{\star} = \boldsymbol{W}_m^{\star s} \left(\boldsymbol{P}_m^{\star s\prime} \boldsymbol{W}_m^{\star s}\right)^{-1} \boldsymbol{Q}_m^{\star s}.$$

    Step 6. Deflate data as: $\boldsymbol{X}_m = \boldsymbol{X}_{m-1} - \boldsymbol{t}_m^{\star s}\, \boldsymbol{p}_m^{\star s\prime}$, $\boldsymbol{y}_m = \boldsymbol{y}_{m-1} - \boldsymbol{t}_m^{\star s}\, q_m^{\star s}$.
    $m \leftarrow m + 1$;
  end while
  Store $\widehat{\boldsymbol{\beta}}_m^{\star (s)} \leftarrow \widehat{\boldsymbol{\beta}}_m^{\star}$, and define the relative importance factors $\mathsf{q}_1, \ldots, \mathsf{q}_p$ according to expression (6.10);
  $s \leftarrow s + 1$;
  Store $(\mathsf{q}_1^{(s)}, \ldots, \mathsf{q}_p^{(s)}) \leftarrow (\mathsf{q}_1, \ldots, \mathsf{q}_p)$; $\mathsf{Q}^{(s)} = \mathsf{diag}(\mathsf{q}_1^{(s)}, \ldots, \mathsf{q}_p^{(s)})$;
end while
Output: Give the final regression coefficient vector $\widehat{\boldsymbol{\beta}}_m^{(s)\star}$ according to

$$\widehat{\boldsymbol{\beta}}_m^{(s)\star} = \mathsf{Q}^{(s)}\, \widehat{\boldsymbol{\beta}}_m^{\star (s-1)} = \mathsf{Q}^{(s)}\, \mathsf{Q}^{(s-1)} \ldots \mathsf{Q}^{(2)}\, \mathsf{Q}^{(1)}\, \widehat{\boldsymbol{\beta}}_m^{\star (1)},$$

and give the final predictions according to $\widehat{\boldsymbol{y}}_m^{\star} = \boldsymbol{X}\, \widehat{\boldsymbol{\beta}}_m^{(s)\star}$.

6.3.3 Using the Helland algorithm

A very convenient and direct way to express the weight vectors and to recover the PLS regression coefficients is given by Helland (1988). We have seen in Lemma 6.1 that for the preconditioned case we still have a tridiagonal matrix $\boldsymbol{R}^{\star\prime}\boldsymbol{R}^{\star}$ which now gives approximations based on $\boldsymbol{W}^{\star\prime}\mathsf{A}\boldsymbol{W}^{\star}$, implying $\mathsf{Q}\ \mathsf{A}\ \mathsf{Q}$ orthogonality for the weight vectors. The PLS regression coefficient is expressed as

$$\widehat{\boldsymbol{\beta}}^{\star}_{m} = \boldsymbol{W}^{\star}_{m}\,(\boldsymbol{W}^{\star\prime}_{m}\,\mathsf{A}\,\boldsymbol{W}^{\star}_{m})^{-1}\,\boldsymbol{W}^{\star\prime}_{m}\,\mathsf{b}. \tag{6.11}$$

It can be reexpressed in terms of $\boldsymbol{W}$ according to

$$\widehat{\boldsymbol{\beta}}^{\star}_{m} = \mathsf{Q}\,\overbrace{\boldsymbol{W}_{m}\,(\boldsymbol{W}'_{m}\,\underline{\mathsf{Q}\ \mathsf{A}\ \mathsf{Q}}\,\boldsymbol{W}_{m})^{-1}\,\boldsymbol{W}'_{m}\,\underline{\mathsf{Q}\ \mathsf{b}}} = \mathsf{Q}\,\widetilde{\widehat{\boldsymbol{\beta}}}^{\star}_{m}, \tag{6.12}$$

where the vector $\widetilde{\widehat{\boldsymbol{\beta}}}^{\star}_{m}$ is the solution to the transformed system with preconditioners given by $\mathsf{Q}\ \mathsf{A}\ \mathsf{Q}$ and $\mathsf{Q}\ \mathsf{b}$. This vector is indicated by the brace in expression (6.12), while $\mathsf{Q}\ \mathsf{A}\ \mathsf{Q}$ and $\mathsf{Q}\ \mathsf{b}$ are underlined to emphasize the preconditioners on the transformed coordinates. It is straighforward to transform back the solution to the original coordinates system for the coefficient vector according to $\widehat{\boldsymbol{\beta}}^{\star}_{m} = \mathsf{Q}\,\widetilde{\widehat{\boldsymbol{\beta}}}^{\star}_{m}$. Expression (6.12) results from (6.11) using induction and by noting that:

$$\begin{aligned}
\text{for } m=1, \quad \boldsymbol{w}_{m} &= \mathsf{b} \text{ while } \boldsymbol{w}^{\star}_{m} = \mathsf{Q}\ \mathsf{b},\\
\text{for } m>1, \quad \boldsymbol{w}_{m} &= \mathsf{b} - \mathsf{A}\,\widehat{\boldsymbol{\beta}}_{m-1},\\
\text{while,} \quad \boldsymbol{w}^{\star}_{m} &= \mathsf{Q}\ \mathsf{b} - \mathsf{Q}\ \mathsf{A}\ \mathsf{Q}\,\widetilde{\widehat{\boldsymbol{\beta}}}^{\star}_{m-1}\\
&= \mathsf{Q}\ \mathsf{b} - \mathsf{Q}\ \mathsf{A}\,\widehat{\boldsymbol{\beta}}^{\star}_{m-1}\\
&= \mathsf{Q}\,(\mathsf{b} - \mathsf{A}\,\widehat{\boldsymbol{\beta}}^{\star}_{m-1}).
\end{aligned}$$

Given the expression (6.12), the Helland algorithm for PLS regression is used in order to improve the interpretation of the coefficient vector based on iteratively preconditioning the Krylov spaces as follows:

1. For $s = 1$ the Helland algorithm for PLS regression is run without preconditioning (that is with $\mathsf{q}_{j} = 1$ for all $j = 1, \ldots, p$).

2. For $s \geq 2$ the Helland algorithm for PLS regression is run using preconditioning and subspace approximation m.

For a given s (not included in the notation below), the Helland's recurrence becomes

$$\boldsymbol{w}^{\star}_{m} \propto \begin{cases} \widetilde{\mathsf{b}} & \text{for } m = 1,\\ \widetilde{\mathsf{b}} - \widetilde{\mathsf{A}}\,\widehat{\boldsymbol{\beta}}^{\star}_{m-1}, & \text{for } m > 1, \end{cases} \tag{6.13}$$

where as usual $\widetilde{\mathsf{b}} = \mathsf{Q}\,\mathsf{b}$, $\widetilde{\mathsf{A}} = \mathsf{Q}\,\mathsf{A}$ and $\widehat{\boldsymbol{\beta}}^{\star}_{m-1}$ is given in expression (6.12). Note once more that for $\widehat{\boldsymbol{\beta}}^{\star}_{m-1}$ in the expression above that

$$\widetilde{\mathsf{A}}\,\widehat{\boldsymbol{\beta}}^{\star}_{m-1} = \mathsf{Q}\;\mathsf{A}\,\widehat{\boldsymbol{\beta}}^{\star}_{m-1} = \mathsf{Q}\;\mathsf{A}\;\mathsf{Q}\,\widetilde{\widehat{\boldsymbol{\beta}}}^{\star}_{m-1},$$

where the last expression gives the regression estimate on the transformed coordinates.

The Helland implementation benefits from the fact that it is direct and fast since neither score nor loading vectors are computed. Data deflation is also avoided. The interpretation for the final solution is straightforward since the regression coefficient vector at each iteration s solves the following optimization problem

$$\arg \min_{\boldsymbol{\beta} \in \mathcal{K}(\mathsf{Q}_s\,\mathsf{b},\,\mathsf{Q}_s\,\mathsf{A})} \|\boldsymbol{y} - \boldsymbol{X}\,\boldsymbol{\beta}\|, \quad \text{for} \quad m \leq p.$$

The Helland implementation of the proposed method is given in Algorithm 6.14. It is based on two loops, on s and on m. As in the NIPALS case the first depends on a chosen threshold, and the second is based upon a model selection criterion.

6.3.4 Implementation aspects

The proposed algorithms support throughout the s iterations the most relevant predictors, while they downweight the predictor with low relative importance. Algorithms 6.13 and 6.14 build regression models which are more easy to interpret. In the context of NIR experiments for example this translates to detecting for which wavelengths the NIR spectra play an important role in predicting the response. The proposed regression models are extensions for the PLS regression model and one should certainly take into account their prediction performance. This is especially true given that PLS regression models are well known for their high predictive performance. This point is discussed in more detail in paragraph 6.3.6. What has to be defined is a stopping rule for s. The choice of the stopping rule is based on the following principles:

1. we seek for models that are easy to interpret and which do not lose much of their good prediction performance,

2. we hope that s is small in order to make the algorithm fast.

The choice for the stopping rule should be based upon a reasonable criterion that respects the two previous points. A simple rule is to construct two subsets of predictors, the relevant predictors subset $\mathcal{S}_A = \{\boldsymbol{x}_j : j \in A\}$, and the non relevant predictor's subset $\mathcal{S}_B = \{\boldsymbol{x}_j : j \in B\}$. Evidently $\mathcal{S}_A \cap \mathcal{S}_B = \emptyset$ and $\mathcal{S}_A \cup \mathcal{S}_B = \mathcal{S}$, with $\mathcal{S} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p\}$. The stopping rule it then follows directly.

Algorithm 6.14 Helland implementation for detection of irrelevant predictors

Input: $\mathsf{A} = \boldsymbol{X}'\boldsymbol{X}$, $\mathsf{b} = \boldsymbol{X}'\boldsymbol{y}$;

$s \leftarrow 1$;
Initialize with $\mathsf{Q}^{(1)} = \mathsf{diag}(1, \ldots, 1)$, and set $\widetilde{\mathsf{A}}^{\star} = \mathsf{Q}^{(1)}\ \mathsf{A}\ \mathsf{Q}^{(1)}$; $\widetilde{\mathsf{b}} = \mathsf{Q}^{(1)}\ \mathsf{b}$;

while *a certain variable selection criterion is not fulfilled* **do**

$m \leftarrow 1$;

while *a certain model selection criterion is not fulfilled* **do**

Step 1. Compute the loadings $\boldsymbol{w}_m^{\star s}$ according to

$$\boldsymbol{w}_m^{\star s} \propto \begin{cases} \widetilde{\mathsf{b}} & \text{for } m = 1, \\ \widetilde{\mathsf{b}} - \widetilde{\mathsf{A}}^{\star}\, \widehat{\boldsymbol{\beta}}_{m-1}^{\star}, & \text{for } m > 1, \end{cases}$$

Step 2. Store the loading vectors $\boldsymbol{w}_m^{\star s}$ into the matrix $\boldsymbol{W}_m^{\star s}$.

Step 3. Recover the transformed implied regression coefficient vector $\widehat{\boldsymbol{\beta}}_m^{\star}$ according to

$$\widehat{\boldsymbol{\beta}}_m^{\star} = \boldsymbol{W}_m^{\star s}\, (\boldsymbol{W}_m^{\star s\prime}\ \mathsf{A}\, \boldsymbol{W}_m^{\star s})^{-1}\, \boldsymbol{W}_m^{\star s\prime}\ \mathsf{b}.$$

$m \leftarrow m + 1$;

end while

Store $\widehat{\boldsymbol{\beta}}_m^{\star(s)} \leftarrow \widehat{\boldsymbol{\beta}}_m^{\star}$, and define $(\mathsf{q}_1, \ldots, \mathsf{q}_p)$ according to expression (6.10);

$s \leftarrow s + 1$;

$(\mathsf{q}_1^{(s)}, \ldots, \mathsf{q}_p^{(s)}) \leftarrow (\mathsf{q}_1, \ldots, \mathsf{q}_p)$;
$\mathsf{Q}^{(s)} = \mathsf{diag}(\mathsf{q}_1^{(s)}, \ldots, \mathsf{q}_p^{(s)})$; $\widetilde{\mathsf{A}}^{\star} = \mathsf{Q}^{(s)}\ \mathsf{A}\ \mathsf{Q}^{(s)}$; $\widetilde{\mathsf{b}} = \mathsf{Q}^{(s)}\ \mathsf{b}$;

end while

Output: Give the final regression coefficient vector $\widehat{\boldsymbol{\beta}}_m^{(s)\star}$ according to

$$\widehat{\boldsymbol{\beta}}_m^{(s)\star} = \mathsf{Q}^{(s)}\, \widehat{\boldsymbol{\beta}}_m^{\star(s-1)} = \mathsf{Q}^{(s)}\ \mathsf{Q}^{(s-1)} \ldots \mathsf{Q}^{(2)}\ \mathsf{Q}^{(1)}\ \widehat{\boldsymbol{\beta}}_m^{\star(1)},$$

and give the final predictions according to $\widehat{\boldsymbol{y}}_m^{\star} = \boldsymbol{X}\, \widehat{\boldsymbol{\beta}}_m^{(s)\star}$.

Definition 6.2 *Stopping Rule*

$$\text{STOP} \quad \textit{when} \quad \mathcal{S}_B^{(s)} = \mathcal{S}_B^{(s+1)}. \tag{6.14}$$

The proposed algorithm stops when the set of the non relevant predictors remains the same for two consecutive iterations. It can also be stated the other way around, that is

$$\text{STOP} \quad \text{when} \quad \mathcal{S}_A^{(s)} = \mathcal{S}_A^{(s+1)}.$$

A threshold c must be finally settled in order to classify the predictor variables into the corresponding sets $(\mathcal{S}_A^{(s)}, \mathcal{S}_B^{(s)})$. This is done according to the classification rule below:

$$\boldsymbol{x}_j \in \mathcal{S}_A^{(s)} \quad \text{iff} \quad \mathsf{q}_j^{(s)} > c \quad \text{and} \quad \boldsymbol{x}_j \in \mathcal{S}_B^{(s)} \quad \text{iff} \quad \mathsf{q}_j^{(s)} \leq c.$$

The threshold value is very important for two reasons. Firstly, because different thresholds may lead to quite different results. Secondly, because its value should have a meaningful statistical interpretation.

The threshold c should be a small value for which one can freely indicate irrelevant predictors. The strength of the association between the predictors with the response is captured by the diagonal elements of Q. Given that the vector of the relative important factor $\mathsf{RIF} = (\mathsf{q}_1, \ldots, \mathsf{q}_p)$ is normalized to one, see expression (6.10), one can measure the redundancy of the j-th predictor by setting c to a small value in the following way: if all the elements in the coefficient vector were equally important then for a given length of RIF (that is p) one can simply set c as the normalized value of one of its elements. For example for a data set with $p = 100$ this equals 0.01, and one can set $c \leq 0.01$. For a data set with $p = 300$ one can choose $c \leq 0.05$. The choice for the threshold value should certainly exploit prior knowledge and prior belief, whenever this is possible. The proposed method above corresponds to no prior knowledge. The connection to the Baeysian approach which is given in the following paragraph motivates the statistical interpretation of the proposed method and the choice of the threshold.

6.3.5 Statistical interpretation and connection to Bayesian methods

The main ideas of Bayesian methods for variable selection in regression are given in Mitchell and Beauchamp (1988), George and McCulloch (1993), George and McCulloch (1997). Bayesian variable selection in NIR spectroscopy is given in Brown et al. (1998a) and Brown et al. (1998b) for multivariate regression problems. The use of Bayesian variable selection is based on the assumption that the regression coefficient vector $\boldsymbol{\beta}$ is normally distributed and each component β_j is considered as being modelled from a mixture of normal densities. In particular, the conditional distribution of the response variable Y is generally given as

$$Y | \, \boldsymbol{\beta} \, , \, \sigma^2 \sim \mathcal{N}(\boldsymbol{X} \, \boldsymbol{\beta}, \, \sigma^2 \boldsymbol{I}),$$

for a 0-1 unknown vector $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$ that identifies the statistical important variables. These have a non-zero coefficient β_j. For the vector $\boldsymbol{\gamma}$ a prior probability density is given according to $\pi(\boldsymbol{\gamma})$ for which $\pi(\boldsymbol{\gamma} \,|\, \theta)$ stands for a Bernoulli structure for $\boldsymbol{\gamma}$ conditional on θ. This allows to give to $\boldsymbol{\gamma}$ a beta mixed binomial distribution, and θ to be either concentrated or dispersed according to prior belief as it is given in the beta hyperparameters. Each component β_j is considered as being modelled from a mixture of normal densities, and therefore $\beta_j | \, \gamma_j$ is given as

$$\beta_j | \, \gamma_j \;\sim\; (1 - \gamma_j)\, \mathcal{N}(0, \tau_j^2) + \gamma_j \, \mathcal{N}(0, c_j^2 \tau_j^2), \tag{6.15}$$

for a small-τ_j and for large-c_j. The former guarantees that for $\gamma_j = 0$ then β_j would be a small value ranging over zero, while given a large-c_j for $\gamma_j = 1$ a non-zero estimate for β_j is included in the model. To obtain the above mixture, George and McCulloch (1993) use a p-variate normal prior for β_j conditional on γ_j as

$$\boldsymbol{\beta} | \, \boldsymbol{\gamma} \;\sim\; \mathcal{N}_p(0, \mathbf{D}_\gamma \, \widetilde{\mathbf{R}} \, \mathbf{D}_\gamma), \tag{6.16}$$

where $\widetilde{\mathbf{R}}$ denotes the prior covariance-correlation matrix, and

$$\mathbf{D}_\gamma = \mathsf{diag}(\alpha_1 \, \tau_1, \ldots, \alpha_p \, \tau_p). \tag{6.17}$$

The latter can be seen as scaling $\widetilde{\mathbf{R}}$ in such a way that (6.15) is satisfied. They propose $\alpha_j = 1$ for $\gamma_j = 0$, and $\alpha_j = c_j$ for $\gamma_j = 1$. The subscript $\boldsymbol{\gamma}$ indicates that the diagonal elements depend upon γ_j. Priors on the residual variance are usually taken according to an Inverse Gamma distribution with some scale parameters. The use of sampling techniques such as the GIBBS or the Metropolis-Hastings algorithms allow for an initial $\boldsymbol{\gamma}$ vector, to generate sequences

$$\boldsymbol{\beta}_s, \, \sigma_s, \, \boldsymbol{\gamma}_s, \quad \text{for} \quad s = 1, 2, \ldots$$

with $\boldsymbol{\beta}_s$, σ_s and $\boldsymbol{\gamma}_s$ simulated by posteriors densities. The existence of the matrix $\mathbf{D}_\gamma \, \widetilde{\mathbf{R}} \, \mathbf{D}_\gamma$ in the expression for the posterior density for $\boldsymbol{\beta}_s$, and for a suitable choice for α_j and τ_j allows for a fast convergence to selected variables ($j \,:\, \gamma_j = 1$ most frequently) since the most promising variables are most commonly selected. For more details on the choice of α_j, c_j, and computational aspects one can see George and McCulloch (1993).

