Diplomarbeit

zum Thema

Partial Least Squares (PLS) Regression and its Robustification

ausgeführt am
Institut für Statistik und Wahrscheinlichkeitstheorie
der Technischen Universität Wien

unter der Anleitung von
A.o.Univ.Prof. Dipl.-Ing. Dr.techn. Peter Filzmoser

durch

Barbara Kavšek
Matr.Nr.: 9426082
1100 Wien, Ranzonigasse 1

Wien, am 23. Mai 2002 Barbara Kavšek

Preface & Acknowledgements

Although I am glad that this work is completed now, I must admit that it is little more than a scratch on the surface on the topic of Partial Least Squares.

During my study of the PLS literature I realized how many different approaches and interpretations there are, and some seem to be only useful in the context of this specific paper. But despite all these confusions I got an idea of the power of this method. Still I think more work should be done concerning PLS, its theoretical background, its implementation and especially its robustification.

I hope I was able to write my thoughts onto paper such that others understand them, too. But I would be glad if possible questions would reach me as every question is a source of answers.

I want to thank Peter Filzmoser, my supervisor, for all the support he gave me – and that was a lot! I also want to emphasize his endurance as I was sticking to what I was taught from my father: "There are no stupid questions, just stupid answers!"

Thanks also to Rudi Dutter for the long discussions about the meaning of statistics, data engineering and about the usage of well-known phrases without thinking of their real meaning.
I found many such examples in literature and I really tried to avoid them in my work – forgive me if I overlooked some!

I also want to thank all the members of the Institute of Statistics and Probability Theory, Wolfgang Werther, who introduced PLS to me, and especially Heidi Filzmoser, who gave me the impression that statistics can be some kind of family-run business! (I found some proofs in the literature, too, but I will omit citations.)

Finally I want to thank Bernhard, his and my parents for all the support they gave me.
I don't know what I would have done without you!

Barbara Kavšek

e-mail: kavsek@pc2.statistik.tuwien.ac.at

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Partial Least Squares (PLS) Regression belongs to the family of Nonlinear Iterative Least Squares (NILES) procedures which were developed by H. Wold and other researchers at the University Institute of Statistics, Uppsala, Sweden. The iterative estimation methods presented in Wold (1966) cover the estimation of nonlinear models such as Principal Components Analysis, Canonical Correlation Analysis, Multiple Regression and Factor Analysis. In this work we will focus on the regression model, estimated with partial least squares regression, and–for explanatory reasons–the model for principal component analysis. The iterative estimation method for principal component analysis is called NIPALS (Nonlinear estimation by Iterative Partial Least Squares). As we will see later PLS (in this work we always mean PLS regression) can be interpreted as a double application of NIPALS.

PLS has been used in social sciences for quite a long time and later found its way to analytical chemistry and the field of mass spectrometry. The main reason for the application of PLS in mass spectrometry is its ability to handle data sets with more variables than observations.

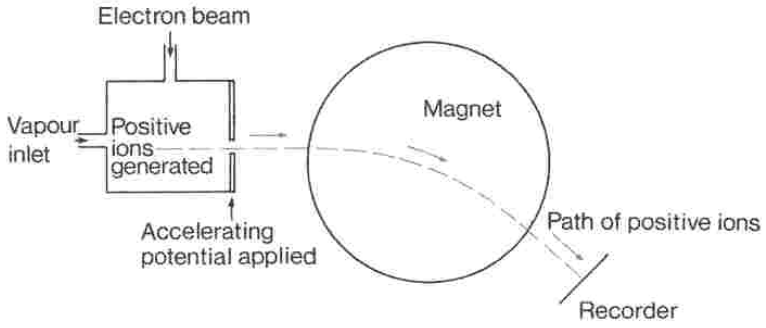


Figure 1.1: Mass spectrometer

The analysation of mass spectra (a sketch of a mass spectrometer is displayed in Figure 1.1) is a very complicated field of chemistry and requires much knowledge about the elements or compounds of elements and their respective spectra. Just to give an idea of the complexity of the subject, an element can occur in nature with different masses, e.g., oxygene of mass 16 or 17. According to the mass the so-called *peak* in the spectrum changes. Simply spoken

the mass spectrometer creates positive ions from the vapourized substance and records their arrival after passing a magnet. But when generating the positive ions the substance may also be split into parts which are recorded, too. Additionally elements occuring in the atmosphere will also be found in the recorded data. As an example the mass spectrum of ethanol is displayed in Figure 1.2.

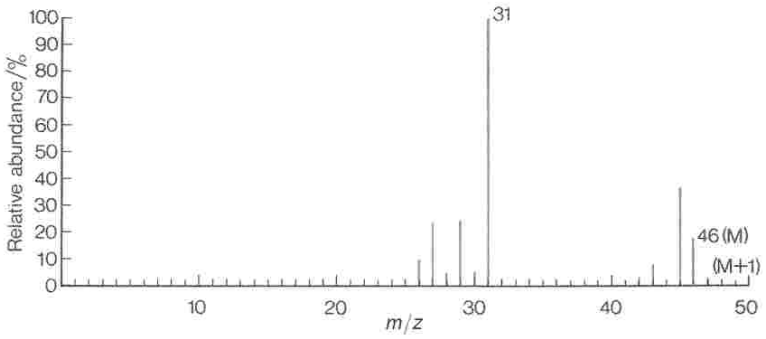


Figure 1.2: Mass spectrum of ethanol

The peak of the ethanol ion ($[CH_3CH_2OH]^+$) is at m/z 46, the largest peak at m/z 31 results from the ion $[CH_2OH]^+$. The peak at m/z 28 stands for N_2, the one at m/z for O_2. There are other peaks, too, created by different ionized parts of ethanol. A good introduction to mass spectrometry can be found in Duckett and Gilbert (2000).

This short and simplified description of mass spectrometry should give an idea of the amount of data created by analysation of various substances. PLS now comes into play when mass spectra of (known) substances are collected and another (unknown) substance has to be deduced (predicted) from this data base.

This work will now focus on the NIPALS algorithm for principal component analysis, then go on to the PLS algorithm itself, its mathematical properties and background. Examples, both for NIPALS and PLS, will be presented and discussed. Furthermore newer developments for PLS, based on the theory of Robust Statistics, are collected. The different ideas for robustifying the PLS algorithm are introduced and discussed. These ideas also lead to a short outlook what may be improved for getting good and applicable robust PLS algorithms.

Chapter 2

NIPALS–Nonlinear Estimation by Iterative Partial Least Squares

The NIPALS algorithm was developed by H. Wold in 1966 (see Wold, 1966) together with other methods that are related to what is called *"alternating"* regression. (It also received the nickname *"criss-cross"* regression.) These methods descend from the idea that by transposition of a regression equation the data matrix and the matrix of the regression coefficients change places and can therefore be reinterpreted.

NIPALS is the alternating regression alternative to classical principal component analysis (PCA) algorithms which rely on singular value decomposition. For further information on the classical methods confer Jackson (1991) or Mardia et al. (1979). The PLS (Partial Least Squares) method, which is the method of interest in this work, can be interpreted as a development on the basis of the NIPALS idea. As we will see in Chapter 3 it is–simply spoken–a double application of NIPALS. Therefore, as far as we are concerned, the study of the NIPALS algorithm is of extreme importance for the understanding of PLS.

In the book of Tenenhaus (1998a) PLS and related methods, including NIPALS, are presented. Tenenhaus (1998a) also introduces a version of NIPALS which is suitable for data matrices with missing values. Section 2.1.1 will go into the details of the changes which are to be performed onto the classical NIPALS procedure.

2.1 The NIPALS Algorithm

As NIPALS was developed as a substitute for principal component analysis we will first shortly present the main idea of this essential statistical method.
The aim of principal component analysis is that starting with a centered data matrix $\mathbf{X}$, $n \times p$, one wants to deduce k (preferably $k < p$) components $\mathbf{t}_h, h = 1, \ldots, k$ which explain as much of the variation of $\mathbf{X}$ as possible with minimal loss of information. In the extreme case of $k = p$ we would have no loss of information at all but we also would have no reduction of components explaining $\mathbf{X}$ as we would exactly deduce as much components as given

variables. Corresponding to these scores loadings $\mathbf{p}_h, h = 1, \ldots, k$ are deduced which explain the relation between the original variables and the newly gained components.

The model for PCA is

$$\mathbf{X} = \mathbf{T}\mathbf{P}^\top + \boldsymbol{\varepsilon}$$

where the matrix $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_k]$ of principal components has dimension $n \times k$ and $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_k]$ has size $p \times k$, where k denotes the number of components. $\boldsymbol{\varepsilon}$ is the matrix of errors which are made by extracting less components than the original number of variables (in other words: the error made by reduction of dimensionality). In the extreme case of $k = p$ components this error term would be equal $\mathbf{0}$. The components (or scores) $\mathbf{t}_1, \ldots, \mathbf{t}_k$ and the loadings $\mathbf{p}_1, \ldots, \mathbf{p}_k$ are calculated by means of spectral decomposition or singular value decomposition. Cf. Jackson (1991) or Mardia et al. (1979) for further information on principal component analysis.

An important question is how many principal components should be selected in order to get a good representation of the original data matrix $\mathbf{X}$. In the classical algorithm for principal component analysis this number is not chosen in advance. On the contrary, first as many components as there are variables are calculated and then the importance of each one is determined. This is usually done by statistical tests, rules of thumb or graphically with the help of the so-called *screeplot*. The underlying calculations for all these methods depend on the variance of the principal components or their proportion on total variation, respectively.

NIPALS takes a different way to calculate principal components. Instead of using singular value decomposition as it is done in classical principal component analysis, NIPALS only uses simple linear regressions.

But still there are great differences to the principle of regression analysis: Usually we have a model of the form

$$\mathbf{Y} = \mathbf{X}\mathbf{K}^\top + \boldsymbol{\varepsilon}$$

where both $\mathbf{X}$ and $\mathbf{Y}$ are given and the regression coefficients $\mathbf{K}$ are fitted to these values. The main goal of this linear regression analysis is to find a linear relation between these two data matrices, often in order to predict unknown $\mathbf{Y}$–values from known $\mathbf{X}$–values.