The proposed method does not exclude any variable from the analysis. Therefore it is not a variable selection proposal. Yet, similarities to Bayesian variable selection are highlighted below. These allow to improve the statistical interpretation of the proposed method, and to select a statistical meaningful threshold.

Bayesian methods initialize with the ones vector, that is $\boldsymbol{\gamma}_1 = \mathrm{c}(1, ..., 1)$, and they generate $\boldsymbol{\beta}_1$, σ_1 by their posterior densities in order to have the new estimate for $\boldsymbol{\gamma}_1$, and so forth. Until convergence they construct the sequence: $\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_s$. The proposed method starts by performing a PLS regression with preconditioning matrix

$$\mathsf{Q}_1 = \mathsf{diag}(1, \ldots, 1),$$

and uses the importance factors q_j to construct the following preconditioners. Both the diagonal elements of the preconditioning matrix $\mathsf{Q}^{(s)}$ and the vectors $\boldsymbol{\gamma}_s$ express belief and evidence on the importance of the predictor variables $\boldsymbol{x}_j$ with $j = 1, \ldots, p$.

In the Bayesian framework the values for c_j and α_j are determined with respect to a binary vector $\boldsymbol{\gamma}$. Our approach does not use a discrete 0-1 valued vector. It rather uses the vector of relative importance factors RIF in order to downweight predictors corresponding to low relevance. This is well suited for high dimensional data sets.

The importance factors q_j, similar to the elements of $\mathbf{D}_\gamma$, identify the variables which are most supported by the data. In this sense the importance factors q_j scale the variance-covariance matrix and yield information on the relevance of each predictor. Relevance here is strongly connected with the association of the predictors with the response vector. To see that note that the data variance-covariance matrix is proportional to

$$\left(\begin{array}{c|c} \mathsf{A} & \mathsf{b} \\ (\boldsymbol{X}'\boldsymbol{X}) & (\boldsymbol{X}'\boldsymbol{y}) \\ \hline \mathsf{b}' & \mathsf{c} \\ (\boldsymbol{X}'\boldsymbol{y})' & (\boldsymbol{y}'\boldsymbol{y}) \end{array}\right). \tag{6.18}$$

Let $\mathsf{A} = \{\alpha_{ij}\}$ with $i, j = 1, \ldots, p$. The elements α_{ij} define the variance-covariance structure of the X-data. Recall Algorithm 6.14 and see that on the transformed coordinates the matrix A is scaled at each iteration s by the relative importance factors according to

$$\mathsf{Q}\,\mathsf{A}\,\mathsf{Q} = \left\{\, \mathsf{q}_i\, \mathsf{q}_j\, \alpha_{ij} \,\right\}.$$

The relative importance factors allow the predictors which are more strongly associated with the response vector more chance than the less important predictors. Note that the j-th predictor which is downweighted throughout the iterations gets values for $\widehat{\boldsymbol{\beta}}_m^{\star(s)}$ very close to zero. This is due to Remark 6.1 and to the threshold value. The same predictor yields a relatively high precision resulting by the small element in $\mathsf{q}_j^2 \alpha_{jj}$. This is illustrated in Figure 6.1 for two coefficients $\widehat{\beta}_1^{(s)}$ and $\widehat{\beta}_2^{(s)}$, for $s = 1, 2$. The red solid line corresponds to a normally distributed $\widehat{\beta}_1^{(1)}$ and the black thick line to the resulting distribution for $\widehat{\beta}_1^{(2)}$ after preconditioning. Similarly, the distribution of $\widehat{\beta}_2^{(s)}$ for $s = 1$ and for $s = 2$ is illustrated by the blue dashed and the green dotted line, respectively. The vertical solid grey line is the 0 vertical line, while the dotted gray lines indicate the threshold value $\pm c$. Note that the threshold is based on the q_j and no hard threshold is imposed

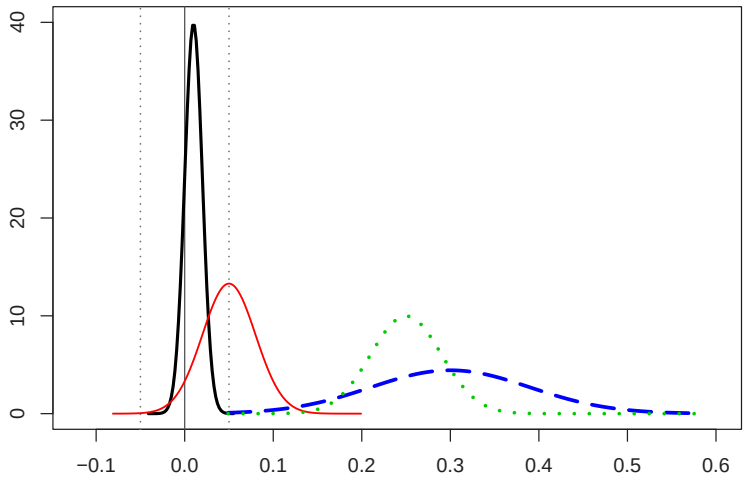


Figure 6.1: Distribution of the regression coefficients' elements $\widehat{\beta}_1^{(s)}$ and $\widehat{\beta}_2^{(s)}$ for $s = 1, 2$. The red solid line corresponds to $\widehat{\beta}_1^{(1)}$ and the black thick line to the distribution of $\widehat{\beta}_1^{(2)}$. The distribution of $\widehat{\beta}_2^{(s)}$ for $s = 1$ and for $s = 2$ is illustrated by the blue dashed and the green dotted line, respectively.

on the coefficients. Yet, the importance of the latter is reflected on their relative importance factors.

Taking normal distributions for the coefficients' elements with variance proportional to the q_j's we see that the precision for both coefficient elements is smaller for $s = 2$. This is normally expected since $\alpha_{jj}^{(s=2)} = \mathsf{q}_j\, \mathsf{q}_j\, \alpha_{jj}^{(s=1)}$, with $\mathsf{q}_j < 1$. Note that the first element $\widehat{\beta}_1^{(s)}$ for $s = 1$ falls below the threshold value. For the consecutive iterations its value falls very close to zero with larger precision. This is not the case for $\widehat{\beta}_2^{(s)}$ which decreases to zero with a much slower rate than $\widehat{\beta}_1^{(s)}$. Its precision is always larger than $\widehat{\beta}_1^{(s)}$ providing less evidence of being zero. As it is seen in Figure 6.1, for a fixed s, $\widehat{\beta}_2^{(s)}$ is more variable compared to $\widehat{\beta}_1^{(s)}$.

6.3.6 Dimension reduction and prediction performance

A question which naturally arises is whether preconditioning affects the subspace m where the solution for the regression coefficient vector is searched for. That is, we want to know if m depends on s since we get our relative importance factors upon a m-dimensional Krylov space. Moreover if we can fix m through the s iterations we need to define m only once in the proposed algorithms. That

is, we fix the dimension of the Krylov space only for $s = 1$, after running the original PLS regression. In this way Algorithms 6.13 and 6.14 are much less computationally expensive. In order to give an answer we need to define the minimal polynomial of a square matrix given below.

Definition 6.3 *The polynomial $q(t)$ is the minimal polynomial of the matrix A if it is the unique monic polynomial of minimal degree such that $q(\mathsf{A}) = 0$. Let $\lambda_1, \ldots, \lambda_p$ denote the p distinct eigenvalues of matrix A, then $q(t)$ is defined as*

$$q(t) = \prod_{j=1}^{p} (t - \lambda_j).$$

The importance of the above definition lies on the following proposition which is taken from Ipsen and Meyer (1998).

Proposition 6.3 *If the minimal polynomial of the nonsingular matrix A has degree m then the solution to $\mathsf{A}\,\mathsf{z} = \mathsf{b}$ lies in the space $\mathcal{K}_m(\mathsf{A}, \mathsf{b})$.*

Proposition 6.3 concerns non singular matrices, and it is not easy to generalize it for the rank deficient case. Yet, in order to treat the rank deficient case one should at least start by the full rank case. The following lemma concerns the degree of the minimal polynomial for the preconditioned matrices $\widetilde{\mathsf{A}}^{(s)}$, and it therefore gives an answer on whether m depends on s.

Lemma 6.2 *For $\ell = 1, 2, \ldots, s$ the degree for the minimal polynomial of the nonsingular matrix $\left(\mathsf{Q}^{(\ell)}\ \mathsf{A}\ \mathsf{Q}^{(\ell)}\right)$ does not exceed m.*

Proof: *See in the Mathematical Appendix II.*

From Lemma 6.2, and at least for the non singular cases, we see that m may be retained fixed throughout the sequence of iterations. The singular case should be further investigated. A remaining important issue is the effect of the iterative method proposed in Algorithms 6.13 and 6.14 on the prediction performance of the regression models. This will be empirically explored in the examples that follow.

6.3.7 Experimentation

Gasoil Data

The Gasoil data consists of 115 samples from which the UV spectra are measured over 572 channels at equally spaced wavelengths between 200.15 nm and 400 nm. Observations 1 to 70 are the calibration samples, and 71 to 114 are the prediction

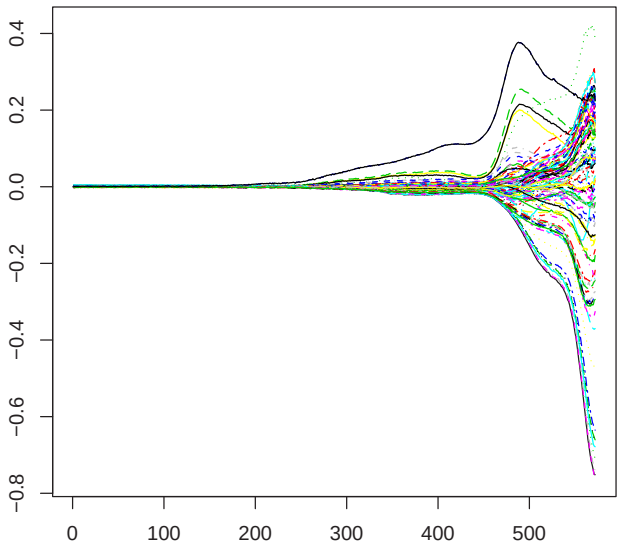
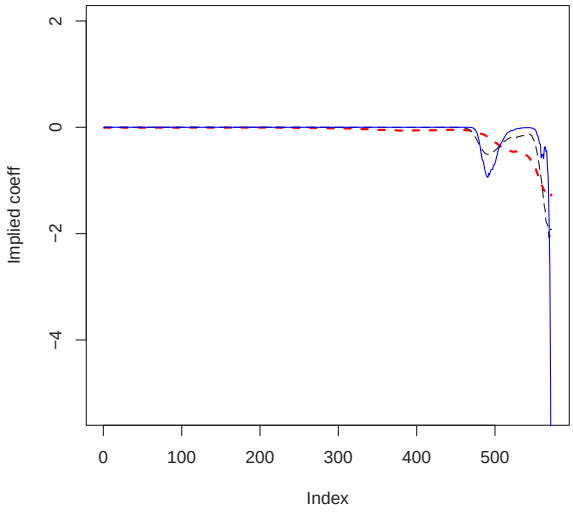


Figure 6.2: Gasoil data. NIR calibration spectra on 115 samples measured throughout 572 wavelengths.

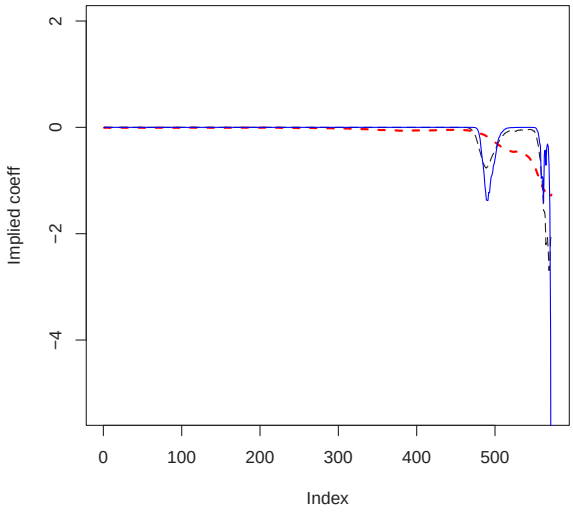
samples. Observation 115 is an outlier. The data set is available through the web link *http://www.dal.ca/ pdwentze/download/gasoil.txt*. The spectra for all 115 samples are given in Figure 6.2.

The Gasoil data include 4 response variables. In our analysis we use only the first column as response in order to run univariate PLS regression. We set the threshold to $c = 0.05$ and we run Algorithm 6.13 and Algorithm 6.14 using as training set the calibration sample, and test set the prediction sample. Figure 6.3 illustrates the regression coefficient lines for Algorithm 6.14 on the upper panel, and 6.13 on the lower panel, respectively. The coefficient lines correspond to iterations $s = 2$ (thin dashed line) and $s = 8$ (solid line). The thick dashed line in both panels illustrates the ordinary PLS regression coefficient. The condition $\mathcal{S}_A^{(s)} = \mathcal{S}_A^{(s+1)}$ is accomplished after 13 and 21 iterations for the NIPALS and the Helland implementation, respectively. The regression lines are almost identical. The non zero PLS regression coefficients cluster around the end of the wavelength range.

Moreover, in Table 6.1 we provide the number of variables belonging to the final subset ($\mathsf{card}(\mathcal{S}_A^{(s)})$), for $s = 0$ (no preconditioning), $s = 8$, as well as for the final iteration. Together we provide the out-of-sample prediction error measured by means of the Root Mean Squared Error of Prediction (RMSEP) for one two



(a) Preconditioning the NIPALS.



(b) Preconditioning the Helland PLS regression.

Figure 6.3: Gasoil data. Regression coefficients after $s = 2$ and $s = 8$. The thick dashed line corresponds to the ordinary PLS regression coefficient, while the coefficient lines for $s = 2$ is illustrated by the thin dashed line and for $s = 8$ by the the solid line.

and three retained components. We remind that the RMSEP for a model based on m components is given by

$$\text{RMSE}_m = \sqrt{\frac{1}{n_{test}} \sum_{i=1}^{n_{test}} (y_i - \widehat{y}_{im})^2}, \tag{6.19}$$

with n_{test} indicating the size for prediction sample, respectively, and $\widehat{y}_{im}$ the fitted value for observation i for the regression model constructed on m components.

Table 6.1: Gasoil data: The number of relevant regression coefficients ($\mathsf{card}(\mathcal{S}_A^{(s)})$) for $s = 8$, $s = 20$ and $s = 16$, together with the corresponding out-of-sample prediction error measured by means of RMSEP_m for $m = 1, 2, 3$.

method	s	$\mathsf{card}(\mathcal{S}_A^{(s)})$	RMSEP_1	RMSEP_2	RMSEP_3
NIPALS	0	572	0.97192	0.92483	1.24553
	8	67	1.18872	0.92334	1.07031
	13	58	1.18339	0.91694	1.05903
Helland	0	572	0.97192	0.92483	1.24553
	8	82	1.31622	0.93573	1.00342
	21	58	1.31467	0.88271	0.93758

From Table 6.1 we note that both algorithms end up with the same number of relevant predictors, yet the Algorithm 6.13 is faster than Algorithm 6.14. It moreover retains a better level of prediction. Note also that both algorithms reduce the prediction loss compared to the PLS regression model. Regarding dimension reduction there is no change for the number of components in the final model; it remains the same for the preconditioned PLS regression. Finally, we mention that a more hard threshold, for example $c = 0.005$, makes convergence faster ($s = 16$ for Algorithm 6.13 and $s = 11$ for Algorithm 6.14), while the dimension reduction and the prediction loss remains almost the same.

Octane Data

The Octane data consists of 39 gasoline samples for which the octanes have been measured in 225 wavelengths (measured in nm). It is a high dimensional data set which has been extensively analyzed in the literature, see for example Tenenhaus (1998); Kondylis and Hadi (2006b). The response of interest is the octane concentration.

We are interested to apply the proposed algorithm in order to detect which wavelengths' intervals are irrelevant. We use here the NIPALS algorithm. We furthermore use cross validation in order to control the dimension reduction and the final out-of-sample prediction error for the final model. To do so we split the data into a train and a test set of dimension equal to 20 and 19 samples,

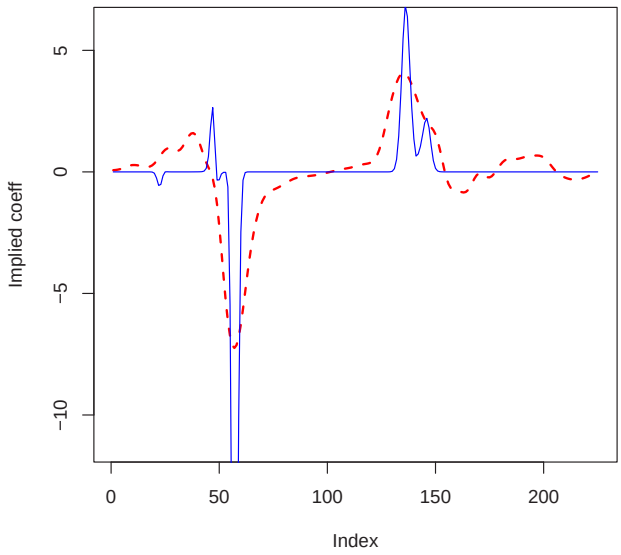
respectively. Finally, we set the threshold $c = 0.05$. Running the ordinary PLS regression (no preconditioning) results in a final model on two components with the corresponding out-of-sample prediction error equal to $\text{RMSEP}_2 = 2.53264$. The RMSEP_2 stands for the root mean squared error of prediction for a PLS regression model based on two components. The thick dashed line in both panels in Figure 6.4 illustrates the ordinary PLS regression coefficient estimate. In the same figure we draw the line for the coefficient estimate after preconditioning. Fifteen was the final number of iterations for Algorithm 6.13, see Figure 6.4(a). Figure 6.4(b) contains the same information for Algorithm 6.14.

For the number for the relevant coefficients we get $\mathsf{card}(\mathcal{S}_A^{(15)}) = 22$. The prediction loss is equal to $\text{RMSEP}_2 = 10.5942$. The reduction of the dimension for both algorithms is very satisfying. Yet, the out-of-sample prediction error has increased. If we make the value for c harder ($c < 0.05$) the algorithm stops for $s = 11$ and $\mathsf{card}(\mathcal{S}_A^{(11)}) = 40$. The prediction loss is now equal to 10.2440 which is quite similar to the previous results. Very similar results are obtained by further reducing the threshold value c.

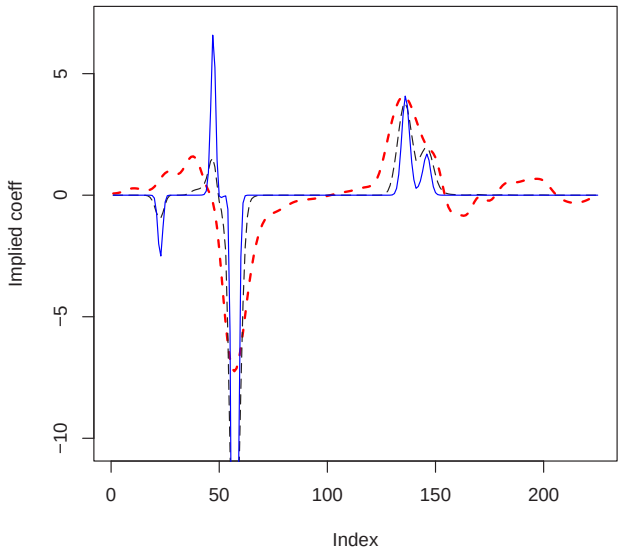
Selwood Data

The Selwood data have been firstly analyzed by Selwood et al. (1990) from where their name comes from. They consist of 31 antimycin congeners studied for their antifilarial activity on 53 quantum-chemical descriptors which construct the predictors matrix. The data set have been mainly analyzed in the framework of Quantitative Structure-Activity Relationships (QSAR). They can be also found in Eriksson et al. (2000).

We run Algorithm 6.13 in order to improve the interpretation of the implied regression coefficients. We additionally use cross validation to control the dimension reduction and the out-of-sample prediction error for the final model. We split randomly the data into a training and a test set of dimensions equal to 15 and 16 samples, respectively. Finally, we set the threshold $c = 0.05$. Running Algorithm 6.13 without preconditioning results in a final model on two components yielding prediction loss $\text{RMSEP}_2 = 1.7497$. By using preconditioning, the algorithm converges fast ($s = 2$) with similar prediction loss, and predictors 37, 38, 39 detected as relevant. We plot in Figure 6.5 the implied regression coefficients. Figure 6.5(a) illustrates in thick circles the PLS implied regression coefficient vector for a model based on two components. This is known as the true dimension for the Selwood data (see Eriksson et al., 2000). Figure 6.5(b) illustrates in thick circles the PLS implied regression coefficient vector for an overfitted model based on six components. The dashed lines in both panels correspond to the coefficient vector after preconditioning is applied. Note that for the overffited model the algorithm converges for $s = 4$. That is the iterative process has been two doubled. This is mainly due to overfitting since the implied regression coefficient vector includes noise. The latter also increases the final group of the relevant predictors to 10. It is very important to note that the proposed method does not



(a) Preconditioning the NIPALS.



(b) Preconditioning the Helland PLS regression.

Figure 6.4: Octane data. Implied regression coefficient vector for ordinary PLS regression (red thick dashed line) and the preconditioned solution after $s = 15$ iterations (blue solid line.

provide the same results (on statistical relevant predictors) for a proper and for an overfitted model.

6.3.8 Conclusions

It is very hard to interpret the regression coefficients of PLS regression models. This is true especially in high dimensional applications. This was the case for all the examples treated here, the Gasoil, the Octane, and the Selwood data. We have used preconditioning techniques in subspace iterative search based on Krylov spaces. We have used a diagonal matrix as the precondition matrix, with its diagonal elements reflecting the importance of the predictors as a function of the PLS regression weight vectors. Throughout a Bayesian motivation we have seen that premultiplying and postmultiplying the covariance matrix A results in scaling the variance-covariance structure of the data in order to let the predictors which are most supported by the data to play a more important role in model construction. Imposing a threshold value c allowed to construct two sets of predictors based on their significance or their redundancy. We let the preconditioning scheme to be iteratively rerun a number of times (s) until the subset of relevant predictors did no longer change.

The results coming from our limited experience are rather optimistic. Despite the large dimensions on all three data sets, our method gave fast and interpretable results. Yet, two points remain open for further research and discussion. Firstly, the threshold value c which may have a better statistical interpretation. Secondly, we need to establish a pure variable selection framework which is used not only to make interpretation of the regression estimates easy, but which allows to remove completely the detected predictors from the final regression model.

6.4 Preconditioning Krylov spaces using eigenvectors

Brief review on eigen and Krylov spaces

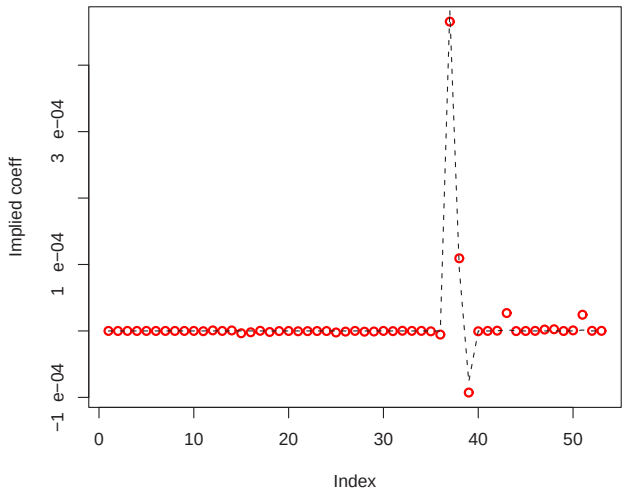
Two important spaces for the interpretation of the PC and the PLS regression estimates are the eigen and the Krylov spaces. The former results by the eigen decomposition of matrix A, given according to

$$\mathsf{A} = \mathsf{W}\,\mathsf{diag}(\lambda_1, \ldots, \lambda_p)\,\mathsf{W}', \tag{6.20}$$

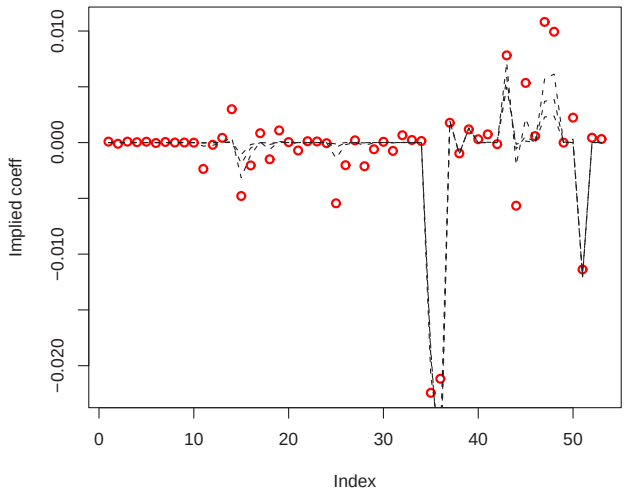
for λ_j and W denoting the ordered eigenvalues and the corresponding orthogonal matrix of eigenvectors. We recall the following definitions which can be also found in the Mathematical Appendix I.

Definition 6.4 *The space spanned by the $\ell \leq p$ eigenvectors corresponding to the first ℓ eigenvalues $(\lambda_1, \ldots, \lambda_\ell)$ in (6.20) is called an eigen space of dimension ℓ, and it is denoted by*

$$\mathsf{span}\,(\,\mathsf{w}_1, \ldots, \mathsf{w}_\ell) = \mathcal{E}_\ell\,(\,\mathsf{A}). \tag{6.21}$$



(a) Preconditioning the NIPALS. Number of components set to 2.



(b) Preconditioning the NIPALS. Number of components set to 6.

Figure 6.5: Selwood data. Implied regression coefficients. The thick circles corresponds to the ordinary PLS regression coefficients, while the coefficients for $s > 1$ are illustrated by the thin dashed lines.

Definition 6.5 *For the matrix* A *and the vector* b*, the space spanned by the sequence*

$$\{\mathsf{b},\ \mathsf{A}^1\,\mathsf{b}, \ldots,\ \mathsf{A}^{m-1}\,\mathsf{b}\},$$

is a Krylov space of dimension $m \leq p$*, denoted as*

$$\mathsf{span}\,(\,\mathsf{b},\ \mathsf{A}^1\,\mathsf{b}, \ldots,\ \mathsf{A}^{m-1}\,\mathsf{b}\,) = \mathcal{K}_m(\,\mathsf{b},\ \mathsf{A}\,). \tag{6.22}$$

Krylov spaces have nice properties such as scaling and translation invariance (see Parlett, 1980, Chapter 12). Finally for an orthogonal basis change induced by the matrix $\boldsymbol{M}$ (for which $\boldsymbol{M}' = \boldsymbol{M}^{-1}$) we get an orthogonal similarity tranformation, that is

$$\mathcal{K}_m(\boldsymbol{M}\ \mathsf{b}, \boldsymbol{M}\ \mathsf{A}\,\boldsymbol{M}') = \boldsymbol{M}\,\mathcal{K}_m(\,\mathsf{b},\ \mathsf{A}\,). \tag{6.23}$$

See also the link between Krylov spaces and Conjugate Gradient methods in Chapter 2 and the Mathematical Appendix I.

Definition 6.6 *Define as algebraic grade of a Krylov space denoted by* $m^\star$ *the lowest value of* m *for which*

$$\dim \mathcal{K}_{m^\star+1}(\,\mathsf{b},\ \mathsf{A}\,) = \dim \mathcal{K}_{m^\star}(\,\mathsf{b},\ \mathsf{A}\,),$$

with dim *denoting the dimension of the generated space.*

Usually $m^\star = p$ unless there are eigenvalues of multiplicity greater than 1, resulting to $m^\star < p$. Without loss of generality and unless it is otherwise noted we assume that $m^\star = p$.

Approximating eigen spaces by using Krylov spaces

Krylov spaces are effectively used in order to approximate the spectrum of A especially for the extreme eigenvalues and eigenvectors when matrix A is large and sparse. This is done by the Lanczos method (see Lanczos, 1950) and the tridiagonalization of matrix A. The result is a tridiagonal matrix $\boldsymbol{H}^{(m)}$ which equals

$$\boldsymbol{H}^{(m)} = \boldsymbol{C}_m{}'\,\mathsf{A}\,\boldsymbol{C}_m, \tag{6.24}$$

given that $\mathsf{span}\,(\boldsymbol{c}_1\ \ldots, \boldsymbol{c}_m\,) = \mathcal{K}_m(\,\mathsf{b},\ \mathsf{A}\,)$. The use of the superscript $^{(m)}$ emphasizes that the dimension of the Krylov space from which the spectrum of A is approximated is equal to m. The approximations to the eigen pairs $(\lambda_j,\ \mathsf{w}_j)$ derived by $\mathcal{K}_m(\,\mathsf{b},\ \mathsf{A}\,)$ are the *ritz* pairs $\left(\theta_j^{(m)},\ \boldsymbol{\phi}_j^{(m)}\right)$ (see Parlett, 1980, chapter 8). The importance of the latter and their connection to PLS regression has been already highlighted in Phatak (1993); Wen and Manne (2000).

6.4.1 PC and PLS regression revisited

Recall from the overview on PLS regression in Chapter 3 that both PC and PLS regression methods replace the data matrix $\boldsymbol{X}$ by an orthogonal score matrix of rank much less than p. The score matrices result from the projection of $\boldsymbol{X}$ on direction vectors which solve different optimization problems. PC weight vectors seek to maximize the variance on the predictor's space subject to certain orthonormality constraints, while PLS regression vectors maximize the covariance between the extracted scores and the response subject to similar constraints; see paragraph 3.2.2. The link between the eigen and Krylov spaces and the PC and PLS optimization criteria stems from the following remarks:

Remark 6.2 *The solution to the PC optimization problem in expression (3.7) corresponds to* w_ℓ*, for* ℓ *taking over* k*, and the PC regression estimate can be expressed as*

$$\widehat{\boldsymbol{\beta}}_\ell^{pcr} = \arg \min_{\boldsymbol{\beta} \in \mathcal{E}_\ell(\mathsf{A})} \|\boldsymbol{y} - \boldsymbol{X}\,\boldsymbol{\beta}\|_2. \tag{6.25}$$

The PC regression solution is the least squares solution truncated in the first ℓ *eigen directions* $(\mathsf{w}_1, \ldots, \mathsf{w}_\ell)$ *which compose the matrix* W_ℓ*. PC regression vector is equally expressed in terms of the eigen pairs of matrix* A *according to*

$$\widehat{\boldsymbol{\beta}}_\ell^{pcr} = \sum_{j=1}^{\ell} \lambda_j^{-1}\, \mathsf{w}_j\, \mathsf{w}_j'\, \mathsf{b} \quad \textit{for} \quad \ell \leq p\,. \tag{6.26}$$

Remark 6.3 *The solution to the PLS regression optimization problem in (3.6) corresponds to the Conjugate Gradient direction vectors* $\boldsymbol{d}_m$ *(with* m *taking over* k*) which compose the matrix* $\boldsymbol{D}_m$*. The PLS regression estimate can be expressed as*

$$\widehat{\boldsymbol{\beta}}_m^{pls} = \arg \min_{\boldsymbol{\beta} \in \mathsf{span}(\boldsymbol{D}_m)} \|\boldsymbol{y} - \boldsymbol{X}\,\boldsymbol{\beta}\|_2. \tag{6.27}$$

It can be also shown that

$$\mathsf{span}\,(\boldsymbol{d}_1, \ldots, \boldsymbol{d}_m) = \mathcal{K}_m(\mathsf{b}, \mathsf{A}), \tag{6.28}$$

and the PLS regression solution is the least squares solution truncated in the first m *conjugate gradient directions. The PLS regression coefficient lies in a Krylov space of dimension* m*. PLS regression vector is equally expressed in terms of the* ritz *pairs according to*

$$\widehat{\boldsymbol{\beta}}_m^{pls} = \sum_{j=1}^{m} \left(\theta_j^{(m)}\right)^{-1} \boldsymbol{\phi}_j^{(m)}\, \boldsymbol{\phi}_j^{(m)\prime}\, \mathsf{b}\,, \quad \textit{for} \quad m \leq p\,. \tag{6.29}$$

Recall that in this chapter m takes over k for PLS regression, while ℓ is used for PC regression. Expressions (6.26) and (6.29) correspond to the expressions (3.30) and (3.24) already given in Chapter 3. They are reproduced here in order

to emphasize that both PC and PLS regression provide a different basis for the regression coefficient vector $\boldsymbol{\beta}$. Indeed PC and PLS regression vectors for regression models based on ℓ and m components may be obtained by the projection of the LS regression coefficient $\widehat{\boldsymbol{\beta}}^{ls}$ on the space spanned by the weight vectors $(\mathsf{w}_1, \ldots, \mathsf{w}_\ell)$ and $(\boldsymbol{d}_1, \ldots, \boldsymbol{d}_m)$, respectively. For $m = \ell = p$ it is well known that the PLS and the PC regression solution coincide to the LS solution. This is given in the following remark.

Remark 6.4 *For* $m = \ell = p$: $\widehat{\boldsymbol{\beta}}_m^{pls} = \widehat{\boldsymbol{\beta}}_\ell^{pcr} = \widehat{\boldsymbol{\beta}}^{ls}$.

Preconditioning, PC and PLS regression

Preconditioning techniques may be adequately used in the frame of PLS regression. This is based on preconditioning Krylov spaces as it has already been seen in the beginning of this chapter. Eigen spaces may be also used to precondition linear systems. In particular, the eigenvectors $\mathsf{w}_1, \ldots, \mathsf{w}_\ell$ for $\ell \leq p$, when they compose the preconditioning matrix $\mathsf{Q} = \mathsf{W}_\ell$ they provide a very interesting preconditioner. The connection between preconditioning and PC regression is straightforward; it is given in the proposition below.

Proposition 6.4 *The PC regression coefficient vector is the solution to the preconditioned system*

$$\widetilde{\mathsf{A}}_\ell \, \mathsf{z} = \widetilde{\mathsf{b}}_\ell, \tag{6.30}$$

with preconditioned matrix $\widetilde{\mathsf{A}}_\ell$ *and vector* $\widetilde{\mathsf{b}}_\ell$ *resulting from the eigen decomposition in (6.20).*

Proof: *See in the Mathematical Appendix II.*

6.4.2 Preconditioning Krylov spaces using eigenvectors

The use of the eigenvectors $\mathsf{w}_1, \ldots, \mathsf{w}_\ell$ in W_ℓ for the preconditioning matrix is here further explored. In fact LS, PC and PLS regression are limiting solutions of such preconditioned linear systems. The connection between these three regression methods is given through the solution to the following preconditioned system

$$\mathsf{W}_\ell' \, \mathsf{A} \, \mathsf{W}_\ell \, \widetilde{\mathsf{z}} = \mathsf{W}_\ell' \, \mathsf{b}\,, \quad \text{where} \quad \mathsf{z} = \mathsf{W}_\ell \widetilde{\mathsf{z}}\,. \tag{6.31}$$

The system (6.31) is solved on the transformed coordinates to obtain $\widetilde{\mathsf{z}}$ and the final solution is recovered by $\mathsf{z} = \mathsf{W}_\ell \widetilde{\mathsf{z}}$. The latter is a $p \times 1$ vector. Note that the matrix $\mathsf{W}_\ell' \, \mathsf{A} \, \mathsf{W}_\ell$ on the transformed coordinates system corresponds to a diagonal matrix $\boldsymbol{\Lambda}_\ell$ with entries the eigenvalues λ_j of A corresponding to the eigenvectors w_j which are retained in $\widetilde{\mathsf{A}}$. Hence the solution to the system on the transformed coordinates above reduces to

$$\boldsymbol{\Lambda}_\ell \widetilde{\mathsf{z}} = \mathsf{W}_\ell' \, \mathsf{b}, \tag{6.32}$$

with $\mathbf{\Lambda}_\ell$ being easily inverted. For reasons that will be clarified at the next paragraph we choose to solve the preconditioned system in (6.31) iteratively, following the structure of the Helland's algorithm for PLS regression. This consists in finding solutions to (6.31) which lie in Krylov spaces. We give in Algorithm 6.15 the detailed structure of the proposed method. We use the superscript * in order to emphasize on the new estimate. We also use the subscript $\ell \,|\, m$ to emphasize dependence on both m and ℓ. The use of $\widetilde{\cdot}$ over $\widehat{\boldsymbol{\beta}}$ or z emphasizes solutions on the transformed coordinates. The use of $\widetilde{\cdot}$ over b and A denotes the preconditioning vector and matrix b and A, respectively. Regarding the choice of the eigenvectors which compose the preconditioning matrix W_ℓ, one can either take eigenvectors ordered by sequence of extraction or equally choose a set of eigenvectors which will construct the preconditioning matrix.

Algorithm 6.15 follows the Helland implementation for PLS regression on the preconditioned system. The weight vectors $\boldsymbol{d}^\star_{\ell\,|\,m}$ form an orthonormal basis for the regression coefficient vector as long as $m \leq \ell$; this will be justified shortly. The matrix $\boldsymbol{H}^{(m)} = (\boldsymbol{D}^{\star\prime}_{\ell\,|\,m}\ \mathsf{A}\ \boldsymbol{D}^\star_{\ell\,|\,m})$ is an easily inverted tridiagonal matrix from which the *ritz* values and vectors are derived. More importantly, Algorithm 6.15 establishes the link between LS, PLS and PC regression. This link is provided in the following remark.