In the case of principal component analysis only $\mathbf{X}$ is known (which would take the $\mathbf{Y}$–part in the above regression model). So the idea of NIPALS is to take a starting value for $\mathbf{T}$ and fit $\mathbf{P}$ to $\mathbf{X}$ and this starting value. This means that $\mathbf{P}$ plays the role of our regression coefficients $\mathbf{K}$. In the next step $\mathbf{T}$ and $\mathbf{P}$ change places (by transposition of the equation), so $\mathbf{T}$ will stand for the regression coefficients $\mathbf{K}$ and is fitted to $\mathbf{X}^\top$ and $\mathbf{P}$. These two steps form the kernel of the NIPALS algorithm and are performed until *"convergence"* in the sense that a certain accuracy of $\mathbf{T}$ is met. Additionally between the two steps the newly gained values for $\mathbf{P}$ are normalized as it is a common demand in principal component analysis to have normalized loadings vectors $\mathbf{p}_1, \ldots, \mathbf{p}_k$. The construction of NIPALS as given below also guarantees the orthogonality of the components as is the case in classical principal component analysis. This follows from the calculation of the residuals and the properties of the least squares (LS) estimator. One of the problems in the context of robustification is

that this orthogonality of the residuals is lost by the usage of robust methods substituting LS.

Because of these alternating definitions of which part of the equation stands for the regression coefficients, Wold (1966) called this principle *"alternating"* or *"criss-cross"* regression. According to Tenenhaus (1998a) the word *"partial"* comes from the fact that for the calculation of a new component in the regression only a part of the parameters (namely, the already deduced ones) is used.

The word *"convergence"* is slightly misleading in this context but it is the standard word used in literature on NIPALS or PLS. Under *"convergence"* we understand the following definition: After each iteration we calculate the difference between the former and the newly calculated component. If the norm of this difference is smaller than a given tolerance value we speak of *convergence of the component*.

The detailed algorithm can now be described as follows:

Let $\mathbf{X}_0 = \mathbf{X}$. For every component $\mathbf{t}_h, h = 1, \ldots, k$ compute the following steps, where Steps 1 to 4 are the steps to be iterated:

Step 0: Choose a column of $\mathbf{X}_{h-1}$, e.g. the first, as a starting value for $\mathbf{t}_h$.
Step 1: Perform the following regression:

$$\mathbf{X}_{h-1} = \mathbf{t}_h \mathbf{p}_h^\top + \boldsymbol{\varepsilon}_1$$

yielding the least squares solution

$$\mathbf{p}_h^\top = \frac{\mathbf{t}_h^\top \mathbf{X}_{h-1}}{\mathbf{t}_h^\top \mathbf{t}_h}.$$

Step 2: Normalize $\mathbf{p}_h^\top$:

$$\mathbf{p}_h^\top := \frac{\mathbf{p}_h^\top}{\|\mathbf{p}_h\|}.$$

Step 3: Perform the regression:

$$\mathbf{X}_{h-1} = \mathbf{t}_h \mathbf{p}_h^\top + \boldsymbol{\varepsilon}_2$$

which is after transposition

$$\mathbf{X}_{h-1}^\top = \mathbf{p}_h \mathbf{t}_h^\top + \boldsymbol{\varepsilon}_2^\top$$

giving the least squares solution

$$\mathbf{t}_h^\top = \frac{\mathbf{p}_h^\top \mathbf{X}_{h-1}^\top}{\mathbf{p}_h^\top \mathbf{p}_h}.$$

Step 4: Check convergence of $\mathbf{t}_h$:
With every iteration the accuracy of $\mathbf{t}_h$ is improved (or stays at least equal) so that a possible convergence check is to look wether the norm of the difference between $\mathbf{t}_h$ of the current and $\mathbf{t}_h$ of the previous iteration is smaller than a given value or not. If the current $\mathbf{t}_h$ is better than the previous $\mathbf{t}_h$ in the above sense then go to Step 1. Otherwise the component $\mathbf{t}_h$ with the

current value is already best and we keep $\mathbf{t}_h$ as the final value (or the best approximation). In this case we compute the next component $\mathbf{t}_{h+1}$ starting with Step 0 after calculating the residuals $\mathbf{X}_h = \mathbf{X}_{h-1} - \mathbf{t}_h \mathbf{p}_h^\top$.

This algorithm now gives as a result the scores $\mathbf{t}_h$ with the loadings $\mathbf{p}_h$, $h = 1, \ldots, k$, which are at least approximately equal to the scores and loadings obtained by PCA.
A little study of the quality of the NIPALS components on the basis of some practical examples is done in Section 2.2.

If we take another look at the steps of the NIPALS algorithm we will see that the least squares solution in Step 3

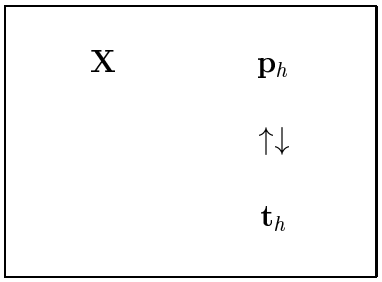
$$\mathbf{t}_h^\top = \frac{\mathbf{p}_h^\top \mathbf{X}_{h-1}^\top}{\mathbf{p}_h^\top \mathbf{p}_h}$$

can be simplified by

$$\mathbf{t}_h^\top = \mathbf{p}_h^\top \mathbf{X}_{h-1}^\top$$

as the denominator is equal 1 by the normalization of $\mathbf{p}_h$ performed in Step 2. We kept the above notation for better understanding as it is clearer that we really apply the least squares method. By performing a robustification of the algorithm, e.g., a replacement of the least squares solution by a robust regression solution, the obtained solution would naturally be different.

A graphical representation of the operation of the NIPALS algorithm, which in this context is quite clear but gains importance in the next chapter about PLS, could look like that:



2.1.1 NIPALS for Missing Values

Tenenhaus (1998a) presents a modification of the NIPALS algorithm in order to be able to handle with missing values. As this problem occurs very often in practice the importance of such methods is out of discussion.

The underlying idea in Tenenhaus (1998a) is quite simple: Only the existing values are taken into account.

Compared to the above defintion of the NIPALS algorithm the following changes have to be made:

For reasons of simplicity let $\mathbf{X}_{h-1}$ be denoted by $\tilde{\mathbf{X}}$.

Replace Step 1 with the following

Step 1^***:** For $j = 1, 2, \ldots, p$ calculate:

$$p_{jh} = \frac{\sum_{i=1, \text{ where } \tilde{x}_{ij} \text{ and } t_{ih} \text{ exist}}^{n} \tilde{x}_{ij} t_{ih}}{\sum_{i=1, \text{ where } \tilde{x}_{ij} \text{ and } t_{ih} \text{ exist}}^{n} t_{ih}^2}.$$

Consequently replace Step 3 with

Step 3^***:** For $i = 1, 2, \ldots, n$ calculate:

$$t_{ih} = \frac{\sum_{j=1, \text{ where } \tilde{x}_{ij} \text{ exists}}^{p} \tilde{x}_{ij} p_{jh}}{\sum_{j=1, \text{ where } \tilde{x}_{ij} \text{ exists}}^{p} p_{jh}^2}.$$

By performing this modified algorithm the scores and loadings which correspond to the remainder of the data set are returned.

2.2 Examples

As NIPALS is an alternative algorithm for principal component analysis (PCA) we will compare its results to those of other conventional algorithms for PCA. In Appendix A a detailed description of the data sets used is given. All data sets can be obtained from the author.

Throughout this work the statistical software R was used. Besides being freeware it is fairly easy to extend with self-written functions or programs. Extensions for this software are produced and made available from scientists all over the world. R can be downloaded from the webpage *http://cran.r-project.org*.

2.2.1 The districts Data Set

The data set concerning the Austrian political districts and their economic behaviour is very suitable for principal component analysis. Trends like the rural exodus or educational differences as well as the proportion of young and old are important catchwords which may be found via principal components.

First we will calculate the principal components with the NIPALS algorithm introduced in Section 2.1 and then we will compare its results to those performed by the function *princomp* of R. This function is an implementation of the classical principal component analysis. Contrary to the functionality of NIPALS the number of components to be extracted by *princomp* can not be chosen in advance. The maximal number of components, namely as many as variables, are calculated. The number of components to be used for explaining the data set is decided afterwards by graphical representation of their importance. This importance is decreasing with increasing index. A possibility for graphical representation of the importance of the components is the *screeplot*. For reasons of comparability of this importance also as many components as are returned by *princomp* have to be calculated with NIPALS.

The districts data matrix consists of 17 variables and 99 observations but still we want to deduce fewer factors describing the data set. Note that if we calculated a certain number of components with NIPALS and then performed NIPALS with a higher number of components the first components would stay the same by construction of the NIPALS algorithm. As the components are becoming less important with increasing index a higher number of extracted components is not necessarily better.

The implemented version of the NIPALS algorithm is printed in Appendix B.

As input we used the mean-centered and scaled data matrix of the districts, the maximal number of iterations to be performed was set to 10 and for the tolerance level of the precision of the components 10^{-4} was selected. Practice has shown that in general 10 iterations are sufficient to get quite a good result. Still, with a higher number of iterations and a lower tolerance level more exact components would be gained but the difference will normally only show up in the last decimals which has no impact on practical surveys.

To present the results gained with either of the algorithms we chose the well-known *biplot* (cf. Gower and Hand, 1996). In this graphic the first component is printed against the second component both with respect to their loadings. The position of each variable is drawn by a vector (corresponding to the loadings).

The biplot of the districts data set calculated by NIPALS is displayed in Figure 2.1. The first and second component returned by NIPALS explain 32% of the total variation of this data set.

The first component makes a clear distiniction concerning education. In the negative part the major cities (E, W, G, I, S, K, L, P) and their surroundings (MD, WN, KS) are grouped with a high proportion of persons having secondary school and university education. These areas show high immigration and most of the employees can be found in services and trade. The positive part of the first component describes regions with a high percentage of mountain farms, education up to the level of primary school and a high percentage of commuters to other districts and of not-daily commuting people. These regions (e.g. HB, WZ, TA, RO, FR) depend mostly on agriculture and industry where agriculture is poorly developed.

The second component takes the age of the population into account, combined with working places in tourism and agriculture. Naturally the number of tourist stays is combined with the proportion of employees in tourism and the districts at this side of the component are well-known touristic regions such as BZ, IL, KB, SL, KU and RE. The positve part of the second component consists of a higher percentage of adults, youth and children and of tourism and touristic stays. Opposed to them are a higher number of old people and agriculture. This can be explained with the rural exodus. These regions are HL, HO, OP, WT, WY, MI, MZ.