Remark 6.5 *For different choice of m and ℓ in Algorithm 6.15 we get the following points:*

1. *Solving the system in (6.31) for $\ell = p$ results in the PLS regression coefficient vector $\widehat{\boldsymbol{\beta}}^{pls}_m$, that is*
$$\widehat{\boldsymbol{\beta}}^{pls}_m = \widehat{\boldsymbol{\beta}}^\star_{\ell=p\ |\ m}\,. \tag{6.33}$$

2. *Solving the system in (6.31) for $m = \ell$, and retaining only $\ell < p$ eigenvectors on W_ℓ results in the PC regression coefficient vector $\widehat{\boldsymbol{\beta}}^{pcr}_\ell$, that is*
$$\widehat{\boldsymbol{\beta}}^{pcr}_\ell = \widehat{\boldsymbol{\beta}}^\star_{\ell\ |\ m=\ell}\,. \tag{6.34}$$

3. *The solution to the system in (6.31) for $\ell = m = p$ is regression coefficient vector equal for all regression methods, that is*
$$\widehat{\boldsymbol{\beta}}^\star_{\ell\,|\,m} = \widehat{\boldsymbol{\beta}}^{ls} = \widehat{\boldsymbol{\beta}}^{pls}_m = \widehat{\boldsymbol{\beta}}^{pcr}_\ell\,.$$

Remark 6.5 motivates the use of Krylov subspace methods for solving the system (6.32). The latter could be solved relatively easy since $\mathbf{\Lambda}_\ell = \mathsf{W}'_\ell\ \mathsf{A}\ \mathsf{W}_\ell$ is a diagonal matrix, and

$$\mathsf{W}'_\ell\ \mathsf{A}\ \mathsf{W}_\ell\,\widetilde{\mathsf{z}} = \mathsf{W}'_\ell\ \mathsf{b} \Rightarrow \mathbf{\Lambda}_\ell\,\widetilde{\mathsf{z}} = \mathsf{W}'_\ell\ \mathsf{b} \Rightarrow \mathsf{W}^{-1}_\ell\,\mathsf{z} = \mathbf{\Lambda}^{-1}_\ell\,\mathsf{W}'_\ell\ \mathsf{b} \Rightarrow \mathsf{z} = \mathsf{W}_\ell\,\mathbf{\Lambda}^{-1}_\ell\,\mathsf{W}'_\ell\ \mathsf{b},$$

Algorithm 6.15 Preconditioning Krylov spaces by eigenvectors (Helland)

Input: For $\mathsf{A} = \boldsymbol{X}'\boldsymbol{X}$ and $\mathsf{b} = \boldsymbol{X}'\boldsymbol{y}$;

Step 1. Decompose A according to $\mathsf{A} = \mathsf{W}\,\boldsymbol{\Lambda}\,\mathsf{W}'$, and recover loading matrix

$$\mathsf{W}_\ell = (\,\mathsf{w}_1, ..., \mathsf{w}_\ell), \quad \text{for } \ell = 1, \ldots, p,$$

where w_ℓ is the eigenvector corresponding to the ℓ^{th} eigenvalue of A.

Step 2. Fix ℓ, and set $\widetilde{\mathsf{A}}^\star = \mathsf{W}'_\ell\,\mathsf{A}\,\mathsf{W}_\ell$ and $\widetilde{\mathsf{b}} = \mathsf{W}'_\ell\,\mathsf{b}$;

Step 3. For $m = 1, ..., p$,

- Compute the loadings $\boldsymbol{d}^\star_{\ell\,|\,m}$ according to

$$\boldsymbol{d}^\star_{\ell\,|\,m} \propto \begin{cases} \widetilde{\mathsf{b}} & \text{for } m = 1, \\ \widetilde{\mathsf{b}} - \widetilde{\mathsf{A}}^\star\,\widetilde{\widehat{\boldsymbol{\beta}}}^\star_{\ell\,|\,m-1}, & \text{for } m > 1, \end{cases}$$

- Store the loading vectors $\boldsymbol{d}^\star_{\ell\,|\,m}$ and form the matrix

$$\boldsymbol{D}^\star_{\ell\,|\,m} = (\boldsymbol{d}^\star_{\ell\,|\,1}, \ldots, \boldsymbol{d}^\star_{\ell\,|\,m})\,.$$

- Compute the transformed implied regression coefficient vector

$$\widetilde{\widehat{\boldsymbol{\beta}}}^\star_{\ell\,|\,m} = \boldsymbol{D}^\star_{\ell\,|\,m}\,(\boldsymbol{D}^{\star\prime}_{\ell\,|\,m}\,\mathsf{A}\,\boldsymbol{D}^\star_{\ell\,|\,m})^{-1}\,\boldsymbol{D}^{\star\prime}_{\ell\,|\,m}\,\mathsf{b}.$$

Output: Get the original regression coefficient vector according to

$$\widehat{\boldsymbol{\beta}}^\star_{\ell\,|\,m} = \mathsf{W}_\ell\,\widetilde{\widehat{\boldsymbol{\beta}}}^\star_{\ell\,|\,m},$$

and give the final predictions according to : $\widehat{\boldsymbol{y}}^\star_{\ell\,|\,m} = \boldsymbol{X}\,\widehat{\boldsymbol{\beta}}^\star_{\ell\,|\,m}$.

which corresponds to the PC regression on ℓ components; see the equivalence to expression (6.26). Yet, solving the system by means of Algorithm 6.15 establishes the link between LS, PLS and PC regression as given in Remark 6.5. In addition, the connection between *ritz* and eigen pairs, and consequently between PC and PLS regression, is easier understood through system (6.32) since its solution for $m < \ell$ corresponds to approximating the eigen values $\lambda_j \in \mathbf{\Lambda}_\ell$ by the ritz values $\theta_j^{(m)}$. It is straightforward to note that for $m = \ell$ these two values coincide and the final solution is the PC regression vector; this is the second point in the Remark 6.5. The first point in Remark 6.5 refers to the case when no preconditioning is used and consequently the PLS regression solution arises for $m \leq p$. Finally, for $m = \ell = p$ we get the well known statistical result which provides the equivalence between LS, PC and PLS regression.

Using preconditioning techniques in Krylov spaces allow to modify ℓ and m in order to get a whole set of subspaces and different bases from where the regression coefficient vector is approximated. The former (ℓ) controls the precondition operation, the latter (m) the Krylov expansion. The former regards PC while the latter regards PLS. These sets of approximations correspond indeed to different *ritz* approximations to the eigen pairs. The use of the former to reduce the dimension of the statistical problem naturally implies that $m << \ell$ especially in large data sets. Note furthermore that for $m > \ell$ the extracted bases are no more orthogonal.

Recall that a PLS regression solution based on m components is the vector $\widehat{\boldsymbol{\beta}}_m^{pls}$ which lies in a Krylov space of dimension equal to m, that is

$$\widehat{\boldsymbol{\beta}}_m^{pls} \in \mathcal{K}_m(\mathsf{b}, \mathsf{A}).$$

This allows to express the PLS regression vector in terms of a polynomial according to the following remark:

Remark 6.6 *Any vector* y *in* $\mathcal{K}_m(\mathsf{b}, \mathsf{A})$ *has a convenient representation in terms of a polynomial in* A*,*

$$\mathsf{y} = \sum_{i=0}^{m-1} (\mathsf{A}^i\ \mathsf{b})\gamma_i = \sum_{i=0}^{m-1} (\gamma_i\ \mathsf{A}^i)\ \mathsf{b} = \pi(\mathsf{A})\ \mathsf{b},$$

with $\pi(\xi) = \sum_i \gamma_i \xi^i$ *is a polynomial of degree less than* m*. Therefore we can write*

$$\mathcal{K}_m(\mathsf{b}, \mathsf{A}) = \left\{\pi(\mathsf{A})\ \mathsf{b} : \pi \in \mathcal{P}^{m-1}\right\},$$

where $\mathcal{P}^{m-1}$ *is the set of polynomials with degree less or equal to* $m-1$*.*

The polynomial expression for the PLS regression vector is given in Phatak and de Hoog (2001), and it will be seen shortly. For the moment recall expression (6.23) and that for an orthogonal matrix $\boldsymbol{M}$,

$$\mathcal{K}_m(\boldsymbol{M}\ \mathsf{b}, \boldsymbol{M}\ \mathsf{A}\ \boldsymbol{M}') = \boldsymbol{M}\ \mathcal{K}_m(\mathsf{b}, \mathsf{A}),$$

which fits very nice for $\boldsymbol{M} = \mathsf{W}'_\ell$ since the matrix W'_ℓ is orthogonal. Given Remark 6.6 and the expression (6.23) we can express the proposed solution, indicated by the superscript *, in a polynomial form according to

$$\widetilde{\widehat{\boldsymbol{\beta}}}^\star_{\ell\,|\,m} = \mathsf{W}'_\ell\,(\,\pi(\,\mathsf{A}\,)\,\mathsf{b}\,) \quad \text{with} \quad \pi \in \mathcal{P}^{m-1}. \tag{6.35}$$

Given that $\widehat{\boldsymbol{\beta}}^\star_{\ell\,|\,m} = \mathsf{W}_\ell\,\widetilde{\widehat{\boldsymbol{\beta}}}^\star_{\ell\,|\,m}$, we finally get

$$\widehat{\boldsymbol{\beta}}^\star_{\ell\,|\,m} = \mathsf{W}_\ell\,\mathsf{W}'_\ell\,(\,\pi(\,\mathsf{A}\,)\,\mathsf{b}\,) \quad \text{with} \quad \pi \in \mathcal{P}^{m-1}. \tag{6.36}$$

One should make the association between the expression $\boldsymbol{M}\,\mathsf{A}\,\boldsymbol{M}'$ in expression (6.23) and the preconditioned system on the transformed coordinates given in (6.31) and the Algorithm 6.15, for the choice $\boldsymbol{M} = \mathsf{W}'_\ell$. Note finally the the preconditioning matrix $\mathsf{Q} = \mathsf{W}'_\ell$ is not symmetric (as it has been Q in paragraph 6.3) and the NIPALS implementation is not trivial. Yet, the term $\mathsf{W}_\ell\,\mathsf{W}'_\ell$ in expression (6.36) provides a very suitable symmetric matrix in order to implement the NIPALS algorithm and to make the link with the Cyclic Subspace Regression (CSR).

6.4.3 NIPALS implementation and connections to CSR

The Cyclic Subspace Regression (CSR) is introduced in Lang et al. (1998) (see also Kalivas, 1999). It is essentially based on the same principle, that is, to use eigenvector basis sets in order to link PLS to PC regression. To do so CSR uses the Singular Value Decomposition of the data matrix $\boldsymbol{X}$, given by $\boldsymbol{X} = \mathsf{U}\,\boldsymbol{S}\,\mathsf{W}'$, where $\boldsymbol{S} = (\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_p})$ and the matrices U and W denote the left and right singular vectors. The CSR runs the NIPALS algorithm for PLS regression (see Wold, 1975) after projecting the response vector $\boldsymbol{y}$ on the space spanned by U, that is

$$\boldsymbol{y}^\star = \mathcal{P}_{\mathsf{U}}\,\boldsymbol{y} = \mathsf{U}\,\mathsf{U}'\,\boldsymbol{y}.$$

By retaining a restricted number of singular vectors in U, the CSR uses eigenvectors bases in order to condition the NIPALS algorithm. Note that by definition the CSR run always for a number of singular vectors larger or equal to the number of the PLS regression dimension.

Corollary 6.2 *The CSR coefficient vector results from the solution of the preconditioned system*

$$\mathsf{Q}\,\mathsf{A}\,\mathsf{z} = \mathsf{Q}\,\mathsf{b}, \;\; \textit{where} \;\; \mathsf{Q} = \mathsf{W}_\ell\,\mathsf{W}'_\ell.$$

The CSR algorithm follows the NIPALS algorithm for PLS regression, with

$$\widehat{\boldsymbol{\beta}}^{\star\,csr}_m = \mathsf{K}^\star_m(\mathsf{K}^{\star\prime}_m\,\mathsf{A}\,\mathsf{K}^\star_m)^{-1}\,\mathsf{K}^{\star\prime}_m\,\mathsf{b},$$

for $\mathsf{K}^\star_m$ a non-singular matrix with columns $\boldsymbol{v}_j$, $j = 1, \ldots, m$, for which

$$\mathsf{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) = \mathcal{K}_m(\,\mathsf{Q}\,\mathsf{b},\;\mathsf{Q}\,\mathsf{A}). \tag{6.37}$$

CSR follows the NIPALS structure for a starting vector depending upon U. Its component $\widetilde{\mathsf{b}} = \boldsymbol{X}'\boldsymbol{y}^{\star}$ is composed by the eigenvectors of matrix $\boldsymbol{X}\boldsymbol{X}'$. This can be computationally advantageous in the case where $n << p$. Yet, CSR needs to compute the score vectors, the loading vectors, and to deflate both $\boldsymbol{X}$ and $\boldsymbol{y}$ as in the NIPALS algorithm for PLS regression. The approach given in Algorithm 6.15 uses the Helland algorithmic structure and constructs directly ortonormal bases for the regression coefficient vector. Note finally that the preconditioned system in (6.31) is solved on the transformed coordinates of dimension ℓ rather than p.

We give in Algorithm 6.16 the NIPALS implementation for preconditioning sing eigenvectors which is directly related to the CSR.

Algorithm 6.16 Preconditioning Krylov spaces by eigenvectors (NIPALS)

Input: $\mathsf{A} = \boldsymbol{X}'\boldsymbol{X}$; $\boldsymbol{X}_0 \leftarrow \boldsymbol{X}$; $\boldsymbol{y}_0 \leftarrow \boldsymbol{y}$;

Step 1. Decompose A according to $\mathsf{A} = \mathsf{W}\,\boldsymbol{\Lambda}\,\mathsf{W}'$, and recover loading matrix

$$\mathsf{W}_\ell = (\mathsf{w}_1, ..., \mathsf{w}_\ell), \text{ for } \ell = 1, \ldots, p,$$

where w_ℓ is the eigenvector corresponding to the ℓ^{th} eigenvalue of A.

Step 2. Fix ℓ;

Step 3. For $m = 1, ..., p$,

- Compute the X-weight vectors according to:

$$\boldsymbol{d}^{\star}_{\ell\,|\,m} = \mathsf{W}_\ell\,\mathsf{W}'_\ell\,\boldsymbol{X}'_{m-1}\boldsymbol{y}_{m-1},$$

  normalize $\boldsymbol{d}^{\star}_{\ell\,|\,m}$ and store in $\boldsymbol{D}^{\star}_{\ell\,|\,m}$;
- Derive score vector according to: $\boldsymbol{t}^{\star}_{\ell\,|\,m} = \boldsymbol{X}_{m-1}\,\boldsymbol{d}^{\star}_{\ell\,|\,m}$ and store in $\boldsymbol{T}^{\star}_{\ell\,|\,m}$;
- Compute X-loading $\boldsymbol{p}^{\star}_{\ell\,|\,m} \propto \boldsymbol{X}'_{m-1}\boldsymbol{t}^{\star}_{\ell\,|\,m}$ and store in $\boldsymbol{P}^{\star}_{\ell\,|\,m}$;
- Compute Y-loading $q^{\star}_{\ell\,|\,m} \propto \boldsymbol{y}'_{m-1}\boldsymbol{t}^{\star}_{\ell\,|\,m}$ and store in $\boldsymbol{q}^{\star}_{\ell\,|\,m}$;
- Deflate data as:

$$\boldsymbol{X}_m = \boldsymbol{X}_{m-1} - \boldsymbol{t}^{\star}_{\ell\,|\,m}\,\boldsymbol{p}^{\star\prime}_{\ell\,|\,m} \quad \text{and} \quad \boldsymbol{y}_m = \boldsymbol{y}_{m-1} - \boldsymbol{t}^{\star}_{\ell\,|\,m}\,q^{\star}_{\ell\,|\,m}.$$

Output: Recover the regression coefficient vector according to

$$\widehat{\boldsymbol{\beta}}^{\star}_{\ell\,|\,m} = \boldsymbol{D}^{\star}_{\ell\,|\,m}\,(\boldsymbol{P}^{\star\prime}_{\ell\,|\,m}\,\boldsymbol{D}^{\star}_{\ell\,|\,m})^{-1}\,\boldsymbol{q}^{\star}_{\ell\,|\,m};$$

and give the final predictions according to : $\widehat{\boldsymbol{y}}^{\star}_{\ell\,|\,m} = \boldsymbol{X}\,\widehat{\boldsymbol{\beta}}^{\star}_{\ell\,|\,m}$.

Remark 6.7 *It is easy to verify the following properties for Algorithm 6.16:*
1. $\boldsymbol{t}^{\star}_{\ell\,|\,m} \perp \boldsymbol{t}^{\star}_{\ell\,|\,m'} \quad for \quad m \neq m'$.
2. $\boldsymbol{d}^{\star}_{\ell\,|\,m} \perp \boldsymbol{d}^{\star}_{\ell\,|\,m'} \quad for \quad m \neq m'$.
3. $\boldsymbol{t}^{\star\prime}_{\ell\,|\,m}\boldsymbol{X}_{m'} = 0$ *for* $m' > m$.
4. *The matrix* $\boldsymbol{R}^{\star}_{\ell\,|\,m} = \boldsymbol{P}^{\star\prime}_{\ell\,|\,m}\boldsymbol{D}^{\star}_{\ell\,|\,m}$ *is right bidiagonal.*
5. *The matrix* $\boldsymbol{R}^{\star\prime}_{\ell\,|\,m}\boldsymbol{R}^{\star}$ *is tridiagonal.*

Shrinkage aspects

PLS regression yield shrinkage estimates (see Goutis, 1996). Yet, their properties have been shown to be rather peculiar (see Butler and Denham, 2000). For a recall on shrinkage estimation and the shrinkage factor for LS and PC regression see Chapter 2. Phatak and de Hoog (2001) give the shrinkage factor for PLS regression as follows

$$f_m(\lambda_j) = 1 - \frac{\chi_m(\lambda_j)}{\chi_{m,0}}, \tag{6.38}$$

where $\chi_{m,0} = (-1)^m \det\left(\boldsymbol{H}^{(m)}\right)$ and $\chi_m(\lambda_j) = \det\left(\lambda_j\,\boldsymbol{I} - \boldsymbol{H}^{(m)}\right)$, for $\det(.)$ indicating the determinant of a matrix. The expressions above are the characteristic polynomial for the tridiagonal matrices $\boldsymbol{H}^{(m)}$ from where eigenvalues and eigenvectors of A are approximated. For notational ease we use $\widetilde{\chi}_m(\lambda_j)$ to denote the scaled polynomial for the second term in the right hand side of (6.38) for which $\widetilde{\chi}_m(0) = 1$. The peculiar shrinkage properties of PLS regression are due to this scaled polynomial.