Compared to these results the function *princomp* yields principal components that are only slightly different (cf. Figure 2.2). The interpretation therefore remains the same. The first and second component calculated with NIPALS explain–similarly to those of *princomp*–32% of the total variation.

Still the number of important components is an interesting question. Generally in classical principal component analysis as many components as variables are computed. The screeplot

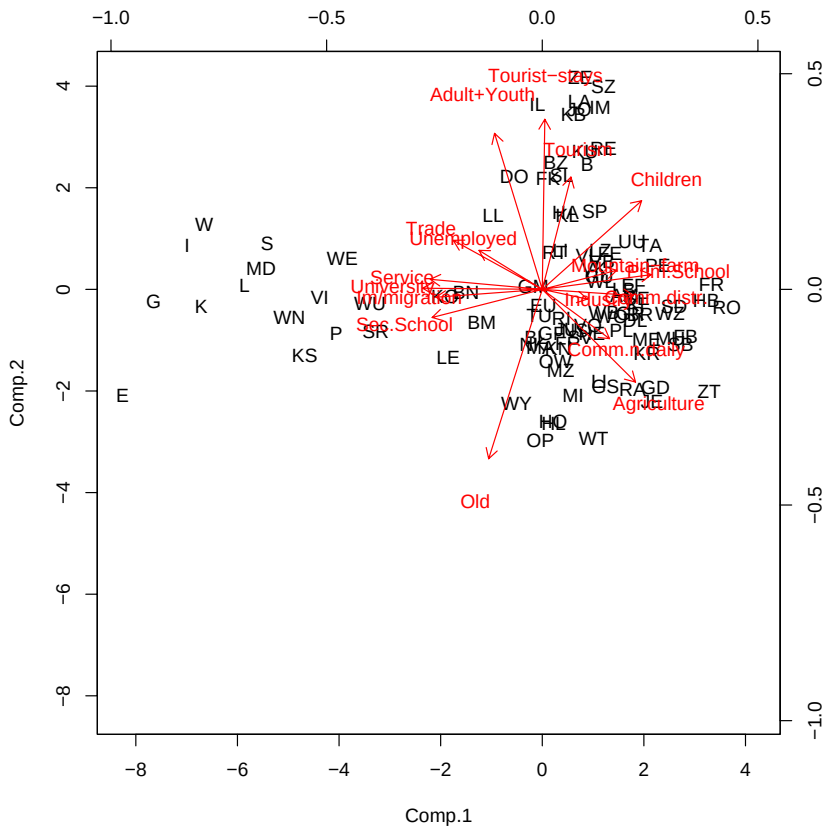


Figure 2.1: Biplot of NIPALS for districts

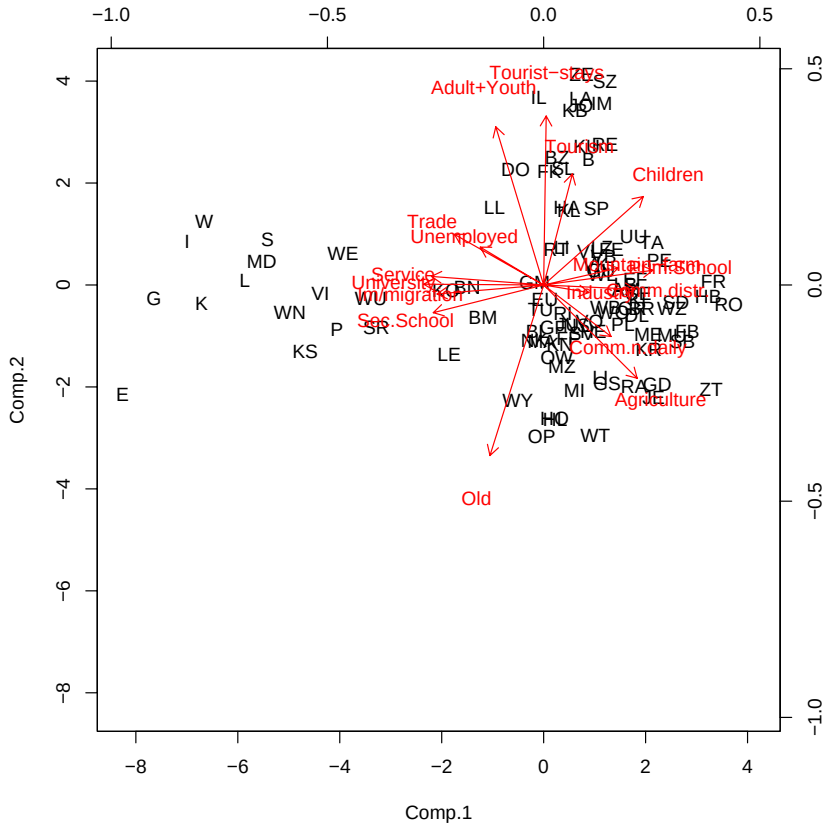


Figure 2.2: Biplot of *princomp* for districts

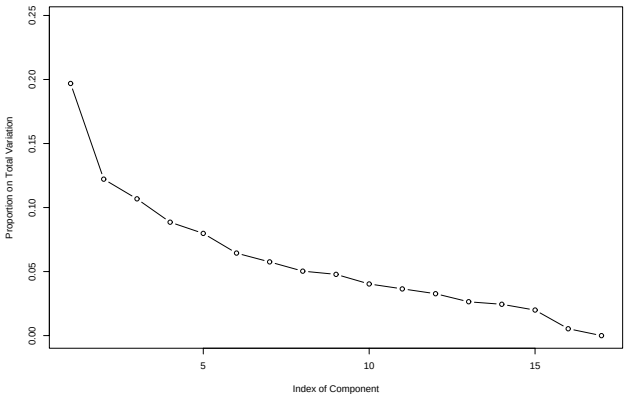


Figure 2.3: Screeplot of *princomp* for districts

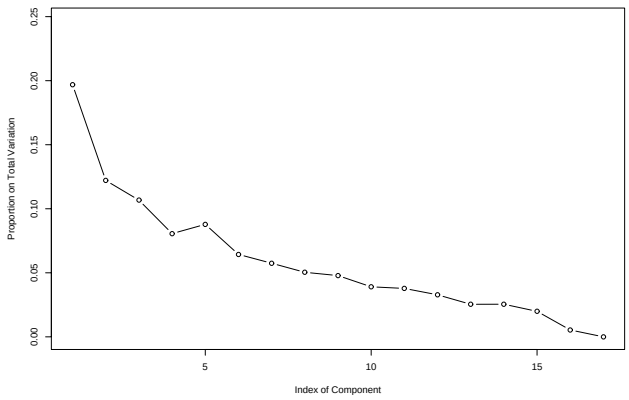


Figure 2.4: Screeplot of NIPALS for districts

showing the proportion of each of them on the total variation is a good tool for the decision on the number of components to be taken for possible further investigations. The screeplot for the results of the function *princomp* as displayed in Figure 2.3 shows a high importance of component 1 followed with successively decreasing components 2 to 7. Figure 2.4 shows the corresponding screeplot of the NIPALS algorithm. The increasing value for component 5 is an artefact of the algorithm. A good number of components could be 5 (59%) or 9 (81%). If we applied the rule of thumb that the number of components explaining at least 90% has to be taken into account, we would have to select the first 12 components. When examining big data sets the number of components to be chosen for further investigations is also a question of computation time.

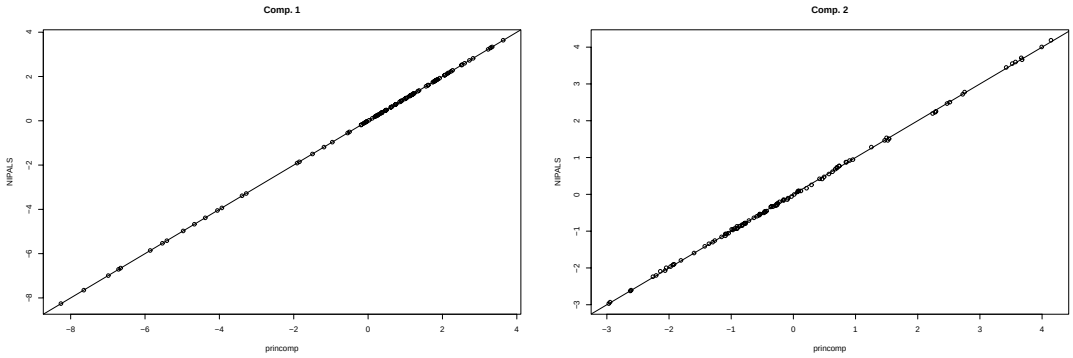


Figure 2.5: Differences of *princomp* and NIPALS in the first and second component for districts

The differences between the first and the second principal component computed by NIPALS and *princomp* are shown graphically in Figure 2.5. In the first plot the values of the first component from the two algorithms are plotted against each other, the same is done in the second plot for the second component. The first components show little difference whereas the second already have some. This is due to the small differences in the first which become bigger with every succeeding calculation of a component by construction of the NIPALS

algorithm as each higher component is calculated on the residuals of $\mathbf{X}$ and the lower components. In classical principal component analysis on the other hand the components are calculated with one singular value decomposition of $\mathbf{X}$ or by spectral decomposition of the empirical covariance matrix of $\mathbf{X}$.

2.2.2 The euro86 Data Set

Dependencies among the different variables are an important aspect of principal component analysis of the euro86 data set. As described in Appendix A life expectation, population growth and infant mortality (among others) in the European countries in 1986 are under survey.

As before we first calculated principal components with the NIPALS algorithm. The biplot of the first and second component is shown in Figure 2.6. They explain 46% of the total variation of the euro86 data set.