We investigate the shrinkage aspect for the preconditioned solution. It seems that preconditioning by means of eigenvectors in matrix W_ℓ does not simplify the peculiar shrinkage properties of PLS regression. Using the Helland implementation in Algorithm 6.15 the solution is searched for on the transformed coordinates given in expression (6.31), while the final estimate is given by expression (6.36). This is equivalent to the following expression for the regression coefficient vector $\widehat{\boldsymbol{\beta}}^{\star}_{\ell\,|\,m}$ in terms of the scaled characteristic polynomial of A, that is

$$\widehat{\boldsymbol{\beta}}^{\star}_{\ell\,|\,m} = \mathsf{W}_\ell\,\mathsf{W}'_\ell\,\underbrace{\{\boldsymbol{I} - \widetilde{\chi}_m(\mathsf{A})\}\;\mathsf{A}^{-}}\,\mathsf{b}, \tag{6.39}$$

with the expression in the underbrace being the polynomial $\pi(\mathsf{A})$ of degree less than m in (6.36) (see also Phatak and de Hoog, 2001). Expression (6.39) is equal to

$$\widehat{\boldsymbol{\beta}}^{\star}_{\ell\,|\,m} = \mathsf{W}_\ell\,\mathsf{W}'_\ell\,\mathsf{A}^{-}\,\mathsf{b} - \mathsf{W}_\ell\,\mathsf{W}'_\ell\,\widetilde{\chi}_m(\mathsf{A})\,\mathsf{A}^{-}\,\mathsf{b}, \tag{6.40}$$

The first term in the right hand side in (6.40) can be seen as the PC-term of the regression vector. The second term in the right hand side in (6.40) expresses how the space generated by the ℓ principal components is modified by the PLS component. For $m = \ell$ expression (6.40) reduces to first term in the right hand

side. For $m = \ell = p$ this is the LS solution, while for $m = \ell < p$ this reduces to the PC regression solution. Finally, for $\ell = p$ we recover the PLS regression solution.

Corollary 6.3 *The shrinkage factor for the proposed regression coefficient is given by*

$$f_m(\lambda_j^\star) = 1 - \widetilde{\chi}_m(\lambda_j^\star), \tag{6.41}$$

for $\lambda_j^\star$ *indicating eigenvalues* λ_j *for* $j : \mathsf{w}_j \in \mathsf{W}_\ell$ *.*

The expression in (6.41) reveals the limitations of the proposed solution with regard to the shrinkage properties of the proposed method. That is, the shrinkage factor for the proposed solution is expected to retain the oscillating behavior of ordinary PLS regression because the shrinkage factor is still in a polynomial form. Therefore, excluding some eigen vectors from W_ℓ will not essentially change the shrinkage behavior for the new estimate. We will illustrate the above using examples in paragraph 6.4.4.

Dimension reduction and prediction performance

PLS is well know to build regression model of high predictive performance. Reducing the dimension of the regression problem in subspace iterative methods is strongly related to the decrease of the residual norm $\| \boldsymbol{r}_m \|$ resulting from the subspace approximations. Ilic and Turner (2005) have demonstrated that the norm of the residual vector after each iterative search in Krylov subspaces is inversely related to the determinant of the tridiagonal matrix $\boldsymbol{H}^{(m)}$ which corresponds to the product of the ritz values. Moreover they show that if $\boldsymbol{S}$ is any orthonormal basis for $\mathcal{K}_m(\mathsf{b}, \mathsf{A})$ then the determinant of $\boldsymbol{H}^{(m)}$ remains the same. By the proposed preconditioning method we seek to obtain better and faster solutions. By appropriately preconditioning Krylov subspaces $\mathcal{K}_m(\widetilde{\mathsf{b}}, \widetilde{\mathsf{A}})$ the decrease of the residual norm may take place faster than in the ordinary $\mathcal{K}_m(\mathsf{b}, \mathsf{A})$. This will be empirically studied in the examples that follow.

6.4.4 Examples

Longley data

The Longley data is a typical data set coming from macroeconomics. It is suitable here because it is relatively small and the results are easily tabulated. Moreover, the recorded variables are highly collinear. Biased regression such as PLS regression and PC regression are therefore commonly used. The data register consists of 6 macroeconomic variables such as the number of unemployed and the Gross National Product (GNP), while the response variable is the number of people employed. We use Multiple Linear regression (MLR), PLS regression, PC regression, and the proposed method (denoted by PREC) and we give in Table 6.2 the resulting regression coefficients.

Table 6.2: Longley data: Regression coefficients for MLR, PLSR, PCR, and PREC. By PREC we indicate the proposed method based on preconditioning $\mathcal{K}_m(\mathsf{b},\ \mathsf{A})_m$ by $\mathcal{E}_\ell$.

	m	ℓ	$\widehat{\beta}_1$	$\widehat{\beta}_2$	$\widehat{\beta}_3$	$\widehat{\beta}_4$	$\widehat{\beta}_5$	$\widehat{\beta}_6$	$\|\widehat{\boldsymbol{\beta}}\|$
MLR	–	–	0.01506	-0.03581	-0.02020	-0.01033	-0.05110	1.82915	1.83041
PLSR	6	–	0.01506	-0.03581	-0.02020	-0.01033	-0.05110	1.82915	1.83041
PREC	6	6	0.01506	-0.03581	-0.02020	-0.01033	-0.05110	1.82915	1.83041
PCR	3	3	0.00380	0.04017	-0.00804	-0.00486	0.00258	0.00165	0.04154
PLSR	2	–	0.00299	0.02882	0.00176	0.01067	0.00184	0.00129	0.03101
PREC	2	6	0.00299	0.02882	0.00176	0.01067	0.00184	0.00129	0.03101
PREC	1	3	0.00249	0.02330	0.01119	0.00758	0.00159	0.00109	0.02712
PREC	2	3	0.00297	0.02882	0.00176	0.01067	0.00186	0.00127	0.03100

A rich variety of approximations arises by the proposed method for different choice of ℓ and m. In Table 6.2 we give some cases. The MLR solution for example arises for $\ell = m = p$. We additionally give a single case for each one of the PC and PLS regression methods. That is, for $m = 6$ and $\ell = 3$ the solution coincides with that of ordinary PC regression on 3 components. Setting $m = 2$ and $\ell = 6$ we recover the ordinary PLS regression solution on 2 components. Ranging the values of m and ℓ we get a much more complete image on the regression estimates. Some intermediate cases are given in the final lines of Table 6.2. It is interesting to note the shrinkage behavior of the regression coefficients. Note especially the sixth component of the regression vector. Finally, in order to see shrinkage we included in the final column of Table 6.2 the norm of the regression coefficient vector which is always much smaller than the norm of the MLR vector. For the intermediate steps the difference on the norms are usually found on the last decimals.

We remain on the Longley data which we now use in order to explore the shrinkage behavior of the preconditioned solution. The studied cases are given in Table 6.3.

Table 6.3 confirms the oscillating behavior for the proposed regression coefficient as it has been already revealed by the expression (6.41). Note that in accordance to Butler and Denham (2000), for two (even number of) factor PLS model ($m = 2$) the solution expands for $j = 2$. Expanding shrinkage factors we also get for one component (odd number of) PLS model with $f(\lambda_1) = 1.0425$ (see Butler and Denham, 2000, Corollary 3). Finally note that removing eigenvectors from the preconditioning matrix sets the corresponding shrinkage factor to zero, and the solution reassembles to a mixture of PLS-PC regression.

Table 6.3: Longley data: Shrinkage factors $f(\lambda_j)$ for PC and PLS regression together with the shrinkage factors for the preconditioned solutions.

		j					
m	$\{\ell\}$	1	2	3	4	5	6
6	{1,...,6}	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
3	{1,...,6}	**1.0000**	0.9999	**1.0000**	0.0016	1.43e-04	2.94e-05
2	{1,...,6}	0.9955	**1.1620**	0.2826	0.0004	3.45e-05	7.06e-06
2	$\{1,2,3\}$	0.9955	**1.1620**	0.2826	4.71e-18	8.85e-19	3.69e-19
2	$\{1,2\}$	**1.0000**	**1.0000**	3.12e-16	4.00e-18	7.74e-19	2.46e-19
1	{1,...,6}	**1.0425**	0.4804	0.0817	1.11e-04	9.40e-06	1.92e-06
1	$\{1,2,3\}$	**1.0425**	0.4804	0.0817	1.22e-18	5.50e-19	1.23e-19

Octane data

The Octane data (already seen in Chapters 4 and 5) consists of 39 gasoline samples on which the digitized spectra of the octanes have been recorded at 225 wavelengths (in nm). Given these spectra for the 39 gasoline samples one seeks to model the octane concentration.

We split here the data on two sets. A training set was constructed including observations 1 to 20, and the test set included observations 21 to 39. Running PLS and PC regression we find that two components are essentially needed to construct the final regression models. This is due to the fact that both models for two components reach the minimum prediction loss. The latter was measured by means of the Root Mean Squared Error of Prediction (RMSEP) on the test set. A different split would not necessarily give the same results, especially for the Octane data set which is known to contain outliers (see Chapters 4 and 5). We are interested here on the analysis results given in Table 6.4.

Table 6.4: Octane data: Out-of-sample prediction loss (RMSEP) for regression models with $m, \ell = 2, 3, 4, 5, \ldots$. The training set includes observations 1 to 20, and the testing set includes observations 21 to 39.

	m					
ℓ	1	2	3	4	5	6
$\{1,\ldots,225\}$	3.6906	**1.2650**	11.2231	28.8451	38.9329	57.6231
$\{1,\ldots,5\}$	3.6970	**1.1172**	3.6367	15.2490	16.8166	
$\{1,\ldots,4\}$	3.6984	**1.1450**	1.2847	7.1303		
$\{1,2,3\}$	3.6945	**1.0320**	0.5124			
$\{1,2\}$	3.7021	**1.2045**				

Table 6.4 gives in bold letters the out-of-sample prediction error for for the

preconditioned solution for $m = 2$. The PLS regression on two components corresponds to $\ell = \{1, \ldots, 225\}$, while the diagonal elements ($m = \ell$) give the prediction loss for the PC regression model on the ℓ components. We use preconditioning in order to investigate the prediction loss for the regression models of dimension $m = 2$ with preconditioning matrix being constructed by the eigenvectors $(\mathsf{w}_1, \ldots, \mathsf{w}_\ell)$ for which we set $\ell = 2, \ldots, 6$. We focus attention on the effect on the prediction loss for $m = 2$ since this is the final model dimension for this split. It is very interesting to observe that retaining a small number of eigenvectors in the preconditioning matrix does not reduces further the dimension of the regression problem. Yet, the prediction performance of the constructed models is better since the prediction loss decreases, especially for $\ell = 3, 4, 5$. The minimum error is obtained for $\ell = 3$. Note finally that for $\ell = 3$ the results show a further gain in prediction but expansion of the dimension of the regression model to 3 components. By restricting the eigenspace on a low dimension for a fixed m we have given an example where we improve the prediction performance of the final model.

We have finally run the regression method for different combinations of $m = 1, \ldots, 6$ and $\ell = 1, \ldots, 6$. The coefficient lines between the wavelength range $(115, 190)$ are illustrated in Figure 6.6. It is not easy to distinguish the lines sice they lie very close one another. Yet, it is interesting to observe that for the first half of the wavelengths there is a strong separation between PLS and PC regression coefficients. Intermediate solutions disappear and cluster either with PLS or with PC regression coefficients. On the second half of the wavelengths the discrimination is much less visible between the two and intermediate solution appear.

6.4.5 Conclusions

PC and PLS regression coefficient vectors have been expressed in terms of Krylov and eigen subspace approximations. These were denoted by $\mathcal{K}_m$ and $\mathcal{E}_\ell$, respectively. Both regression coefficients reach the LS regression coefficient vector for $m = \ell = p$. PC regression has been seen as the limiting case depending on ℓ when $m = \ell$. The PLS regression coefficient vector is recovered by fixing $\ell = p$ and running through the Krylov spaces indexed by m. The intermediate solutions provide a whole set of approximations to the regression coefficient vector. The latter has been expressed in terms of a polynomial due to the property of the Krylov spaces. We have furthermore given a NIPALS based implementation for the proposed method and we have emphasized on its link with the Cyclic Subspace regression. The shrinkage aspects and the dimension reduction of the proposed method have been investigated in some limited real world data sets. The proposed solutions still retain the oscillating behavior of the PLS regression coefficient. Concerning the dimension reduction performance of the proposed method, in our limited experience it did not change by using eigenvectors to construct preconditioning matrices. The latter however may improve the final

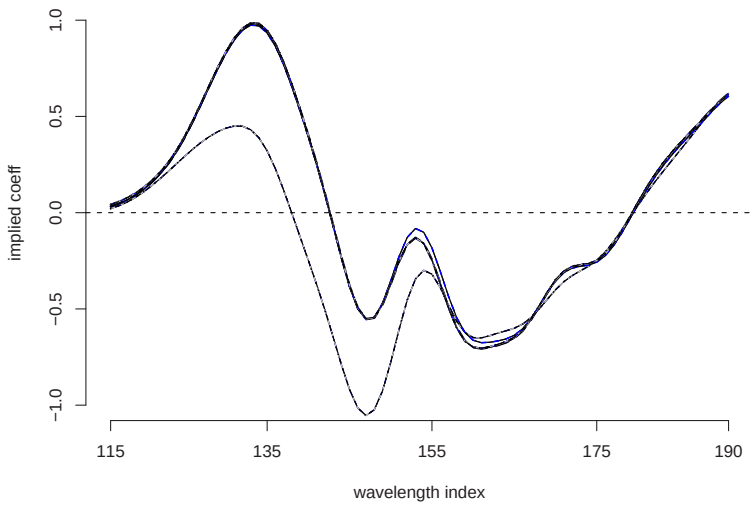


Figure 6.6: Octane data. Implied regression coefficients of different combinations for m and ℓ on the wavelength range 115-190nm. In the beginning of the illustrated wavelengths a strong separation between PLS and PC regression coefficients is clear. Intermediate solutions tend to cluster to PLS or PC regression results. In the second half of the wavelengths a less visible discrimination between the two is present.

prediction error for a fixed number of components. This has been verified in the examples, yet, it should be certainly tested on more data sets especially of high dimensions. Such data sets may yield matrices A with more complicated spectrum, and removing some clustered eigenvectors may further improve the prediction accuracy of the final model or even further reduce the dimension of the regression problem.

Chapter 7

Final conclusions and future research

Regularization methods in statistics is a constantly expanding research field. PLS methods are applied in regression in order to regularize the regression problem especially when the available data sets include many collinear predictors. The case where their number exceeds the available observations is at the forefront of exciting topics in international statistics research. PLS regression is widely applied to solve such problems and to reduce the dimension of the regression from a large number of predictors to a small number of derived components. This thesis has focused on the following topics:

Firstly, we attempted to replace the L_2 norm of PLS regression by the L_1 norm which is associated with the least absolute deviation regression (see also Dodge et al., 2004). Following an algorithm structurally similar to the PLS regression on orthogonal scores we have tried to model the median instead of the mean. The Partial LAD regression has been then tested and compared to PLS regression using the bootstrap (see also Kondylis and Whittaker, 2006). The comparison included real world data coming from NIR experiments and medicine, as well as experimental data. In the examples and experiments considered, we have established that Partial LAD and PLS estimate different features. This gives confidence in the basic structure of the Partial LAD algorithm Yet, unless a considerable amount of data is available, the PLS is superior to Partial LAD in the sense that the implied regression coefficient estimates have smaller bootstrap confidence intervals. Finally, a nice feature we observed in the experimental studies is that the Partial LAD regression is much more resistant to overfitting in comparison to PLS.

Secondly, we have used the multivariate BACON algorithm for outlier detection and robust estimation in the PLS framework. The use of the multivariate BACON algorithm led to the BACON PLS regression method which proposes an efficient and robust alternative to ordinary PLS (see also Kondylis and Hadi, 2006b,a). Robust methods for PC regression were also included. The presented

methods were tested on real and simulated data, on relatively low and high dimensional settings. The former were generated within a concrete statistical framework, the latter by following the bilinear factor model commonly used in PLS regression. For the high dimensional setting the multivariate BACON algorithm has been extended in order to detect outliers on rank deficient data. The use of the bilinear factor model has allowed to test the BACON extension on PLS regression problems with fixed and a priori known model dimension. The use of the BACON PLS regression for rank deficient data demonstrated the efficiency of the proposed method with respect to the regression coefficient loss, and protection of the model dimension against overfitting which is due to the outliers. Finally, a systematic comparison of the BPLS and the BPC regression with other robust proposals revealed that the BACON approach is much more efficient when no contamination is present, it is easy to compute compared to its competitors, while it is robust on reasonable levels of contamination.

Preconditioning Krylov spaces has been the subject of Chapter 6. Preconditioning methods for solving linear systems, their implementation on Krylov subspace research methods and their connection to PLS regression have been motivated in the beginning of this chapter. The use of such methods in the PLS framework has been illustrated in two cases.

1. We have proposed a method that improves the interpretation of the PLS regression coefficient vector. This has been achieved by downweighting predictors intervals, rather than individual variables, with no statistical relevance. The latter was measured via a relative importance statistic. In order to improve the interpretation of the PLS regression coefficient vector we have iteratively preconditioned the Krylov space by diagonal matrices with entries corresponding to the relative importance of the predictors. The solutions for the regression coefficient vector based on preconditioned Krylov subspace approximations allowed to retain the good prediction performance and the dimension reduction of PLS, while it made the interpretation of the final coefficient vector much more easy. The proposed method has been further motivated by a bayesian approach and a more concrete statistical interpretation. It has been finally tested on high dimensional data sets revealing very promising results.

2. We have furthermore used preconditioning methods to show the strong link between PLS and PC regression. Indeed, we have found PLS and PC regression estimates to be the limiting solutions when solving linear systems using preconditioned Krylov spaces. Using lower dimensional expansion either on eigen or Krylov space we have showed a rich ensemble of basis vector for the regression coefficient vector. We have then given the link between the proposed method and the Cyclic Subspace Regression. The dimension reduction and the shrinkage aspects of the proposed method for have been investigated in real world data sets. The mathematical development of the shrinkage factor of the presented method is still subject to further research.

The dimension reduction results from our experience on real world data sets did not show any significant improvement in dimension reduction. Yet while assessing the prediction performance for a given dimension showed that the prediction loss for the final model may attain lower levels.

The present work naturally creates more questions than the answers it was supposed to provide. The interpretation of PLS regression models, and the use of preconditioning methods in order to improve our knowledge and interpretation on the PLS regression coefficient vector has just only started. Imposing penalties on the weight vectors in PLS should be tested in the framework of preconditioning linear systems. The corresponding precondition matrix may therefore be more elaborated in order to avoid iterative schemes which make computation expensive. Additionally, a topic of future research is the use of the Krylov subspace approximation in getting inverse of singular matrices in the context of MCMC simulation for Bayesian algorithms such as the GIBBS sampling. The use of such approximations should be finally directed towards the Gene Regulatory Networks and graphical modelling when the data are rank deficient.

Mathematical Appendix I

The singular value decomposition

The singular value decomposition of the $n \times p$ matrix $\boldsymbol{X}$ is given according to

$$\boldsymbol{X} = \mathsf{U}\,\mathsf{S}\mathsf{W}', \tag{7.1}$$

for $\mathsf{S} = \mathsf{diag}(ß_1, \ldots, ß_p)$ a diagonal matrix with entries the singular values $ß_j$, for $j = 1, \ldots, p$, and $\mathsf{U}'\mathsf{U} = \mathsf{U}\mathsf{U}' = \mathbf{I}_n$, $\mathsf{W}'\mathsf{W} = \mathsf{W}\mathsf{W}' = \mathbf{I}_p$. The orthonormal matrices U and W are composed by the left and right singular vectors of $\boldsymbol{X}$, that is, $(\mathsf{u}_1, \ldots, \mathsf{u}_p)$ and $(\mathsf{w}_1, \ldots, \mathsf{w}_p)$, respectively.

The eigen decomposition

The eigen decomposition of a square symmetric matrix A is given according to

$$\mathsf{A} = \mathsf{W}\,\mathsf{diag}(\lambda_1, \ldots, \lambda_p)\,\mathsf{W}', \tag{7.2}$$

for λ_j and W denoting the ordered eigenvalues and the corresponding orthogonal matrix of eigenvectors. We then have :

Definition 7.1 *The space spanned by the $\ell \leq p$ eigenvectors corresponding to the first ℓ eigenvalues $(\lambda_1, \ldots, \lambda_\ell)$ in (7.2) is called an eigenspace of dimension ℓ, we write*

$$\mathsf{span}\,(\mathsf{w}_1, \ldots, \mathsf{w}_\ell) = \mathcal{E}_\ell\,(\mathsf{A}). \tag{7.3}$$

Computing a dominant eigenpair $(\lambda_1, \mathsf{w}_1)$

Algorithm 7.17 Power Method for computing a dominant eigenpair $(\lambda_1, \mathsf{w}_1)$

Intput: A $(p \times p)$ matrix A, a vector z_0;
For $k = 1, 2, 3, \ldots$ **do**
  compute $\mathsf{y}_k = \mathsf{A}\,\mathsf{z}_{k-1}$;
  normalize $\mathsf{z}_k = \mathsf{y}_k / \|\,\mathsf{y}_k\|$;
  test convergence on z_k;
Output: At convergence: $\lambda_1 = \|\,\mathsf{z}_k\|$ and $\mathsf{w}_1 = \mathsf{A}\,\mathsf{z}_k$.

1. If there is a unique dominant λ_1 and $\mathsf{w}_1' \mathsf{z}_0 \neq 0$ then in Algorithm 7 we have : $\mathsf{z}_k \to \mathsf{w}_1$ linearly with convergence factor $\max\{\lambda_{p-1}/\lambda_p, |\lambda_1|/\lambda_p\}$.

2. The Inverse Iteration Method (IIM) is the power method applied on the matrix A^{-1} instead of A. No need to inverse A, just replace $\mathsf{y}_k = \mathsf{A}\,\mathsf{z}_{k-1}$ above by
$$\text{solve for } \mathsf{y}_k \ : \ \mathsf{A}\,\mathsf{f}_k = \mathsf{z}_{k-1}.$$
Its linear convergence is given by the factor $|\lambda_1/\lambda_2|$.

3. The Rayleigh Quotient Iteration (RQI) is another powerful method to solve the same problem. It is based on the Rayleigh Quotient, given by
$$\rho(\mathsf{z}) = \mathsf{z}'\,\mathsf{A}\,\mathsf{z}/\,\mathsf{z}'\,\mathsf{z}.$$
Convergence $(\rho_k, \mathsf{z}_k) \to (\lambda_1, \mathsf{w}_1)$ is reached cubically.

Deflation

The methods described above to compute a dominant eigenpair $(\lambda_1, \mathsf{w}_1)$ can be removed from $\mathsf{S}_{(n\times n)}$ in order to go on for new information without the risk to compute over again the same quantities.

For example we can remove the first dominant eigenpair $(\lambda_1, \mathsf{w}_1)$ for matrix $\mathsf{A} \equiv \mathsf{A}_{old}$ detected by the power method and go on searching the dominant eigenpair for the residual matrix S_{new}.

Remark 7.1 *Deflation is often carried out by subtraction, by restriction, and by similarity transormations.*

1. Subtraction - Orthogonalization: Work on the matrix S_{new} given by
$$\mathsf{A}_{new} = \mathsf{A}_{old} - \lambda_1\,\mathsf{w}_1\,\mathsf{w}_1' = \sum_{i=2}^{p} \lambda_j\,\mathsf{w}_j\,\mathsf{w}_j'.$$

2. Restriction: Work on $\mathsf{A}_{new} = \mathsf{A}^{\perp}$, that is the restriction of A_{old} to the space orthogonal to w_1. It has all eigenpairs $(\lambda_j, \mathsf{w}_j)$ except for the first one.

3. Similarity transformation may also be used to deflate. We find a $(n-1)\times(n-1)$ matrix which represents $\mathsf{A}^{\perp}$, and we work on it. For an orthogonal matrix $\boldsymbol{P}$ with first column the $\boldsymbol{u}_1$ (the 1st eigenvalue) we can write
$$\boldsymbol{P}'\,\mathsf{A}\boldsymbol{P} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \boldsymbol{C} \end{bmatrix},$$

and $\boldsymbol{C}$ is an adequate representation of $\mathsf{A}^{\perp}$. If the off diagonal elements are not close to zero, we continue with similarity transformations induced by matrices $\boldsymbol{Q}, \boldsymbol{W}$, etc. which result in

$$\boldsymbol{W}'\boldsymbol{Q}'\boldsymbol{P}'(\,\mathsf{A}\,)\,\boldsymbol{P}\boldsymbol{Q}\boldsymbol{W}.$$

Krylov spaces

Definition 7.2 *For a nonzero vector* b *and a square matrix* A*, the matrix of the form*

$$\mathsf{K}_m(\,\mathsf{b},\ \mathsf{A}) = (\,\mathsf{b},\,\mathsf{A}\,\mathsf{b},\,\mathsf{A}^2\,\mathsf{b},\ldots,\,\mathsf{A}^{m-1}\,\mathsf{b}), \tag{7.4}$$

is called a Krylov matrix*.*

Definition 7.3 *The space spanned by* $\{\,\mathsf{b},\ \mathsf{A}^1\,\mathsf{b},\ldots,\,\mathsf{A}^{m-1}\,\mathsf{b}\}$ *is called a Krylov space of dimension* $m \leq p$*, we write*

$$\mathsf{span}\,(\,\mathsf{b},\ \mathsf{A}^1\,\mathsf{b},\ldots,\,\mathsf{A}^{m-1}\,\mathsf{b}\,) = \mathcal{K}_m(\,\mathsf{b},\ \mathsf{A}\,). \tag{7.5}$$

Basic Properties:

1. Scale invariance: $\mathcal{K}_m(\sigma\ \mathsf{b}, \tau\,\mathsf{A}) = \mathcal{K}_m(\,\mathsf{b},\,\mathsf{A})\quad$ for $\quad\sigma,\,\tau \neq 0$;
2. Translation invariance: $\mathcal{K}_m(\,\mathsf{b},(\,\mathsf{A} - \tau\boldsymbol{I})) = \mathcal{K}_m(\,\mathsf{b},\,\mathsf{A})$;
3. Change of basis: $\mathcal{K}_m(\boldsymbol{M}\ \mathsf{b}, \boldsymbol{M}\,\mathsf{A}\boldsymbol{M}') = \boldsymbol{M}\,\mathcal{K}_m(\,\mathsf{b},\,\mathsf{A})\quad$ for $\quad\boldsymbol{M}' = \boldsymbol{M}^{-1}$.
4. Any vector in $\mathcal{K}_m(\,\mathsf{b},\,\mathsf{A})$ has a convenient representation in terms of a polynomial in A. Therefore we can write

$$\mathcal{K}_m(\,\mathsf{b},\,\mathsf{A}) = \left\{\pi(\,\mathsf{A})\ \mathsf{b} : \pi \in \mathcal{P}^{m-1}\right\},$$

where $\mathcal{P}^{m-1}$ is the set of polynomials with degree less equal $m-1$.

Definition 7.4 *Define as algebraic grade of a Krylov space denoted by* $m^{\star}$ *the lowest value of* m *for which*

$$\dim\mathcal{K}_{m^{\star}+1}(\,\mathsf{b},\ \mathsf{A}\,) = \dim\mathcal{K}_{m^{\star}}(\,\mathsf{b},\ \mathsf{A}\,),$$

with dim *denoting the dimension of the generated space.*

Tridiagonal matrices and the Lanczos method

Tridiagonal matrices

Definition 7.5 *A square symmetric matrix of the following form*

$$\mathsf{Trd}_j = \begin{bmatrix} \alpha_1 & \beta_1 & & \dots & & 0 \\ \beta_1 & \alpha_2 & \beta_2 & & & \\ & \beta_2 & \alpha_3 & & & \vdots \\ \vdots & & & \ddots & & \\ & & & & \ddots & \beta_{j-1} \\ 0 & & \dots & & \beta_{j-1} & \alpha_j \end{bmatrix}.$$

is called as tridiagonal matrix. It is unreduced if and only if all subdiagonal elemnts are different than zero, that is, $\beta_j \neq 0$.

Proposition 7.1 *(page 124 Parlett, 1980)*
The eigenvalues of an unreduced tridiagonal matrix Trd *are distinct.*

Krylov spaces, RR procedure and the Lanczos method

Remark 7.2 *The Lanczos method applies the Rayleigh-Ritz procedure on the sequence of Krylov spaces* K_j*, with* $j = 1, \dots, m$.

The 1st step in Rayleigh-Ritz (RR) procedure is to define an orthonormal basis for $\mathcal{K}_m(\mathsf{b}, \mathsf{A})$.

- We have the Krylov sequence $\mathsf{K}_j(\mathsf{b}, \mathsf{A})$ which we simply denote as K_j with $j = 1, \dots, m$. We can write

$$\mathsf{K}_j = \boldsymbol{Q}_j \, \boldsymbol{R}_j^{-1},$$

with $\boldsymbol{R}_j^{-1}$ an $j \times j$ upper triangular matrix [1]. The matrix $\boldsymbol{Q}_j$ consists of $(\boldsymbol{q}_1, \dots, \boldsymbol{q}_j)$ with $\boldsymbol{q}$'s being orthonormal vectors (**Lanczos vectors**) which form an orthonormal basis for K_j. This represents a kind of QR factorization of K_j arising from a Gram-Schmidt orthonormalization of the j vectors of the Krylov sequence K_j.[2]

The 2nd step in RR procedure consists in computing the rayleigh quotient matrix.

[1] The columns of $\boldsymbol{R}_j$ contain the coefficients of the **Lanczos polynomials**.

[2] Generally applying the Gram-Schmidt orthonormalization is a burden. Yet, here because of the Krylov sequence K_j the process simplifies significantly and we get a three-term recurrence conecting the columns of $\boldsymbol{Q}_j$ arising from the tridiagonal Trd.

- Projecting A on the Krylov space spanned by the $\boldsymbol{q}_j$'s, that is $\mathsf{K}_j(\boldsymbol{q}_1,\ \mathsf{A})$, it gives the tridiagonal matrix Trd_j.

$$\rho(\boldsymbol{Q}_j) = \boldsymbol{Q}_j' \,\mathsf{A}\, \boldsymbol{Q}_j = \mathsf{Trd}_j = \begin{bmatrix} \alpha_1 & \beta_1 & & . & 0 \\ \beta_1 & \alpha_2 & \beta_2 & . & . \\ . & \beta_2 & ... & ... & . \\ . & & ... & ... & \beta_{j-1} \\ 0 & . & . & \beta_{j-1} & \alpha_j \end{bmatrix}.$$

The 3rd step in RR procedure consists in computing the eigen pairs of the tridiagonal matrix Trd_j *given by* $(\theta_i^{(j)}, \boldsymbol{s}_i^{(j)})$.

- The pairs $(\theta_i^{(j)}, \boldsymbol{s}_i^{(j)})$ are easily obtained by the tridiagonal matrix Trd_j. From the RR procedure it is seen that the pairs $(\theta_i^{(j)}, \boldsymbol{\phi}_i^{(j)})$ with $\boldsymbol{\phi}_i^{(j)} = \boldsymbol{Q}_j\, \boldsymbol{s}_i^{(j)}$ with $i = 1, \ldots, j$ are the best set of approximations to the eigenpairs of A that can be obtained by the subspace $\mathcal{K}_j$.

The Lanczos Algorithm

Algorithm 7.18 Lanczos algorithm

Output: $\boldsymbol{r}_0$ and $\beta_0 = \|\boldsymbol{r}_0\| \neq 0$;
For $j = 1, 2, 3, \ldots$ **do**
Step 1: $\boldsymbol{q}_j \leftarrow \boldsymbol{r}_{j-1}/\beta_{j-1}$;
Step 2: $\boldsymbol{u}_j \leftarrow \mathsf{A}\, \boldsymbol{q}_j$;
Step 3: $\boldsymbol{r}_j \leftarrow \boldsymbol{u}_j - \boldsymbol{q}_{j-1}\, \beta_{j-1}$; $\quad (\boldsymbol{q}_0 = 0)$
Step 4: $\alpha_j \leftarrow \boldsymbol{q}_j'\, \boldsymbol{r}_j$;
Step 5: $\boldsymbol{r}_j \leftarrow \boldsymbol{r}_j - \boldsymbol{q}_j\, \alpha_j$;
Step 6: $\beta_j \leftarrow \|\boldsymbol{r}_j\| = \|\boldsymbol{q}_{j+1}\, \beta_j\|$;
Output: The eigen pair $(\theta_i^{(j)}, \boldsymbol{s}_i^{(j)})$ or $(\theta_i^{(j)}, \boldsymbol{\phi}_i^{(j)})$.

- *Step 1* determines the orthonormal basis for $\mathcal{K}_{j-1}$, which is easily computed since we only need $\boldsymbol{q}_j$.
- *Steps 2,3,4,5* compute the rayleigh quotient $\boldsymbol{Q}_j' \,\mathsf{A}\, \boldsymbol{Q}_j$using the recurrence

$$\mathsf{A}\, \boldsymbol{q}_j = \boldsymbol{q}_{j+1}\beta_j + \boldsymbol{q}_j\alpha_j + \boldsymbol{q}_{j-1}\beta_{j-1}.$$

- *Step 6* recycles the algorithm.
- The *output* is the eigenpairs $(\theta_i^{(j)}, \boldsymbol{s}_i^{(j)})$ used to compute the *ritz* pairs $(\theta_i^{(j)}, \boldsymbol{\phi}_i^{(j)})$.

- The Krylov sequence K does not appear in the algorithm. The matrices $\boldsymbol{Q}_j$ and Trd_j are built directly from A and $\boldsymbol{q}_1$.
- An important restriction is the case where z is orthogonal to A. In such cases the rayleigh quotient ρ can not detect eigenpairs based on Krylov spaces.
- Finally, when z is an eigenvector of A the Lanczos stops after $j = 1$.

Some final remarks on Lanczos methods and Krylov spaces

Remark 7.3 *By choosing the Krylov subspace as the basis of the RR procedure, we obtain a sequence of tridiagonal matrices* Trd_j*, for* $j = 1, \ldots, m$*,* [3] *with the RR property that the eigenvalues of* Trd_j *are the best set of approximations to the eigenpairs of* A*.*

Remark 7.4 *The Lanczos method is a useful technique to approximate extremal eigenvalues of large and sparse symmetric matrices.*

Remark 7.5 *The extremal eigenvalues of the tridiagonal matrix* Trd_j *converge to the extremal eigenvalues of* A *well before* j *reaches the dimension of* A*.*

Remark 7.6 *The Lanczos method is nothing more than the RR approximation from a Krylov subspace.*

Krylov spaces and Conjugate Gradients

Conjugate Gradient (CG) methods are very effective to solve large and sparse systems of linear equations given by

$$\mathsf{A}\,\mathsf{x} = \mathsf{b}. \tag{7.6}$$

For an initial guess x_0, the solution is approximated as

$$\mathsf{x}_{m+1} = \mathsf{x}_m + \boldsymbol{d}_m$$

by the minimization of the quadratic function

$$\phi(\mathsf{x}) = \frac{1}{2}\,\mathsf{x}'\,\mathsf{A}\,\mathsf{x} - \mathsf{b}'\mathsf{x}, \tag{7.7}$$

over a sequence of spaces of increasing dimension. These are Krylov spaces of dimension $m + 1$, that is

$$\mathcal{K}_{m+1}(\mathsf{x}_0,\, \mathsf{A}). \tag{7.8}$$

[3] Tridiagonal matrices have a polynomial representation just like Krylov sequences, see Parlett(1980), chapter 7.

The minimization of (7.7) corresponds to the solution of the linear system in (7.6). This is verified by noting that the minimum in (7.7) is reached at $-1/2\ \mathsf{b}'\ \mathsf{A}^{-1}\ \mathsf{b}$ which is achieved if one sets $\mathsf{x} = \mathsf{A}^{-1}\ \mathsf{b}$. Therefore minimizing (7.7) and solving (7.6) are equivalent.

The space in (7.8) is generated by the Krylov sequence

$$\mathsf{K}_{m+1} = (\mathsf{x}_0,\ \mathsf{A}^1\mathsf{x}_0,\ \mathsf{A}^2\mathsf{x}_0, \ldots,\ \mathsf{A}^m\mathsf{x}_0).$$

Hence, $\boldsymbol{d}_m \in \mathcal{K}_{m+1}(\mathsf{x}_0,\ \mathsf{A})$ and $\boldsymbol{d}_m$ is chosen in order to have $\{\ \boldsymbol{d}_m : \min \|\mathsf{x}\|_{\mathsf{A}}\ \}$. Vectors $\boldsymbol{d}_j$ with $j = 1, \ldots, m, \ldots, p$ are conjugate orthogonal with respect to A, that is

$$\boldsymbol{d}_i'\,\mathsf{A}\,\boldsymbol{d}_j = 0, \quad \text{for} \quad i < j,$$

which is usually denoted as $\boldsymbol{d}_i \perp_{\mathsf{A}} \boldsymbol{d}_j$.

The vector $\boldsymbol{r} = \mathsf{b} - \mathsf{A}\,\mathsf{x}$ is the gradient vector of the quadratic function in (7.7) and it is called the residual vector. It can be also shown that

$$\mathsf{span}(\boldsymbol{d}_1, \ldots, \boldsymbol{d}_m) = \mathsf{span}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_m) = \mathcal{K}_{m+1}.$$

The canonical algorithm of CG (see Hestens and Stiefel, 1952) which is given below is reproduced from Phatak and de Hoog (2001):

1. Initialize with : $\mathsf{x}_0 = 0,\ \boldsymbol{d}_0 = \boldsymbol{r}_0 = \mathsf{b} - \mathsf{A}\,\mathsf{x}_0 = \mathsf{b}$;
2. $\alpha_i = \dfrac{\boldsymbol{d}_i'\,\boldsymbol{r}_i}{\boldsymbol{d}_i'\,\mathsf{A}\,\boldsymbol{d}_i}$;
3. $\mathsf{x}_{i+1} = \mathsf{x}_i + \alpha_i\,\boldsymbol{d}_i$;
4. $\boldsymbol{r}_{i+1} = \mathsf{b} - \mathsf{A}\,\mathsf{x}_{i+1}$, $(= \boldsymbol{r}_i - \alpha_i\ \mathsf{A}\,\boldsymbol{d}_i)$;
5. $b_i = -\dfrac{\boldsymbol{r}_{i+1}'\,\mathsf{A}\,\boldsymbol{d}_i}{\boldsymbol{d}_i'\,\mathsf{A}\,\boldsymbol{d}_i}$;
6. $\boldsymbol{d}_{i+1} = \boldsymbol{r}_{i+1} + b_i\,\boldsymbol{d}_i$.