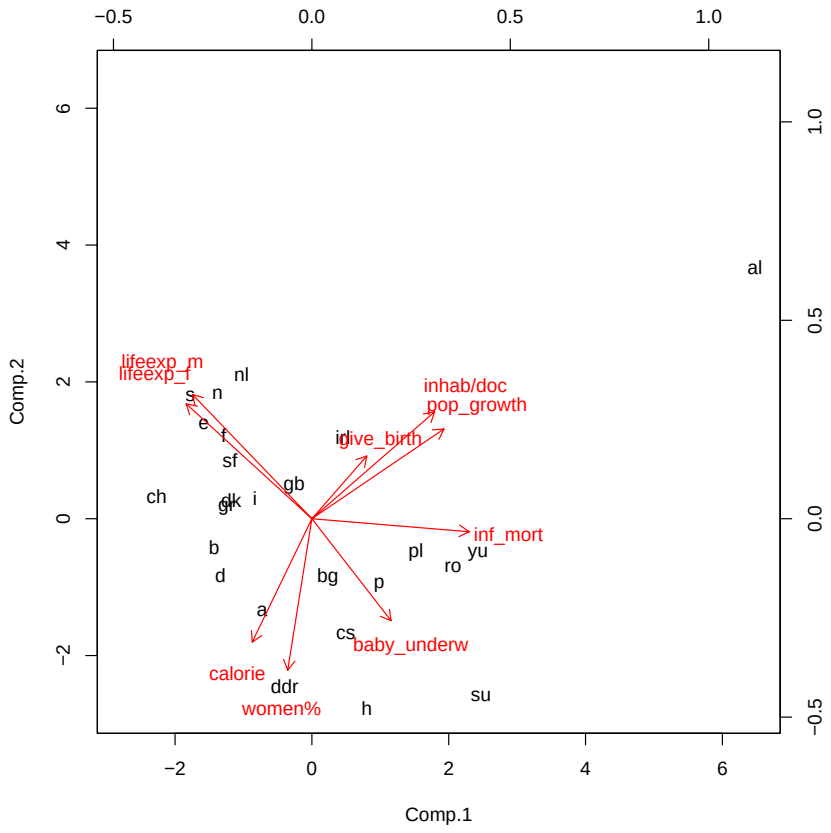


Figure 2.6: Biplot of NIPALS for euro86

A first glance at the biplot shows an outlier: Albania (al). This suspect is proved by further investigation. Its impact on the size and direction becomes clear when we look at the biplot

of the euro86 data set reduced by the observation Albania (see Figure 2.7). In this case the first and second component explain 40% of the total variation of the remaining data set.

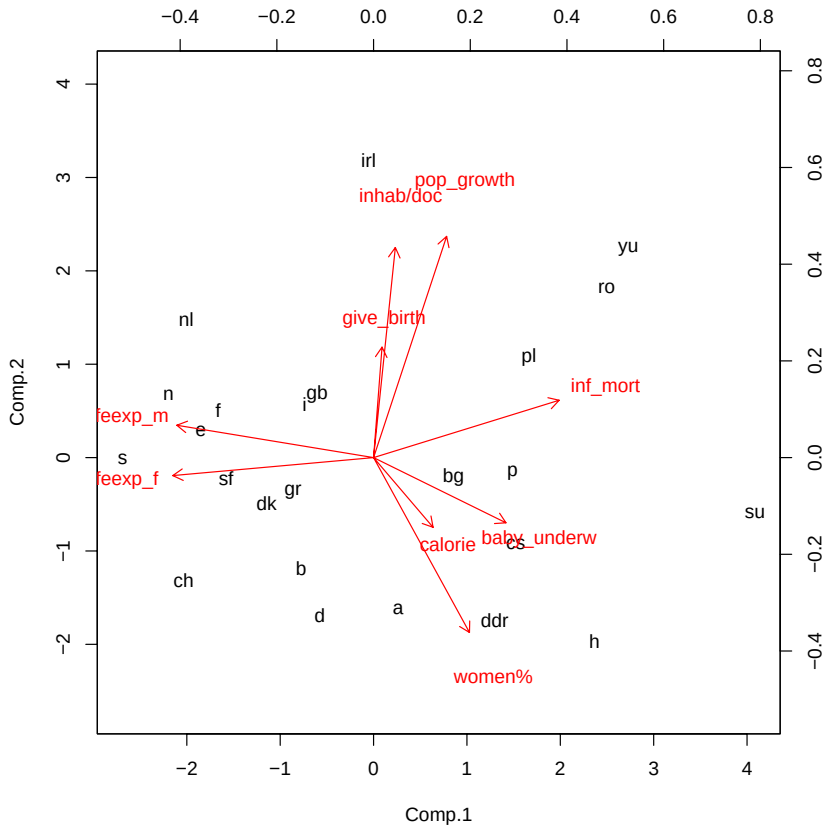


Figure 2.7: Biplot of NIPALS for euro86 without Albania (al)

The same behaviour can be found when applying *princomp* to the euro86 data set with (Figure 2.8, 46% of total variation) and without Albania (Figure 2.9, 40%).

The change in the importance of the components calculated with either of the algorithms is striking (Figures 2.10 and 2.11 for *princomp* and Figures 2.12 and 2.13 for NIPALS).

Note that in the case of the robustified data set NIPALS really gives decreasing importances. This demand is missed slightly in the case of the whole data set. While the first result with Albania shows great importance in the first and second component but only minor importance in component 3 to 5 (the rest is too small), there is a shift of importance mainly from the second towards components 3 to 6. So, in the case of further investigations on this data set, if we decided to use five components (81%) by working with the contaminated data set we would now, after removal of Albania, revise our decision and also take the sixth component into account (87%, only 78% for components 1 to 5).

As *princomp* and NIPALS again give similar results the interpretations of the first and second principal component are valid for both.

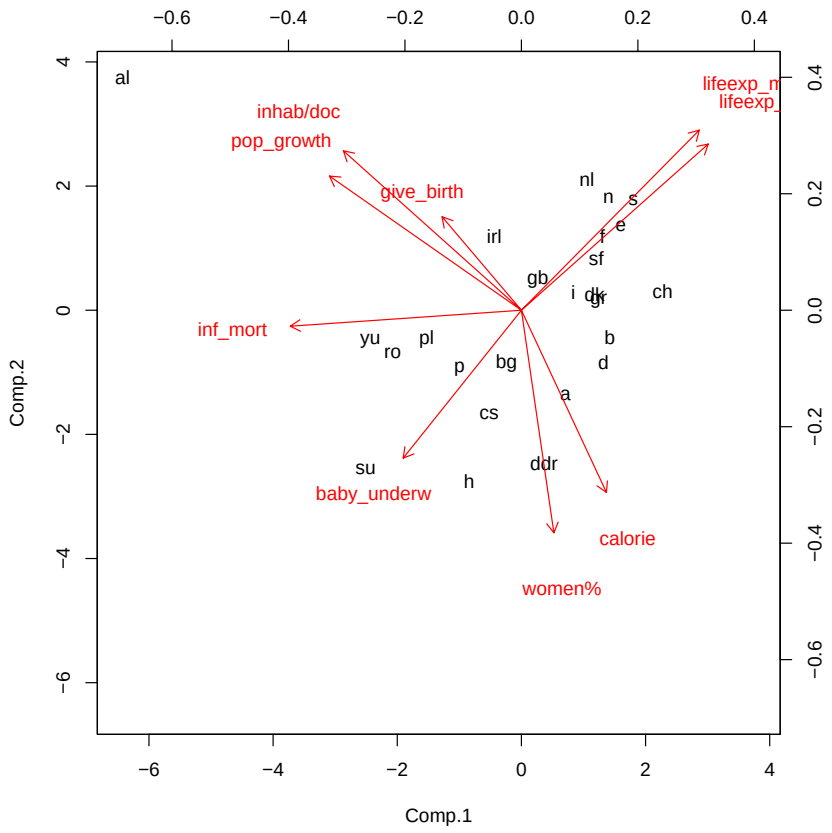


Figure 2.8: Biplot of *princomp* for euro86

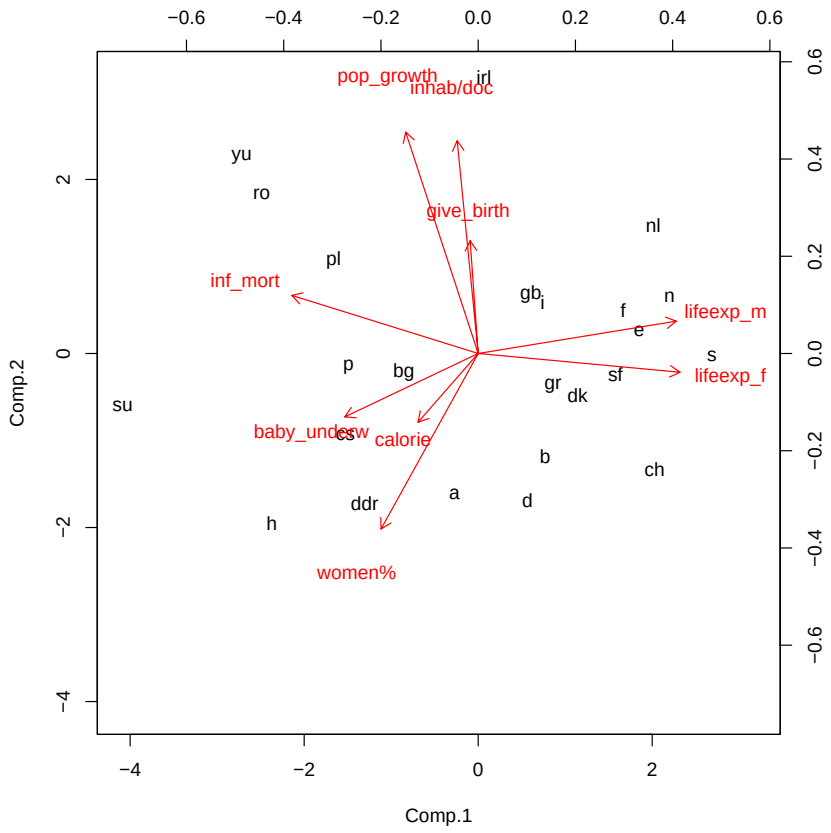


Figure 2.9: Biplot of *princomp* for euro86 without Albania (al)