Bibliographic notes

For Krylov spaces, the Lanczos algorithm and the computation of eigenpairs a huge amount of literature exists. The above points are extracted mainly from Parlett (1980). The Lanczos method is due to Lanczos (1950). Krylov iterative methods are described in Ruhe (1998); De Sturler (1999). A very nice article on the idea behind Krylov methods is Ipsen and Meyer (1998). Numerous articles on CG methods exist in the literature with different implementations for the CG algorithm. For an overview on CG and a taxonomy of different CG algorithms one can see Hestens and Stiefel (1952); Golub and Van Loan (1996) and Ashby et al. (1990).

Mathematical Appendix II

Proof of Proposition 3.4

The proof is given geometrically starting from Figure 7.1 which is reproduced from Goutis (1996).

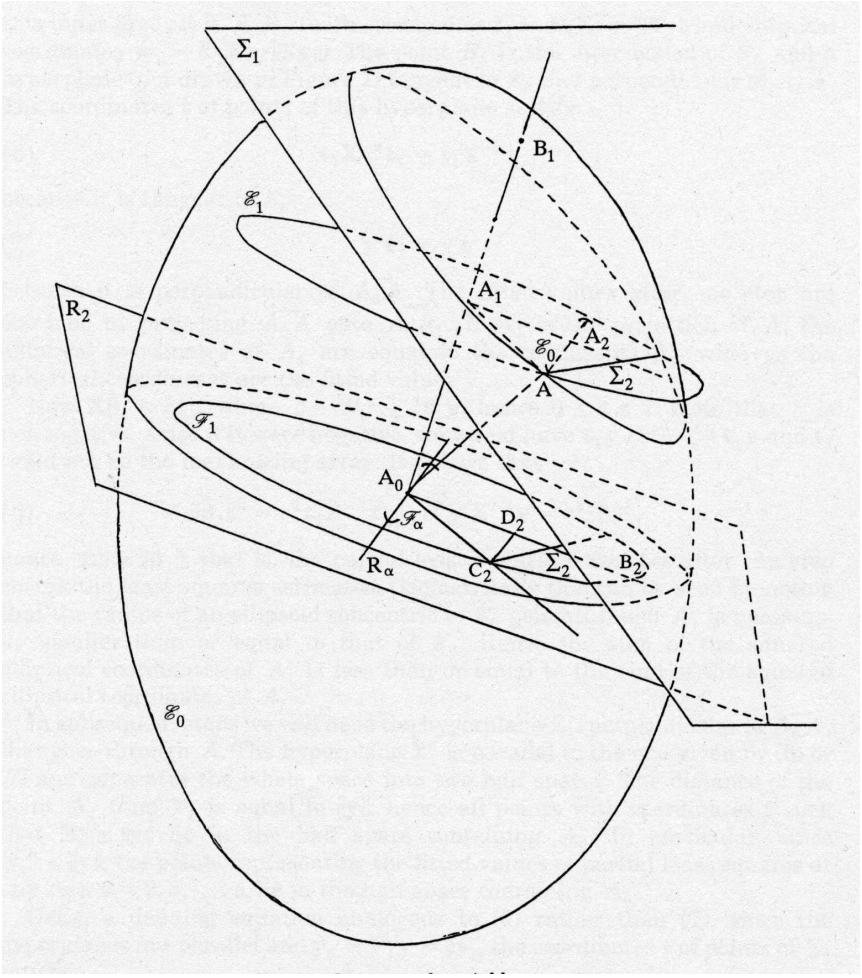


Figure 7.1: Geometry of Partial Least Squares. Figure reproduced from Goutis (1996).

Figure 7.1 illustrates the geometry of PLS after the regression problem has been transformed to its canonical form by applying $\boldsymbol{X} = \boldsymbol{P}' \,\mathsf{diag}(d_1 \ldots, d_p)\boldsymbol{Q}$, for orthogonal matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$. Note that the ellipsoid $\mathcal{E}_0$ has axes with lengths proportional to the diagonal elements $d_1, \ldots, d_p$. The least squares fitted values are represented by the point A lying on the ellipsoid $\mathcal{E}_0$. Goutis (1996) shows geometrically that

$$\text{For every } \alpha \leq p \text{ we get } \|\widehat{\boldsymbol{\beta}}\| \leq \|\widehat{\boldsymbol{\beta}}_\alpha\|.$$

To demonstrate the above he notes that all vectors with an endpoint in a hyperplane perpendicular to $\overrightarrow{A_0A}$ have the same inner product with $\overrightarrow{A_0A}$. The normalization of the weight vectors $\boldsymbol{w}$ in the PLS regression algorithm restricts the vectors to have an endpoint on the ellipsoid $\mathcal{E}_0$. In Figure 7.1 this vector is the $\overrightarrow{A_0B_1}$. The point B_1 is the intersection of the ellipsoid $\mathcal{E}_0$ and a hyperplane tangent to $\mathcal{E}_0$ and perpendicular to $\overrightarrow{A_0A}$ (this stems from the geometry of Krylov sequences). Note that for the next iteration the dimension reduces to $r-1$. Initially $r = 3$ so now we are in a surface. The new surface is perpendicular to $\overrightarrow{A_0B_1}$ due to orthogonality constraints on the weights and the score vectors in the PLS regression algorithm. It furthermore passes from point A, and the ellipsoid now corresponds to $\mathcal{E}_1$ (note that $\mathcal{F}_1$ is just the same ellipsoid centered at A_0). In the second iteration we seek for the direction vector with an endpoint in $\mathcal{F}_1$ that maximizes the inner product with $\overrightarrow{A_0C_2}$. This is the vector $\overrightarrow{A_0B_2}$. Projecting C_2, or equivalently A (deflation is optional for univariate PLS), onto $\overrightarrow{A_0B_2}$ we obtain D_2. By adding $\overrightarrow{A_0A_1}$ and $\overrightarrow{A_0D_2}$ we obtain $\overrightarrow{A_0A_2}$. The process is repeated for further dimensions.

The proof in Proposition 3.4 stems from the fact that for $\boldsymbol{X}'\boldsymbol{X} \propto \boldsymbol{I}_p$ the problem can be set in the canonical form by applying $\boldsymbol{X} = \boldsymbol{P}' \,\mathsf{diag}(d_1 \ldots, d_p)\boldsymbol{Q}$, just like above. Yet, in this case $\mathcal{E}$ is no more an ellipsoid but a sphere. In this case the vector with an endpoint in a hyperplane perpendicular to $\overrightarrow{A_0A}$ which have the same inner product with $\overrightarrow{A_0A}$ under the normalization constraint for the weight vectors $\boldsymbol{w}$ in the PLS regression algorithm is the vector $\overrightarrow{A_0A}$. See that now the vector $\overrightarrow{A_0B_1}$ is replaced by $\overrightarrow{A_0A}$. Projecting this vector on the direction of $\overrightarrow{A_0A}$, in order to take the next iterations, means projecting $\boldsymbol{y}$ on itself. Then PLS regression solution reaches the OLS solution after only one step.

Proof of Proposition 6.1

It should be proven that

$$\widehat{\boldsymbol{\beta}}^\star_m = \mathsf{K}^\star_m(\mathsf{K}^{\star\prime}_m \,\mathsf{A}\,\mathsf{K}^\star_m)^{-1}\,\mathsf{K}^{\star\prime}_m\,\mathsf{b}. \tag{7.9}$$

for $\mathsf{K}^\star_m$ a non-singular matrix $\boldsymbol{V}_m = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)$ such that

$$\mathsf{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) = \mathcal{K}_m(\widetilde{\mathsf{A}}, \widetilde{\mathsf{b}}). \tag{7.10}$$

Recall that in the NIPALS algorithm (see Wold, 1975) the PLS regression coefficient vector is given by the regression of the PLS components $\boldsymbol{T}_m = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_m)$ on the response $\boldsymbol{y}$ using ordinary least squares regression. The regression coefficient vector equals to

$$\widehat{\boldsymbol{q}}_m = (\boldsymbol{T}'_m \boldsymbol{T}_m)^{-1} \boldsymbol{T}'_m \boldsymbol{y},$$

and the PLS regression coefficient vector is the recovered in terms of the deflated data $\boldsymbol{X}_{m-1}$ according to $\widehat{\boldsymbol{\beta}}^{pls}_m = \boldsymbol{W}_m \boldsymbol{q}_m$, or in terms of the original data $\boldsymbol{X}_0$ according to

$$\widehat{\boldsymbol{\beta}}^{pls}_m = \widetilde{\boldsymbol{W}}_m \widehat{\boldsymbol{q}}_m,$$

with $\widetilde{\boldsymbol{W}}_m = \boldsymbol{W}_m (\boldsymbol{P}'_m \boldsymbol{W}_m)^{-1}$ indicating the weight vector's matrix $\boldsymbol{W}_m = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m)$ expressed in terms of the original predictors and $\boldsymbol{P}_m$ the matrix with the PLS regression loadings.

We start by the general case. Let Q be a $p \times p$ non-singular matrix, and define the matrix $\widetilde{\mathsf{A}} = \mathsf{Q}\,\mathsf{A}$ and the vector $\widetilde{\mathsf{b}} = \mathsf{Q}\,\mathsf{b}$. The NIPALS algorithm for preconditioned PLS regression is based on the following operations:

$$\boldsymbol{v}_m = \mathsf{Q}\,\boldsymbol{X}'_{m-1}\boldsymbol{y} \text{ and } \boldsymbol{t}_m = \boldsymbol{X}_{m-1}\,\boldsymbol{v}_m, \text{ for } \boldsymbol{X}_m = (\mathbf{I}_n - \mathcal{P}_{T_m})\,\boldsymbol{X}_0,$$

with $\mathcal{P}_{T_m}$ denoting the projector on $\boldsymbol{T}_m$. For the weight vectors $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)$ of the preconditioned PLS regression we show that

$$\mathsf{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) = \mathcal{K}_m(\widetilde{\mathsf{b}}, \widetilde{\mathsf{A}}).$$

We use induction,

$$\begin{aligned}
\text{for } m = 1, \quad \boldsymbol{v}_1 &= \mathsf{Q}\,\boldsymbol{X}'\boldsymbol{y} = \mathsf{Q}\,\mathsf{b} = \widetilde{\mathsf{b}}, \qquad (7.11)\\
\text{for } m \geq 2, \quad \boldsymbol{v}_m &= \mathsf{Q}\,\boldsymbol{X}'_{m-1}\boldsymbol{y}\\
&= \mathsf{Q}\,\left[(\boldsymbol{I} - \mathcal{P}_{T_{m-1}})\,\boldsymbol{X}_0\right]'\boldsymbol{y}\\
&= \mathsf{Q}\,\left[\boldsymbol{X}'_0\,(\boldsymbol{I} - \mathcal{P}_{T_{m-1}})'\right]\boldsymbol{y}\\
&= \mathsf{Q}\,\boldsymbol{X}'_0\,\boldsymbol{y} - \mathsf{Q}\,\boldsymbol{X}'_0\,\mathcal{P}'_{T_{m-1}}\,\boldsymbol{y}\\
&= \mathsf{Q}\,\boldsymbol{X}'_0\,\boldsymbol{y} - \mathsf{Q}\,\boldsymbol{X}'_0\,\mathcal{P}_{T_{m-1}}\,\boldsymbol{y}\\
&= \mathsf{Q}\,\boldsymbol{X}'_0\,\boldsymbol{y} - \mathsf{Q}\,\boldsymbol{X}'_0\,\boldsymbol{T}_{m-1}\,(\boldsymbol{T}'_{m-1}\,\boldsymbol{T}_{m-1})^{-1}\,\boldsymbol{T}'_{m-1}\,\boldsymbol{y}\\
&= \mathsf{Q}\,\boldsymbol{X}'_0\,\boldsymbol{y} - \mathsf{Q}\,\boldsymbol{X}'_0\,\boldsymbol{X}_0\,\widetilde{\boldsymbol{W}}_{m-1}\,\widehat{\boldsymbol{q}}_{m-1}\\
&= \mathsf{Q}\,\boldsymbol{X}'_0\,\boldsymbol{y} - \mathsf{Q}\,\boldsymbol{X}'_0\,\boldsymbol{X}_0\,\widehat{\boldsymbol{\beta}}_{m-1}\\
&= \widetilde{\mathsf{b}} - \widetilde{\mathsf{A}}\,\widehat{\boldsymbol{\beta}}_{m-1}. \qquad (7.12)
\end{aligned}$$

Proof of Lemma 6.1

1. For the first point one should only note that the following properties still hold:

   (a) $\boldsymbol{X}_i \boldsymbol{w}^\star_j = 0$ for $i > j$,

(b) $\boldsymbol{t}_i^{\star\prime}\boldsymbol{X}_j = 0$ for $i-1 < j$.

We have for (a)

$$\boldsymbol{X}_i\boldsymbol{w}_j^{\star} = \left[\boldsymbol{X}_{i-1} - \mathcal{P}_{t_i^{\star}}\boldsymbol{X}_{i-1}\right]\boldsymbol{w}_j^{\star},$$

with $\mathcal{P}_{t_i^{\star}}$ denoting the projector matrix. Given that $j < i$ we get

$$\boldsymbol{X}_j\boldsymbol{w}_j^{\star} - \mathcal{P}_{t_j^{\star}}\boldsymbol{X}_j\boldsymbol{w}_j^{\star} = \boldsymbol{t}_j^{\star} - \mathcal{P}_{t_j^{\star}}\boldsymbol{t}_j^{\star} = \boldsymbol{t}_j^{\star} - \boldsymbol{t}_j^{\star} = 0.$$

For (b) we have

$$\boldsymbol{t}_i^{\star\prime}\boldsymbol{X}_j = \boldsymbol{w}_i^{\star\prime}\boldsymbol{X}_{i-1}'\boldsymbol{X}_j = 0 \quad \text{for} \quad i-1 < j.$$

2. $\boldsymbol{R}^{\star\prime}\boldsymbol{R}^{\star}$ is tridiagonal since $\boldsymbol{R}^{\star}$ is a bidiagonal matrix.

Proof of Proposition 6.2

For $s = 1$, the proof is given in Proposition 6.1 for both $m = 1$ and $m > 1$.
For $s > 1$, let $\widetilde{\mathsf{A}} = \mathsf{Q}_s\,\mathsf{A} = (\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\mathsf{A}))$ and $\widetilde{\mathsf{b}} = \mathsf{Q}_s\,\mathsf{b} = (\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\mathsf{b}))$.
We use induction,

for $m = 1$,

$$\boldsymbol{w}_1^{\star s} = \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\mathsf{b})) = \mathsf{Q}^{(1)}\,\widetilde{\mathsf{b}}$$

for $m \geq 2$,

$$\begin{aligned}
\boldsymbol{w}_m^{\star s} &= \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_{m-1}'\boldsymbol{y})) \\
&= \mathsf{Q}^{(s)}\,\Big(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\left[\,(\boldsymbol{I} - \mathcal{P}_{T_{m-1}})\,\boldsymbol{X}_0\,\right]'\,\boldsymbol{y}\,)\Big) \\
&= \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\left[\boldsymbol{X}_0'\,(\boldsymbol{I} - \mathcal{P}_{T_{m-1}})'\right]\,\boldsymbol{y}\,)) \\
&= \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\boldsymbol{y}\,)) - \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\mathcal{P}_{T_{m-1}}'\,\boldsymbol{y}\,)) \\
&= \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\boldsymbol{y}\,)) - \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\mathcal{P}_{T_{m-1}}\,\boldsymbol{y}\,)) \\
&= \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\boldsymbol{y}\,)) - \\
&\quad - \mathsf{Q}^{(s)}\,\Big(\,\mathsf{Q}^{(s-1)} \ldots \Big(\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\left(\boldsymbol{T}_{m-1}\,(\boldsymbol{T}_{m-1}'\,\boldsymbol{T}_{m-1})^{-1}\,\boldsymbol{T}_{m-1}'\,\boldsymbol{y}\right)\Big)\Big) \\
&= \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\boldsymbol{y}\,)) - \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\boldsymbol{X}_0\,\widetilde{\boldsymbol{W}}_{m-1}\,\widehat{\boldsymbol{q}}_{m-1}\,)) \\
&= \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\boldsymbol{y}\,)) - \mathsf{Q}^{(s)}\,(\,\mathsf{Q}^{(s-1)} \ldots (\,\mathsf{Q}^{(1)}\,\boldsymbol{X}_0'\,\boldsymbol{X}_0\,\widehat{\boldsymbol{\beta}}_{m-1}\,)) \\
&= \widetilde{\mathsf{b}} - \widetilde{\mathsf{A}}\,\widehat{\boldsymbol{\beta}}_{m-1}. \qquad (7.13)
\end{aligned}$$

Proof of Lemma 6.2

We start the proof by reproducing a formal definition of equivalence transformations (see Lancaster and Miron, 1985, page 255) as follows: two matrix polynomials $B(\lambda)$ and $A(\lambda)$ are equivalent if the former can be obtained from $A(\lambda)$ by a finite sequence of elementary operations. That is if there are elementary matrices

$$E_1(\lambda), \ldots, E_k(\lambda), E_{k+1}(\lambda), \ldots, E_s(\lambda),$$

such that

$$B(\lambda) = E_1(\lambda) \cdots E_k(\lambda) \cdot A(\lambda) \cdot E_{k+1}(\lambda) \cdots E_s(\lambda).$$

Setting $Q(\lambda) = E_1(\lambda), \ldots, E_k(\lambda) = E_{k+1}(\lambda), \ldots, E_s(\lambda)$ we have that $B(\lambda)$ and $A(\lambda)$ are equivalent if there is unimodular matrix $Q(\lambda)$ such that

$$B(\lambda) = Q(\lambda)\, A(\lambda)\, Q(\lambda).$$

The sequence of elementary operations $E(\lambda)$ can be generally described by the elementary matrix below

$$\begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & c & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix},$$

with c in our case representing the relative importance factor q_j on iteration ℓ.

Coming back to our Proposition we give the proof for $\ell = 1$ and $\ell = 2$ which is then generalized to $3 \leq \ell \leq s$. For $\ell = 1$, the diagonal $p \times p$ precondition matrix Q is the identity matrix since for $\ell = 1$ the predictor variables have the same relative importance. The identity matrix is the simplest case of a unimodular matrix. For $\ell = 2$, the diagonal $p \times p$ precondition matrix equals to

$$\mathsf{Q}(\lambda) = E_1(\lambda), \ldots, E_j(\lambda), \ldots, E_p(\lambda),$$

with $E_j(\lambda)$ an elementary matrix as above with $c = \mathsf{q}_j$, that is with c the relative importance factor for variable j. By performing a number of elementary row operations $E_j(\lambda)$ we still have an elementary matrix Q with diagonal entries the relative importance factors for all the predictor variables. Then the matrix polynomial as well as the minimal polynomial (the latter divides any polynomial P of a matrix M for which $P(M) = 0$) of $\mathsf{Q}^\ell\ \mathsf{A}\ \mathsf{Q}^\ell$ for $\ell = 1, 2$ is identical and equivalent, respectively, to A. Consequently, the degree of $\mathsf{Q}^2\ \mathsf{A}\ \mathsf{Q}^2$ does not exceed the degree of $\mathsf{Q}^1\ \mathsf{A}\ \mathsf{Q}^1$. By induction and using the same steps we generalize for $\ell > 2$ and the proof is complete.

Proof of Proposition 6.4

We start by diagonalizing matrix A following the eigen decomposition, we have :

$$\begin{aligned} \mathsf{A}\,\mathsf{z} &= \mathsf{b} \\ \mathsf{W}_p\,\mathsf{diag}(\lambda_1,\ldots,\lambda_p)\,\mathsf{W}_p'\,\mathsf{z} &= \mathsf{b} \\ \mathsf{W}_p\,\mathsf{diag}(\lambda_1,\ldots,\lambda_p)\,\mathsf{W}_p'\,\mathsf{z} &= \mathsf{W}_p\,\mathsf{W}_p'\,\mathsf{b}. \end{aligned} \tag{7.14}$$

The PC regression vector solves the preconditioned system below for $\ell < p$,

$$\begin{aligned} \mathsf{W}_\ell\,\mathsf{diag}(\lambda_1,\ldots,\lambda_\ell)\,\mathsf{W}_\ell'\,\mathsf{z} &= \mathsf{W}_\ell\,\mathsf{W}_\ell'\,\mathsf{b} \\ \widetilde{\mathsf{A}}_\ell\,\mathsf{z} &= \widetilde{\mathsf{b}}_\ell \end{aligned} \tag{7.15}$$

for the preconditioned matrix and vector $\widetilde{\mathsf{A}}_\ell$ and $\widetilde{\mathsf{b}}_\ell$ depending on ℓ, that is the index of the subspace approximation, or the number of retained components.

Bibliography

Ashby, S., Manteuffel, T., and Saylor, P. (1990). A taxonomy for conjugate gradient methods. *SIAM Journal on Numerical Analysis*, 27(6):1542–1568.

Bastien, P., Vinzi, V., and Tenenhaus, M. (2005). PLS generalized linear regression. *Computational Statistics & Data Analysis*, 48:17–46.

Belsley, D. (1991). *Conditioning Diagnostics: Collinearity and Weak Data in Regression*. John Wiley & Sons, New York.

Belsley, D., Kuh, E., and Welsch, R. (1980). *Regression Diagnostics*. John Wiley & Sons, New York.

Billor, N., Hadi, A. S., and Velleman, P. F. (2000). BACON: Blocked adaptive computationally-efficient outlier nominators. *Computational Statist & Data Analysis*, 34:279–298.

Birkes, D. and Dodge, Y. (1993). *Alternative Methods of Regression*. John Wiley & Sons, New York.

Breiman, L. (1996). Heuristics of instability and stabilization in model selection. *The Annals of Statistics*, 24:2350–2383.

Breiman, L. and Friedman, J. (1997). Predicting multivariate responses in multiple linear regression. *Journal of the Royal Statistical Society, Series B*, 59:3–37.

Brown, P., Fearn, T., and Vannucci, M. (2001). Bayesian Wavelet Regression on Curves with Application to a Spectroscopic Calibration Problem. *Journal of the American Statistical Association*, 96:398–408.

Brown, P., Vannucci, M., and Fearn, T. (1998a). Multivariate bayesian variable selection and prediction. *Journal of the Royal Statistical Society, Series B*, 60(3):627–641.

Brown, P., Vannucci, M., and T., F. (1998b). Bayesian wavelength selection in multicomponent analysis. *Journal of Chemometrics*, 12(3):173–182.

Buja, A., Hastie, T., and Tibshirani, R. (1989). Linear smoothers and additive models. *The Annals of Statistics*, 17:453–510.

Butler, N. and Denham, M. (2000). The peculiar shrinkage properties of partial least squares regression. *Journal of the Royal Statistical Society, Series B*, 62:585–594.

Cade, B. and Richards, D. (1996). Permutation tests for least absolute deviation regression. *Biometrics*, 52:886–902.

Calvetti, D., Reichel, L., and Shuibi, A. (2005). Invertible smoothing preconditioners for linear discrete ill-posed problems. *Appl. Numer. Math.*, 54:135–149.

Calvetti, D. and Somersalo, E. (2005). Priorconditioners for linear systems. *Inverse Problems*, 21:1397–1418.

Carreira, M. (1997). A review of dimension reduction techniques. Technical Report CS–96–09, Dept. of Computer Science, University of Sheffield.

Chatterjee, S. and Hadi, A. (1988). *Sensitivity Analysis in Linear Regression*. John Wiley, New York.

Chui, C. (1992). *An Introduction to wavelets*. Academic Press, New York.

Cook, R. (1977). Detection of influential observations in linear regression. *Technometrics*, 19:15–18.

de Jong, S. (1993). SIMPLS: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 18:251–263.

de Jong, S. (1995). PLS shrinks. *Journal of Chemometrics*, 9:323–326.

De Sturler, E. (1999). The idea behind Krylov Methods. *SIAM Journal on Numerical Analysis*, 36(3):864–889.

Denham, M. (2000). Choosing the number of factors in partial least squares regression: estimating and minimizing the mean squared error of prediction. *Journal of Chemometrics*, 14:351–361.

Devlin, S., Gnanadesikan, R., and Kettenring, J. R. (1981). Robust estimation of dispersion matrices and principal components. *Journal of the American Statistical Association*, 76:354–362.

Dodge, Y. (1987). An introduction to L1-norm based on statistical data analysis. *Computational Statistics & Data Analysis*, 5:329–253.

Dodge, Y. (1992). *L1 – Statistical data analysis and related methods*. North-Holland, Amsterdam.

Dodge, Y. (1997). LAD regression for detection of outliers in response and explanatory variables. *Journal of Multivariate Analysis*, 61:144–158.

Dodge, Y. and Jureckova, J. (1987). Adaptive combination of least squares and least absolute deviations estimators. In Dodge, Y., editor, *Statistical Data Analysis based on L1-norm and Related Methods*. Elsevier Science.

Dodge, Y., Kondylis, A., and Whittaker, J. (2004). Extending PLS1 to PLAD regression and the use of the L1 norm in soft modelling. In Antoch, J., editor, *COMPSTAT 2004*, pages 935–942. Physica/Verlag - Springer.

Draper, N. and Smith, H. (1998). *Applied Regression Analysis*. John Wiley & Sons, New York.

Efron, B. (1987). Better bootstrap confidence intervals. *Journal of the American Statistical Association*, 92:171–185.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *Annals of Statistics*, 32:407–451.

Efron, B. and Tibshirani, R. (1993). *An Introduction to the Bootstrap*. Chapman and Hall, New York.

Efron, B. and Tibshirani, R. (1997). Improvements on cross-validation: The .632+ bootstrap method. *Journal of the American Statistical Association*, 92:548–560.

Engelen, S., Hubert, M., Branden, V. K., and Verboven, S. (2004). Robust PCR and robust PLSR: a comparative study. In Hubert, M. Pison, G. S. A. and Van Aelst, S., editors, *Theory and Applications of Recent Robust Methods, Series: Statistics for Industry and Technology*, pages 105–117. Birkhauser, Basel.

Eriksson, L., Johansson, E., Lindgren, F., and Wold, S. (2000). GIFI-PLS: Modeling of Non-Linearities and Discontinuities in QSAR. *QSAR & Combinatorial Science*, 19:345–355.

Fahrmeir, L. and Tutz, G. (2001). *Multivariate Statistical Modelling Based on Generalized Linear Models*. Springer Series in Statistics, Berlin.

Fearn, T. (1983). A missue of ridge regression in the calibration of a near infrared reflectance instrument. *Applied Statistics*, 32:73–79.

Frank, I. and Friedman, J. (1993). A statistical view of some chemometrics regression tools. *Technometrics*, 35:109–135.

Friedman, H. and Stuetzle, W. (1981). Projection pursuit regression. *Journal of American Statistical Association*, 76:735–743.

Geladi, P. and Kowalski, B. R. (1986). Partial least-squares regression: A tutorial. *Analytica Chimica Acta*, 185:1–17.

George, E. I. and McCulloch, R. E. (1993). Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, 88(423):881–889.

George, E. I. and McCulloch, R. E. (1997). Approaches for Bayesian variable selection. *Statistica Sinica*, 7:339–373.

Gnanadesikan, R. and Kettenring, J. R. (1972). Robust estimates, residuals, and outlier detection with multiresponse data. *Biometrics*, 28:81–124.

Golub, G. and Kahan, W. (1965). Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*, 2(2):205–224.

Golub, G. and Van Loan, C. (1996). *Matrix Computation*. Johns Hopkins University Press, Baltimore.

Goutis, C. (1996). Partial least squares algorithm yields shrinkage estimators. *The Annals of Statistics*, 24:816–824.

Gunst, R. F. and Mason, R. L. (1977). Biased estimation in regression: An evaluation using mean squared error. *Journal of the American Statistical Association*, 72:616–628.

Hadi, A. (1992a). A new measure of potential influence. *Computational Statistics & Data Analysis*, 14:1–27.

Hadi, A. S. (1992b). Identifying multiple outliers in multivariate data. *Journal of the Royal Statistical Society. Series B (Methodological)*, 54:761–771.

Hadi, A. S. (1994). A modification of a method for the detection of outliers in multivariate samples. *JRSS, Series B*, 56:393–396.

Hampel, F., Ronchetti, E., Rousseuw, P., and Stahel, W. (1986). *Robust Statistics*. John Wiley, New York.

Hastie, T., Tibshirani, R., and Friedman, J. (2004). *Elements of Statistical Learning*. Springer, New York.

Helland, I. (1988). On the structure of partial least squares regression. *Communications in Statistics - Simulation and Computation*, 17:581–607.

Helland, I. S. (1990). Partial Least Squares Regression and Statistical Models. *Scand. Journal of Statistics*, 17:97–114.

Helland, I. S. (2001). Some theoretical aspects of partial least squares regression. *Special Issue on PLS regression in Chemometrics and Intelligent Laboratory Systems*, 58:97–107.

Helland, I. S. (2004). Statistical inference under symmetry. *Internat. Statist. Rev.*, 72:409–422.

Helland, I. S. and Almøy, T. (1994). Comparison of prediction methods when only a few components are relevant. *Journal of American Statistical Association*, 89:583–591.

Hestens, M. and Stiefel, E. (1952). Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards, Section B*, 49(6):409–436.

Höskuldsson, A. (1988). PLS regression methods. *Journal of Chemometrics*, 2:211–228.

Hössjer, O. and Croux, C. (1995). Generalizing univariate signed rank statistics for testing and estimating a multivariate location parameter. *J. Nonparam. Statist.*, 4:293–308.

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35:73–101.

Huber, P. J. (1981). *Robust Statistics*. John Wiley & Sons, New York.

Huber, P. J. (1987). The place of the L_1 norm in robust estimation. In Dodge, Y., editor, *Statistical Data Analysis based on L1-norm and Related Methods*, pages 23–33. Elsevier Science.

Hubert, M. and Branden, K. (2003). Robust methods for Partial Least Squares Regression. *Journal of Chemometrics*, 17:537–549.

Hubert, M., Rousseeuw, P., and Vanden Branden, K. (2005). ROBPCA: a new approach to robust principal component analysis. *Technometrics*, 47:64–79.

Ilic, M. and Turner, I. (2005). Krylov Subspaces and the Analytic Grade. *Numerical Linear Algebra and its Applications*, 12:55–76.

Ipsen, I. and Meyer, C. (1998). The idea behind Krylov Methods. *The American Mathematical Monthly*, 105(10):889–899.

Kalivas, J. (1999). Interrelationships of multivariate regression methods using eigenvector basis sets. *Journal of Chemometrics*, 13:111–132.

Koenker, R. (2005). *Quantile Regression*. Cambridge University Press, Cambridge.

Koenker, R. and Bassett, G. (1978). Regression quantiles. *Econometrica*, 46:33–50.

Kondylis, A. and Hadi, A. S. (2006a). The BACON approach for rank deficient data. *working paper*.

Kondylis, A. and Hadi, A. S. (2006b). Derived Components Regression using the BACON algorithm. *In Press in Computational Statistics & Data Analysis*.

Kondylis, A. and Whittaker, J. (2006). Bootstrapping Partial LAD regression. *to appear in the Special Issue on PLS in Computational Statistics*.

Krämer, N. (2006). An Overview on the Shrinkage Properties of Partial Least Squares Regression. *to appear in the Special Issue on PLS in Computational Statistics*.

Krämer, N., Boulesteix, A., and Tutz, G. (2006). Penalizing PLS based on B-Splines. *in preparation*.

Lancaster, M. and Miron, T. (1985). *The Theory of Matrices*. Academic Press INC., Florida.

Lanczos, C. (1950). An iteration method for the solution on the eigenvalue problem of linear differential and integral operators. *Journal of Research of the National Bureau of Standards*, 45(4):252–282. Research paper 2133.

Lang, P. M., Brenchley, J. M., Nieves, R. G., and Kalivas, J. H. (1998). Cyclic subspace regression. *Journal of Multivariate Analysis*, 65:58–70.

Leroy, A. and Rousseeuw, P. (1987). *Robust Regression & Outlier Detection*. John Wiley & Sons, New York.

Locantore, N., Marron, J., Simpson, D., Tripoli, N., Zhang, J., and Cohen, K. (1999). Robust principal components analysis for functional data. *Test*, 8:1–73.

Manne, R. (1987). Analysis of two partial-least-squares algorithms for multivariate calibration. *Chemometrics and Intelligent Laboratory Systems*, 2:187–197.

Martens, H. (1985). *Multivariate Calibration*. PhD thesis, Technical University of Norway, Trondheim, Norway.

Martens, H. and Naes, T. (1989). *Multivariate Calibration*. John Wiley & Sons, New York.

McIntosh, A. R., Bookstein, F. L., Haxby, J. V., and Grady, C. L. (1996). Spatial pattern analysis in of functional brain imagaes using partial least squares. *Neuroimage*, 3:143–157.

Mitchell, T. and Beauchamp, J. (1988). Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032.

Naes, T. (1985). Multivariate calibration when the error covariance matrix is structured. *Technometrics*, 27:301–311.

Neumaier, A. (1998). Solving ill-conditioned and singular linear systems: A tutorial on regularization. *SIAM Review*, 40:636–666.

Nguyen, D. V. and Rocke, D. M. (2002a). Partial least squares proportional hazard regression for application to DNA microarray survival data. *Bioinformatics*, 18(12):1625–1632.

Nguyen, D. V. and Rocke, D. M. (2002b). Tumor classification by partial least squares using gene expression data. *Bioinformatics*, 18(1):39–50.

Parlett, B. (1980). *The Symmetric Eigenvalue Problem*. Prentice-Hall Series in Applied Mathematics, NJ.

Phatak, A. (1993). *Evaluation of some Multivariate Methods and their Applications in Chemical Engineering*. PhD thesis, University of Waterloo, Ontario, Canada.

Phatak, A. and de Hoog, F. (2001). PLSR, Lanczos, and Conjugate Gradients. Technical Report CMIS 01/122, Commonwealth Scientific and Industrial Research Organisation.

Ramsay, J. and Silverman, B. (2005). *Functional Data Analysis*. Springer, Springer-Verlag, New York.

Rosipal, R. and Trejo, L. (2001). Kernel PLS in Reproducing Kernel Hilbert Spaces. *Journal of Machine Learning Research*, 2:97–123.

Ruhe, A. (1998). Rational Krylov: A practical algorithm for large sparse nonsymmetric matrix pencils. *SIAM J. Sci. Comput.*, 19(5):1535–1551.

Sardy, S. (1998). *Regularization Techniques for Linear Regression with a Large Set of Carriers*. PhD thesis, University of Washington.

Seber G.A.F., Lee, A. (2003). *Linear Regression Analysis*. John Wiley and Sons, N.Y.

Selwood, D., Livingstone, D., Comley, J., O'Dowd, A., Hudson, A., Jackson, P., Jandu, K., Rose, V., and Stables, J. (1990). Structure-activity relationships of antifilarial antimycin analogues, a multivariate pattern recognition study. *J.Med.Chem.*, 33:136–142.

Serneels, S., Croux, C., Filzmoser, P., and Van Espen, P. J. (2005a). Partial Robust M-regression. *Chemometrics and Intelligent Laboratory Systems*, 79(1–2):55–64.

Serneels, S., Filzmoser, P., Croux, C., and Van Espen, P. (2005b). Robust continuum regression. *Chemometrics and Intelligent Laboratory Systems*, 76:197–204.

Stone, M. and Brooks, R. J. (1990). Continuum regression: Cross-validated sequentially constructed prediction embracing ordinary least squares, partial least squares and principal components regression. *Journal of the Royal Statistical Society. Series B*, 52:237–269.

Tenenhaus, M. (1998). *La régression PLS. Théorie et pratique.* Technip, Paris.

Tikhonov, A. (1963). Solution of incorrectly formulated problems and the regularization method. *Soviet Math Dokl*, 4:1035–1038.

Visuri, S., Koivunen, V., and Oja, H. (2000). Sign and rank covariance matrices. *Journal of Statistical Planning and Inference*, 91:557–575.

Wahba, G. (1990). *Spline Models for Observational Data.* SIAM, Philadelphia.

Walker, E. and Birch, J. (1988). Influence measures in ridge regression. *Technometrics*, 30(2):221–227.

Wang, H., Li, G., and Jiang, G. (2006). Robust Regression Shrinkage and Consistent Variable Selection through the LAD-Lasso. *Journal of Business & Economics Statistics, to appear.*

Wen, W. and Manne, R. (2000). Fast regression methods in a Lanczos (or PLS-1) basis. Theory and applications. *Chemometrics and Intelligent Laboratory Systems*, 51:145–161.

West, M. (2003). Bayesian Factor Regression Models in the 'Large p, Small n' Paradigm. In Bernardo, J. M., Bayarri, J. M., Berger, J. O., Dawid, A. P., Heckerman, D., Smith, A. F. M., and West, M., editors, *BAYESIAN STATISTICS.* Oxford University Press.

Wold, H. (1966). Estimation of principal components and related models by iterative least squares. In Krishnaiah, P. R., editor, *Multivariate Analysis*, pages 391–420, New York. Academic Press.

Wold, H. (1975). Soft modelling by latent variables: the nonlinear iterative partial least squares approach. In Gani, J., editor, *Perspectives in Probability and Statistics, Papers in Honour of M.S.Bartlett*, pages 520–540, London. Academic Press.

Wold, S., Martens, H., and Wold, H. (1983). The multivariate calibration problem in chemistry solved by the PLS method. *Lecture notes in mathematics. Proc. Conf. Matrix Pencils*, In A. Ruhe, and B. Kgstrm (ed.):286–293.

Wold, S., Ruhe, A., Wold, H., and Dunn, W. J. (1984). The collinearity problem in linear regression. The partial least squares (PLS) approach to generalized inverses. *SIAM Journal of Scientic and Statistical Computing*, 5(3):735–743.

Wold, S., Sjostrom, M., and Eriksson, L. (2001). PLS-regression: a basic tool of chemometrics. *Chemometrics and Intelligent Laboratory Systems*, 58:109–130.

List of Algorithms

List of Figures

List of Tables