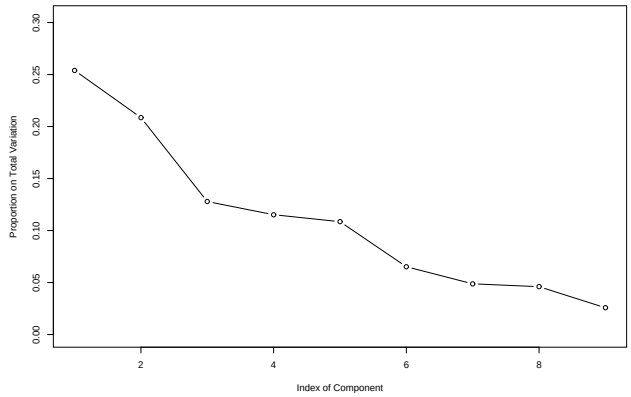


Figure 2.10: Screeplot of *princomp* for euro86

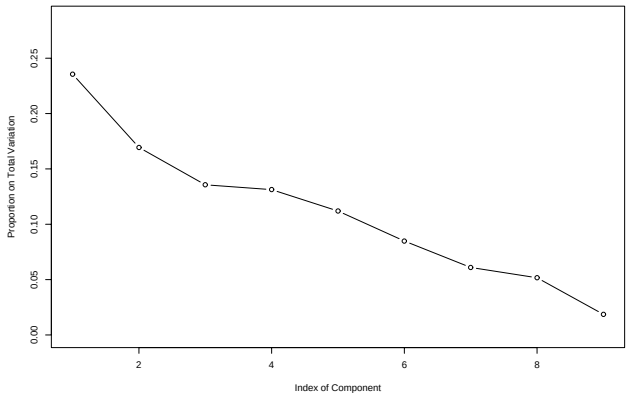


Figure 2.11: Screeplot of *princomp* for euro86 without Albania (al)

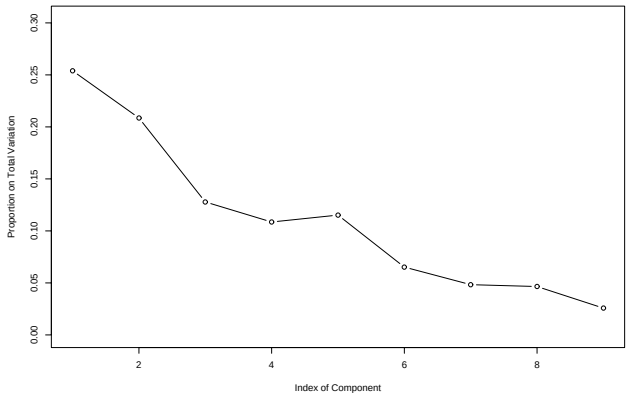


Figure 2.12: Screeplot of NIPALS for euro86

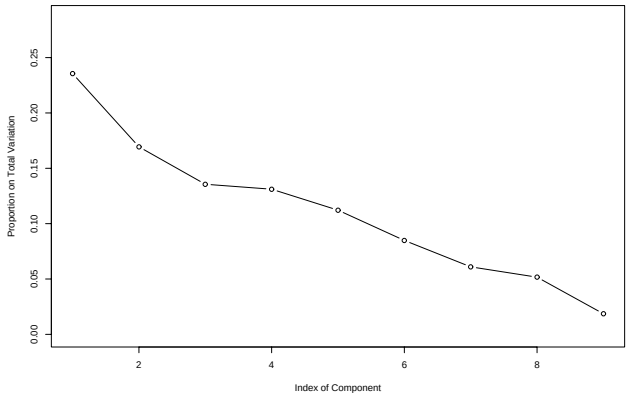


Figure 2.13: Screeplot of NIPALS for euro86 without Albania (al)

The first principal component (of euro86 without Albania) has in its negative part female and male life expectation as the only influence. In this region we find the rich countries: The northern European countries (nl, n, s, sf, f, dk, b) but also Switzerland (ch), Spain (s), Greece (gr), Germany (d), Great Britain (gb) and Italy (i). Austria (a) shows no high life expectation. Opposed to it, in the positive part, the infant mortality has strong influence along with baby underweight. This makes sense as better life circumstances reduce the cases of infant mortality.

The second principal component is expressed by high values of calories per day, baby underweight and a higher percentage of women in the negative axis and high population growth combined with women in the right age for giving birth and many inhabitants per doctor in the positive part. Again many inhabitants per doctor reduce the life quality whereas higher portions of calories are a sign of wealth. The higher percentage of women may result from a better medical infrastructure.

The differences between the components for euro86 without Albania calculated with *princomp* and NIPALS are shown in Figure 2.14. In these plots another feature can be seen: Sometimes the components differ by the multiplicative factor -1. But the corresponding loadings are also adapted. This phenomenon can also be observed when computing principal components with different statistical software packages. This results from the fact that different criteria are applied for the choice of the sign of the loadings vectors.

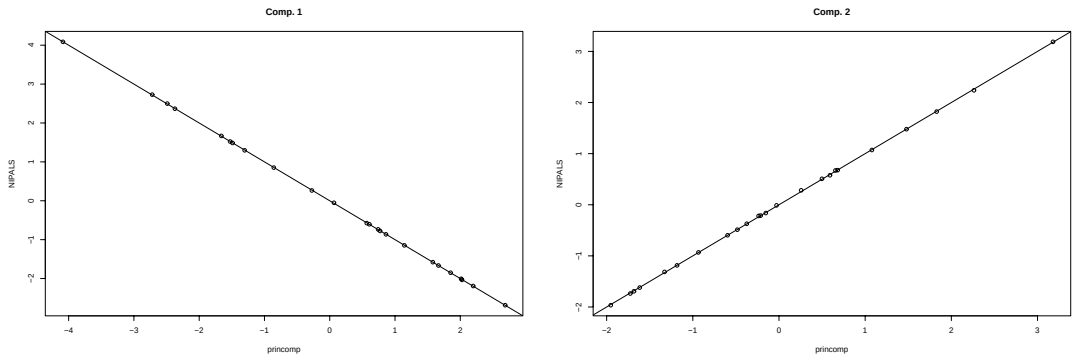


Figure 2.14: Differences of *princomp* and NIPALS in the first and second component for euro86 without Albania (al)

Chapter 3

PLS–Partial Least Squares Regression

Partial Least Squares Regression is built on the ideas of the NIPALS algorithm. Generally a regression between one or more y-variables and a data matrix $\mathbf{X}$ is performed. These variables can be correlated to each other and–as a very important part in applications–there can also be more variables than observations. The idea of performing iterations for searching the most important "component" and their refinement by a convergence criterion is adapted from the NIPALS algorithm.

We distinguish between two different PLS regressions, namely the PLS1 and the PLS2 regression. PLS1 decribes the case when there is only one y-variable and PLS2 stands for PLS with more than one y-variable.

First we will present the PLS2 algorithm which is naturally more general, and then stress the changes which lead from PLS2 to PLS1. Usually, when PLS is mentioned, the PLS2 algorithm is considered as for its generality.

The description below follows the one of Geladi and Kowalski (1986) but can also be found in Wakeling and Macfie (1992). In general, there are many different descriptions of the PLS algorithm in the literature (especially in the algorithmic details of the various steps), all of which give the main ideas of PLS but naturally differ in the mathematical properties of the estimators. The definition of the PLS algorithm preferred in this work leads to some useful properties which will be discussed later.

3.1 The PLS2 Algorithm

The relation to be analysed with PLS2 is an ordinary linear regression equation

$$\mathbf{Y} = \mathbf{X}\mathbf{K}^\top + \boldsymbol{\varepsilon},$$

where $\mathbf{X}$ and $\mathbf{Y}$ are data matrices of dimensions $n \times p$ and $n \times q$, respectively. $\mathbf{K}$ represents the matrix of the regression coefficients with dimension $q \times p$ and $\boldsymbol{\varepsilon}$ is the matrix of regression residuals of dimension $n \times q$.

Although the regression model is aimed at finding a functional relation between $\mathbf{X}$ and $\mathbf{Y}$, the idea behind PLS is somewhat different. Here one tries to get the most important common part out of both data matrices which therefore represents a kind of connection between them. In other words, two principal component analyses, one for $\mathbf{X}$ and one for $\mathbf{Y}$, are carried out using the NIPALS algorithm. But within each iteration the result from $\mathbf{X}$ is passed over to the calculation for $\mathbf{Y}$ and vice versa. As PLS shows the same characteristics as NIPALS (only twice!) also for PLS the nickname of *"criss-cross"* regression is used in literature. This complicated description gets more understandable when one looks at the algorithmic formulation given below.

According to the idea of performing a PCA for $\mathbf{X}$ and one for $\mathbf{Y}$, the model for the so-called *outer relations* is

$$\mathbf{X} = \mathbf{T}\mathbf{P}^\top + \mathbf{E} \tag{3.1}$$
$$\mathbf{Y} = \mathbf{U}\mathbf{Q}^\top + \mathbf{F}^* \tag{3.2}$$

with $\mathbf{T}$ and $\mathbf{U}$ of dimension $n \times k$, $\mathbf{P}$ of dimension $p \times k$ and $\mathbf{Q}$ of dimension $q \times k$. $\mathbf{E}$ and $\mathbf{F}^*$ are the corresponding residual matrices of dimensions $n \times p$ and $n \times q$, respectively.

The connection between $\mathbf{X}$ and $\mathbf{Y}$ is realized via the *inner relation*

$$\mathbf{U} = \mathbf{T}\mathbf{B} + \mathbf{H} \tag{3.3}$$

where $\mathbf{B}$ is a diagonal matrix of dimension $k \times k$ and the residual matrix $\mathbf{H}$ has dimension $n \times k$.

With this inner relation the final model of the *mixed relation* is

$$\mathbf{Y} = \mathbf{T}\mathbf{B}\mathbf{Q}^\top + \mathbf{F} \tag{3.4}$$

where $\mathbf{F}$ denotes the residual matrix of the mixed relation whereas $\mathbf{F}^*$ and $\mathbf{E}$ stand for the residuals of the outer relations ($\mathbf{E}$ and $\mathbf{F}$ will be used within the iterations of the algorithm).

The algorithm now seeks to obtain estimates for $\mathbf{T}$, $\mathbf{P}$, $\mathbf{U}$, $\mathbf{Q}$ and $\mathbf{B}$ and can be outlined as follows. Let $\mathbf{E}_0$ and $\mathbf{F}_0$ denote the mean-centered and scaled data matrices $\mathbf{X}$ and $\mathbf{Y}$, respectively. For every component $\mathbf{t}_h, h = 1, \ldots, k$, compute the following steps where Steps 1 to 7 are iterated:

Step 0: Choose the first column of the matrix $\mathbf{F}_{h-1}$ as a starting value for $\mathbf{u}_h$.
Step 1: Perform the following regression

$$\mathbf{E}_{h-1} = \mathbf{u}_h \mathbf{w}_h^\top + \boldsymbol{\varepsilon}_1$$

giving the least squares solution

$$\mathbf{w}_h^\top = \frac{\mathbf{u}_h^\top \mathbf{E}_{h-1}}{\mathbf{u}_h^\top \mathbf{u}_h}.$$

Step 2: Normalize $\mathbf{w}_h^\top$ obtaining the new value:

$$\mathbf{w}_h^\top := \frac{\mathbf{w}_h^\top}{\|\mathbf{w}_h\|}.$$

Step 3: Perform the regression

$$\mathbf{E}_{h-1} = \mathbf{t}_h \mathbf{w}_h^\top + \boldsymbol{\varepsilon}_2$$

which is after transposition

$$\mathbf{E}_{h-1}^\top = \mathbf{w}_h \mathbf{t}_h^\top + \boldsymbol{\varepsilon}_2^\top$$

having the least squares solution

$$\mathbf{t}_h^\top = \frac{\mathbf{w}_h^\top \mathbf{E}_{h-1}^\top}{\mathbf{w}_h^\top \mathbf{w}_h}.$$

Step 4: Perform the regression

$$\mathbf{F}_{h-1} = \mathbf{t}_h \mathbf{q}_h^\top + \boldsymbol{\varepsilon}_3$$

with the least squares solution

$$\mathbf{q}_h^\top = \frac{\mathbf{t}_h^\top \mathbf{F}_{h-1}}{\mathbf{t}_h^\top \mathbf{t}_h}.$$

Step 5: Normalize $\mathbf{q}_h^\top$ obtaining the new value

$$\mathbf{q}_h^\top := \frac{\mathbf{q}_h^\top}{\|\mathbf{q}_h\|}.$$

Step 6: Perform the following regression to calculate a new value for $\mathbf{u}_h$

$$\mathbf{F}_{h-1} = \mathbf{u}_h \mathbf{q}_h^\top + \boldsymbol{\varepsilon}_4$$

which is after transposition

$$\mathbf{F}_{h-1}^\top = \mathbf{q}_h \mathbf{u}_h^\top + \boldsymbol{\varepsilon}_4^\top$$

giving the least squares solution

$$\mathbf{u}_h^\top = \frac{\mathbf{q}_h^\top \mathbf{F}_{h-1}^\top}{\mathbf{q}_h^\top \mathbf{q}_h}$$

Step 7: Check the convergence of $\mathbf{t}_h$ with the help of a convergence criterion which will be described later on. If the newly calculated $\mathbf{t}_h$ is better than the old one according to this criterion then another iteration starting at Step 1 with the new value of $\mathbf{t}_h$ will be performed. If not go to Step 8.

Step 8: Perform the regression

$$\mathbf{E}_{h-1} = \mathbf{t}_h \mathbf{p}_h^\top + \boldsymbol{\varepsilon}_5$$

giving the least squares solution

$$\mathbf{p}_h^\top = \frac{\mathbf{t}_h^\top \mathbf{E}_{h-1}}{\mathbf{t}_h^\top \mathbf{t}_h}.$$

Step 9: Fit $\mathbf{t}_h$ to the newly gained $\mathbf{p}_h$:

$$\mathbf{t}_h = \mathbf{t}_h \|\mathbf{p}_h\|.$$

Step 10: Normalize $\mathbf{p}_h^\top$ obtaining the new value

$$\mathbf{p}_h^\top := \frac{\mathbf{p}_h^\top}{\|\mathbf{p}_h\|}.$$

Step 11: Calculate b_h, which is the hth diagonal element of B, through the regression

$$\mathbf{u}_h = \mathbf{t}_h b_h + \boldsymbol{\varepsilon}_6$$

with the least squares solution

$$b_h = \frac{\mathbf{t}_h^\top \mathbf{u}_h}{\mathbf{t}_h^\top \mathbf{t}_h}.$$

Step 12: Update the data matrices

$$\begin{aligned} \mathbf{E}_h &= \mathbf{E}_{h-1} - \mathbf{t}_h \mathbf{p}_h^\top \\ \mathbf{F}_h &= \mathbf{F}_{h-1} - \mathbf{t}_h b_h \mathbf{q}_h^\top. \end{aligned}$$

Now we give a short discussion of the algorithm:

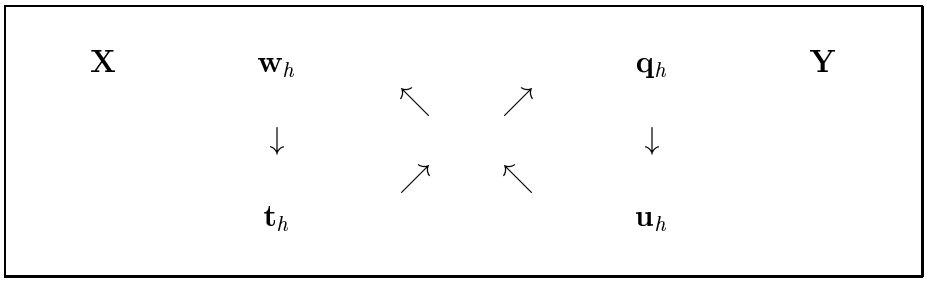
Steps 1 to 3 represent the NIPALS algorithm applied to the $\mathbf{X}$–data matrix, whereas in Steps 4 to 6 NIPALS is applied to the $\mathbf{Y}$–data. Normally, as a criterion for convergence in Step 7, the norm of the difference between the newly calculated $\mathbf{t}_h$ and the one from the previous iteration has to be smaller than a given constant. After this convergence check the $\mathbf{p}_h$ are fitted to the solution $\mathbf{t}_h$ in Step 8.

As in Step 10 the $\mathbf{p}_h$ are normalized to length 1 in order to make them comparable to principal components, the $\mathbf{t}_h$ have to be fitted to the new $\mathbf{p}_h$ in Step 9. Sometimes the demand of loadings $\mathbf{P}$ which have unit length as in principal component analysis is neglected so that the calculations in Steps 9 and 10 are not necessary anymore.

In Step 11 b_h is computed, which is not only important for updating the data matrix $\mathbf{Y}$ in Step 12 but also later for prediction.

Within Step 12 the updates of the matrices $\mathbf{X}$ and $\mathbf{Y}$ are calculated.

The graphical representation of the PLS2 algorithm underlines the idea of performing a NIPLAS algorithm twice with an additional cross-over:



3.1.1 Mathematical Properties of PLS

Properties of the Deduced Matrices

The main properties of interest are orthogonality and unit length of the columns of the matrices. The $\mathbf{t}_h$ building the matrix $\mathbf{T}$ are orthogonal, but don't have unit length. Additionally they are centered around 0, i.e., their arithmetic mean is equal 0. This makes them

uncorrelated, too. The matrix $\mathbf{W}$ is orthogonal but its columns are neither centered aroung 0 nor have unit length. The columns $\mathbf{u}_h$ of the matrix $\mathbf{U}$ are neither orthogonal nor have unit length, but they are as the $\mathbf{t}_h$ centered around 0. The elements of the matrices $\mathbf{P}$ and $\mathbf{Q}$ have unit length as is the classical demand for the loadings in principal component analysis.

A discussion of the properties of the matrices can also be found in Jackson (1991) or Geladi and Kowalski (1986).

Simplification of Least Squares Solutions

When we take a closer look at the "inner" steps of the algorithm we find that the least squares solutions for $\mathbf{t}_h$ and $\mathbf{u}_h$ can be simplified. In Step 2 we perform a normalization of $\mathbf{w}_h$ and then in Step 3 we calculate the least squares solution for $\mathbf{t}_h$:

$$\mathbf{t}_h^\top = \frac{\mathbf{w}_h^\top \mathbf{E}_{h-1}^\top}{\mathbf{w}_h^\top \mathbf{w}_h}.$$

This solution can be simplified as the denominator is equal to 1 because of the normalization of $\mathbf{w}_h$ in Step 2:

$$\mathbf{t}_h^\top = \mathbf{w}_h^\top \mathbf{E}_{h-1}^\top.$$

The same is valid for Steps 5 and 6. Because of the normalization of $\mathbf{q}_h$ the least squares solution

$$\mathbf{u}_h^\top = \frac{\mathbf{q}_h^\top \mathbf{F}_{h-1}^\top}{\mathbf{q}_h^\top \mathbf{q}_h}$$

can be replaced by

$$\mathbf{u}_h^\top = \mathbf{q}_h^\top \mathbf{F}_{h-1}^\top.$$

In the presentation of the PLS2 algorithm we kept the unsimplified version as we think the idea of the algorithm is easier to understand with the full least squares solution. Additionally when it comes to robustification of PLS with other methods than least squares this simplification may not be applicable anymore so that it could lead to some confusions.

Updating the Data Matrices

The reason of the usage of $\mathbf{F}$ instead of $\mathbf{F}^*$ is the goal of PLS to *predict* values. This is guaranteed by the usage of the mixed relation (Equation (3.4)) instead of the outer relation for the $\mathbf{Y}$–matrix (Equation (3.2)). Besides the rank of $\mathbf{Y}$ is not decreased by 1 for each component, so it is really possible to deduce the maximal number of components, namely as many as the rank of $\mathbf{X}$.

3.1.2 Number of Components

The decision on the number of components extracted from the $\mathbf{X}$–data is a very important question in PLS. In the extreme case as many components as the rank of $\mathbf{X}$ can be deduced.

But this does normally not make sense. On one hand PLS is often applied when one wants to explain a data set consisting of many variables with a few components describing the main characteristics of this data set and on the other hand the last components, which are smaller in importance, often only describe *"noise"* in the data set and are affected with the problem of collinearity.

There are several possibilites to decide on the number of PLS components, often referred to as *stopping rules*.

Calculation of the Residual of the Mixed Relation

After the determination of a new component $\|\mathbf{F}_h\|$ is calculated. Similar to a screeplot a plot of $\|\mathbf{F}_h\|$ versus the number of components can be created. By choice of a threshold for $\|\mathbf{F}_h\|$ a possible stopping rule is defined.

Also based on the residual of the mixed relation is the following stopping rule: Calculate the difference between $\|\mathbf{F}_h\|$ and $\|\mathbf{F}_{h-1}\|$ and stop when this difference becomes smaller than some predefined error.

Geladi and Kowalski (1986) suggest a combination of the threshold and difference method.

Analysis of Variance

For the inner relation (Equation (3.3)) an analysis of variance (with F-test) can be performed. The significance of various numbers of components to a chosen level of significance is determined. For a short discussion of this stopping rule see Jackson (1991).

Cross-Validation

If prediction by usage of the PLS components is wanted the usual choice for the decision on the number of components is *cross-validation*, a *resampling method*. Resampling methods also include the *jackknife* and the *bootstrap*. For detailed information on these methods (see, e.g., Basilevsky, 1994).

In the case of PLS cross-validation is used, see e.g., Rao and Toutenberg (1995) or Jackson (1991). The main idea is to divide the data set into groups. Afterwards the goodness of fit is verified by estimating the model in one group and applying it to the rest of the groups. Then the sum of squares of the difference between the true (observed) and the predicted values is calculated. This computation is done for every group such that every data point is left out exactly once. The resulting total sum of squares of predictions minus observations, called *PRESS* (Prediction Residual Sum of Squares) is a measure of the predictive power of the h components calculated in this phase of the model building process. The following relation then serves as a stopping rule for the number of components:

$$\mathrm{PRESS}_{h+1} - \mathrm{PRESS}_h < \mathrm{const},$$

where const is a certain chosen constant and PRESS_h denotes the PRESS-statistic calculated after the hth component was determined.

3.1.3 Prediction with PLS

Prediction is in practice the most important feature of PLS. Unlike other regression models it is necessary to perform some calculations before prediciton of the unknown $\mathbf{Y}$–part from the $\mathbf{X}$–part is possible.

Suppose the two data matrices $\mathbf{X}$ and $\mathbf{Y}$ be given and the PLS2 algorithm already applied as above such that the matrices $\mathbf{T}$, $\mathbf{P}$, $\mathbf{U}$, $\mathbf{Q}$, $\mathbf{W}$ and $\mathbf{B}$ are specified. Now let an additional $\mathbf{X}'$–block of size $r \times p$ be given. From this independent part we wish to predict the dependent one $\mathbf{Y}'$. For this prediction we use the model parameters calculated before but we have to adjust the matrix $\mathbf{T}$ to the new $\mathbf{X}'$. Note that it is necessary to apply the *same* mean-centering and scaling to $\mathbf{X}'$ as to $\mathbf{X}$ as otherwise the model parameters would not fit anymore!

Let $\mathbf{E}_0'$ denote the mean-centered and scaled data matrix $\mathbf{X}'$, where from each column of $\mathbf{X}'$ we substract the corresponding mean of the columns of $\mathbf{X}$ and divide by the standard deviation of the corresponding $\mathbf{X}$–columns. For $h = 1, \ldots, k$ perform the following Steps a and b to get fitted values for $\mathbf{t}_h$:

Step a: Estimate $\mathbf{t}_h$ with the help of $\mathbf{w}_h$ as in the regression equation from Step 3 of the PLS2 algorithm:

$$\widehat{\mathbf{t}}_h^\top = \frac{\mathbf{p}_h^\top \mathbf{E}_{h-1}'^\top}{\mathbf{p}_h^\top \mathbf{p}_h}.$$

Step b: Update the data matrix for the calculation of the next component

$$\mathbf{E}_h' = \mathbf{E}_{h-1}' - \widehat{\mathbf{t}}_h \mathbf{p}_h^\top$$

With the help of these values $\widehat{\mathbf{t}}_h$ we get the prediction for the $\mathbf{Y}$–matrix:

$$\widehat{\mathbf{Y}} = \widehat{\mathbf{F}}_h = \sum_{h=1}^{k} b_h \widehat{\mathbf{t}}_h \mathbf{q}_h^\top$$

3.1.4 PLS2 for Missing Values

The idea for PLS2 for missing values is the same as for NIPALS for missing values and can again be found in the book of Tenenhaus (1998a). As PLS2 can be interpreted as double application of NIPALS accordingly we have to replace 4 steps relying only on the given values.

For simplicity of notation, $\mathbf{E}_{h-1}$ will be denoted by $\tilde{\mathbf{E}} = [\tilde{\mathbf{e}}_{ij}]_{n \times p}$ and $\mathbf{F}_{h-1}$ by $\tilde{\mathbf{F}} = [\tilde{\mathbf{f}}_{ij}]_{n \times q}$. To make PLS2 applicable for data matrices with missing values replace Step 1 by

Step 1^***:** For $j = 1, 2, \ldots, p$ calculate:

$$w_{jh} = \frac{\sum_{i=1, \text{ where } \tilde{e}_{ij} \text{ and } u_{ih} \text{ exist}}^{n} \tilde{e}_{ij} u_{ih}}{\sum_{i=1, \text{ where } \tilde{e}_{ij} \text{ and } u_{ih} \text{ exist}}^{n} u_{ih}^2}$$

and Step 3 by

Step 3^***:** For $i = 1, 2, \ldots, n$ calculate:

$$t_{ih} = \frac{\sum_{j=1, \text{ where } \tilde{e}_{ij} \text{ exists}}^{p} \tilde{e}_{ij} w_{jh}}{\sum_{j=1, \text{ where } \tilde{e}_{ij} \text{ exists}}^{p} w_{jh}^2}.$$

Consequently replace Step 4 by

Step 4^***:** For $j = 1, 2, \ldots, q$ calculate:

$$q_{jh} = \frac{\sum_{i=1, \text{ where } \tilde{f}_{ij} \text{ and } t_{ih} \text{ exist}}^{n} \tilde{f}_{ij} t_{ih}}{\sum_{i=1, \text{ where } \tilde{f}_{ij} \text{ and } t_{ih} \text{ exist}}^{n} t_{ih}^2}$$

and Step 6 by

Step 6^***:** For $i = 1, 2, \ldots, n$ calculate:

$$u_{ih} = \frac{\sum_{j=1, \text{ where } \tilde{f}_{ij} \text{ exists}}^{q} \tilde{f}_{ij} q_{jh}}{\sum_{j=1, \text{ where } \tilde{f}_{ij} \text{ exists}}^{q} q_{jh}^2}.$$

Application of this modified algorithm returns the PLS components and scores related to the data set with the missing values.

3.2 The PLS1 Algorithm

The only difference between the model for PLS1 and PLS2 is the dimension of the y–part. While in PLS2 $\mathbf{Y}$ has dimension $n \times q$, we have a one-dimensional vector $\mathbf{y}$ in PLS1. The differences in the PLS1 and PLS2 algorithm now only result from this dimensional change. So first we want to give a version of the PLS1 algorithm as a simple translation of the PLS2 algorithm to the reduced dimensionality. Then we want to present the simplifications which become possible. Finally we give the short version of the PLS1 algorithm where all these simplifications are already considered.

The PLS1 algorithm as a dimensional reduction of the PLS2 algorithm looks as follows:

Let $\mathbf{E}_0$ and $\mathbf{f}_0$ denote the mean-centered and scaled data matrix $\mathbf{X}$ and vector $\mathbf{y}$, respectively. For every component $\mathbf{t}_h, h = 1, \ldots, k$, compute the following steps:

Step 0: Choose $\mathbf{f}_{h-1}$ as a starting value for $\mathbf{u}_h$.
Step 1: Perform the following regression

$$\mathbf{E}_{h-1} = \mathbf{u}_h \mathbf{w}_h^\top + \boldsymbol{\varepsilon}_1$$

giving the least squares solution

$$\mathbf{w}_h^\top = \frac{\mathbf{u}_h^\top \mathbf{E}_{h-1}}{\mathbf{u}_h^\top \mathbf{u}_h}.$$

Step 2: Normalize $\mathbf{w}_h^\top$ obtaining the new value:

$$\mathbf{w}_h^\top := \frac{\mathbf{w}_h^\top}{\|\mathbf{w}_h\|}.$$

Step 3: Perform the regression

$$\mathbf{E}_{h-1} = \mathbf{t}_h \mathbf{w}_h^\top + \boldsymbol{\varepsilon}_2$$

which is after transposition

$$\mathbf{E}_{h-1}^\top = \mathbf{w}_h \mathbf{t}_h^\top + \boldsymbol{\varepsilon}_2^\top$$

having the least squares solution

$$\mathbf{t}_h^\top = \frac{\mathbf{w}_h^\top \mathbf{E}_{h-1}^\top}{\mathbf{w}_h^\top \mathbf{w}_h}.$$

Step 4: Perform the regression

$$\mathbf{f}_{h-1} = \mathbf{t}_h q_h + \boldsymbol{\varepsilon}_3$$

with the least squares solution

$$q_h = \frac{\mathbf{t}_h^\top \mathbf{f}_{h-1}}{\mathbf{t}_h^\top \mathbf{t}_h}.$$

Step 5: Normalize q_h obtaining the new value

$$q_h := \frac{q_h}{\|q_h\|}.$$

Step 6: Perform the following regression to calculate a new value for $\mathbf{u}_h$

$$\mathbf{f}_{h-1} = \mathbf{u}_h q_h + \boldsymbol{\varepsilon}_4$$

which is after transposition

$$\mathbf{f}_{h-1}^\top = q_h \mathbf{u}_h^\top + \boldsymbol{\varepsilon}_4^\top$$

giving the least squares solution

$$\mathbf{u}_h^\top = \frac{q_h \mathbf{f}_{h-1}^\top}{q_h q_h}$$

Step 7: Check the convergence of $\mathbf{t}_h$ with the help of a convergence as described in Section 3.1. If the newly calculated $\mathbf{t}_h$ is better than the old one according to this criterion then another iteration starting at Step 1 with the new value of $\mathbf{t}_h$ will be performed. If not go to Step 8.

Step 8: Perform the regression

$$\mathbf{E}_{h-1} = \mathbf{t}_h \mathbf{p}_h^\top + \boldsymbol{\varepsilon}_5$$

giving the least squares solution

$$\mathbf{p}_h^\top = \frac{\mathbf{t}_h^\top \mathbf{E}_{h-1}}{\mathbf{t}_h^\top \mathbf{t}_h}.$$

Step 9: Fit $\mathbf{t}_h$ to the newly gained $\mathbf{p}_h$:

$$\mathbf{t}_h = \mathbf{t}_h \|\mathbf{p}_h\|.$$

Step 10: Normalize $\mathbf{p}_h^\top$ obtaining the new value

$$\mathbf{p}_h^\top := \frac{\mathbf{p}_h^\top}{\|\mathbf{p}_h\|}.$$

Step 11: Calculate b_h, which is the hth diagonal element of B, through the regression

$$\mathbf{u}_h = \mathbf{t}_h b_h + \boldsymbol{\varepsilon}_6$$

with the least squares solution

$$b_h = \frac{\mathbf{t}_h^\top \mathbf{u}_h}{\mathbf{t}_h^\top \mathbf{t}_h}.$$

Step 12: Update the data matrices

$$\begin{aligned} \mathbf{E}_h &= \mathbf{E}_{h-1} - \mathbf{t}_h \mathbf{p}_h^\top \\ \mathbf{f}_h &= \mathbf{f}_{h-1} - \mathbf{t}_h b_h q_h. \end{aligned}$$

Note that through the regression for $\mathbf{q}_h$ as in PLS2 we receive a scalar q_h and by normalizing it in the next step we get the value 1. Therefore there is no change in $\mathbf{u}_h = \mathbf{f}_{h-1}$ as would be in PLS2 in Step 6. By performing the regression for b_h and subtracting $\mathbf{t}_h b_h q_h$ from $\mathbf{f}_h$, we get a residual $\mathbf{f}_h$ from which further PLS components can be calculated. Still the normalization of q_h is important for a good interpretation of the resulting PLS components.

For the same reason, namely q_h being a scalar in PLS1 and its effects on $\mathbf{u}_h$, no iterations or convergence checks as in PLS2 are meaningful, so Step 7 can be omitted.

As q_h is always equal 1 and $\mathbf{u}_h$ stays equal to $\mathbf{f}_{h-1}$ these variables can be replaced by these values in the whole algorithm and Steps 4 to 6 can be omitted, too.

As was discussed in Section 3.1.1 for the PLS2 algorithm some simplifications of the least squares solutions also come to pass in PLS1. In Step 2 of PLS1 we perform a normalization of $\mathbf{w}_h$ and then calculate $\mathbf{t}_h$ in Step 3:

$$\mathbf{t}_h^\top = \frac{\mathbf{w}_h^\top \mathbf{E}_{h-1}^\top}{\mathbf{w}_h^\top \mathbf{w}_h}.$$

So Step 3 can be replaced by:

$$\mathbf{t}_h^\top = \mathbf{w}_h^\top \mathbf{E}_{h-1}^\top.$$

All these simplifications together give the following final version of the PLS1 algorithm (under the condition of performing all regressions with least squares):

For every component $\mathbf{t}_h, h = 1, \ldots, k$, compute the following steps:

Step 1:

$$\mathbf{w}_h^\top = \frac{\mathbf{f}_{h-1}^\top \mathbf{E}_{h-1}}{\mathbf{f}_{h-1}^\top \mathbf{f}_{h-1}}.$$

Step 2:

$$\mathbf{w}_h^\top := \frac{\mathbf{w}_h^\top}{\|\mathbf{w}_h\|}.$$

Step 3:

$$\mathbf{t}_h^\top = \mathbf{w}_h^\top \mathbf{E}_{h-1}^\top.$$

Step 4:

$$\mathbf{p}_h^\top = \frac{\mathbf{t}_h^\top \mathbf{E}_{h-1}}{\mathbf{t}_h^\top \mathbf{t}_h}.$$

Step 5:

$$\mathbf{t}_h = \mathbf{t}_h \|\mathbf{p}_h\|.$$

Step 6:

$$\mathbf{p}_h^\top := \frac{\mathbf{p}_h^\top}{\|\mathbf{p}_h\|}.$$

Step 7:

$$b_h = \frac{\mathbf{t}_h^\top \mathbf{f}_{h-1}}{\mathbf{t}_h^\top \mathbf{t}_h}.$$

Step 8:

$$\begin{aligned} \mathbf{E}_h &= \mathbf{E}_{h-1} - \mathbf{t}_h \mathbf{p}_h^\top \\ \mathbf{f}_h &= \mathbf{f}_{h-1} - \mathbf{t}_h b_h. \end{aligned}$$

The demand of loadings $\mathbf{P}$ which have unit length is sometimes neglected by omitting Steps 5 and 6. This is often done in surveys on robust PLS procedures.

3.3 Examples

The following examples show on one hand that the PLS components really differ from principal components and on the other hand they show the power of these PLS components for the explanation of the combined data sets. The first example uses the PLS2 algorithm whereas in the second the PLS1 algorithm is applied.

Following the ideas of the presentation of our results with the NIPALS algorithm we will now present the results for PLS2 in an analogous form. We will show biplots of the "scores" $\mathbf{T}$ and "loadings" $\mathbf{P}$ derived from the $\mathbf{X}$–matrix and the "scores" $\mathbf{U}$ and "loadings" $\mathbf{Q}$ from the $\mathbf{Y}$–matrix. Additionally we will compare the PLS components from the $\mathbf{X}$– and $\mathbf{Y}$–part as the connections between $\mathbf{X}$ and $\mathbf{Y}$ are of crucial interest. Therefore we will present a graphical comparison of the first PLS components, i.e., the first component $\mathbf{t}_1$ calculated from $\mathbf{X}$ against the first component $\mathbf{u}_1$ from $\mathbf{Y}$, and also of the second PLS comonents $\mathbf{t}_2$ and $\mathbf{u}_2$. As we will see later these graphical representations yield very interesting results.

3.3.1 Example for PLS2: The oeamtc Data Set

The oeamtc (abbreviation of the Austrian automobile club) data set is probably the most suitable example for usage with PLS2 as it examines the (21) characteristic features of 18 quite different cars.

The variables price, tax, liability and full comprehensive insurance taken as **Y**–data should show dependencies on the rest of the variables (used as **X**–data). The most striking one is the fact that the tax and liability insurance in Austria are calculated from the performance of the car.

In Appendix A a detailed description of the data set is given. In Table A.6 the names of all the cars and their corresponding numbers in the data set are given as it is necessary with respect to readability to use those numbers in all plots.

PLS2 Applied on the oeamtc Data Set

The results of the application of PLS2 on the oeamtc data set are shown graphically. In Figure 3.1 a biplot of the first and second component of the **X**–data is printed whereas Figure 3.2 shows the corresponding biplot for the **Y**–data.

Figures 3.3 and 3.4 show the screeplots of the components of the **X**– and **Y**–data, respectively. Again we find some artefacts of increasing values which are due to the algorithm.

So the first two components of the **X**–data, which are shown in the biplot (Figure 3.1) explain 49% of the total variation. According to the screeplot a good number of components could be 6 (79%) or–when applying the rule of thumb to take at least 90%–9 (91%). The corresponding values for the importance of the **Y**–components are 32% (2 components), 59% (6 components) and 76% (9 components).

Let us now discuss the components of the **X**–data. The first component shows "surrounding" characteristics such as length, width, boot or consumption. It is interesting that also capacity contributes to the first component. On the other hand the second component consists mainly of the variables speed and performance and opposed to them (which also makes sense from the technical point of view) are acceleration, elasticity and height.

The components of the **Y**–data are not so clearly distinguished. The first component is more or less influenced by all four variables whereas the second mainly by tax and liability insurance. The most interesting fact in this figure is maybe the confirmation of the tight relation between tax and liability insurance and price and full comprehensive insurance, respectively.

The plot of the first PLS components $\mathbf{t}_1$ and $\mathbf{u}_1$ against each other (Figure 3.5) gives quite interesting relations among **X** and **Y**. In the first quadrant cars with high length, width, boot, consumption together with high price, tax and insurances occur, e.g., Chrysler Voyager (4), Mercedes C220 (2) or Hyundai Trajet (5). Also in this direction but not that strong are the so-called compact vans Opel Zafira (17), Nissan Almera (18) or Fiat Multipla (14). The second quadrant, standing for lower consumption and size combined with a higher price,

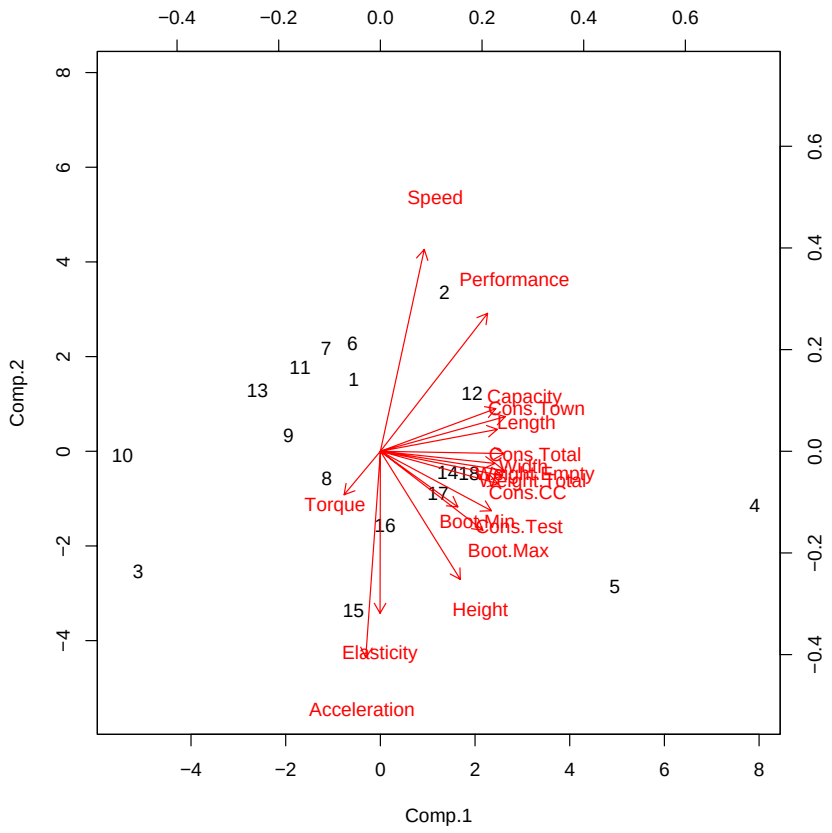


Figure 3.1: Biplot of PLS2 for **X**–data of oeamtc

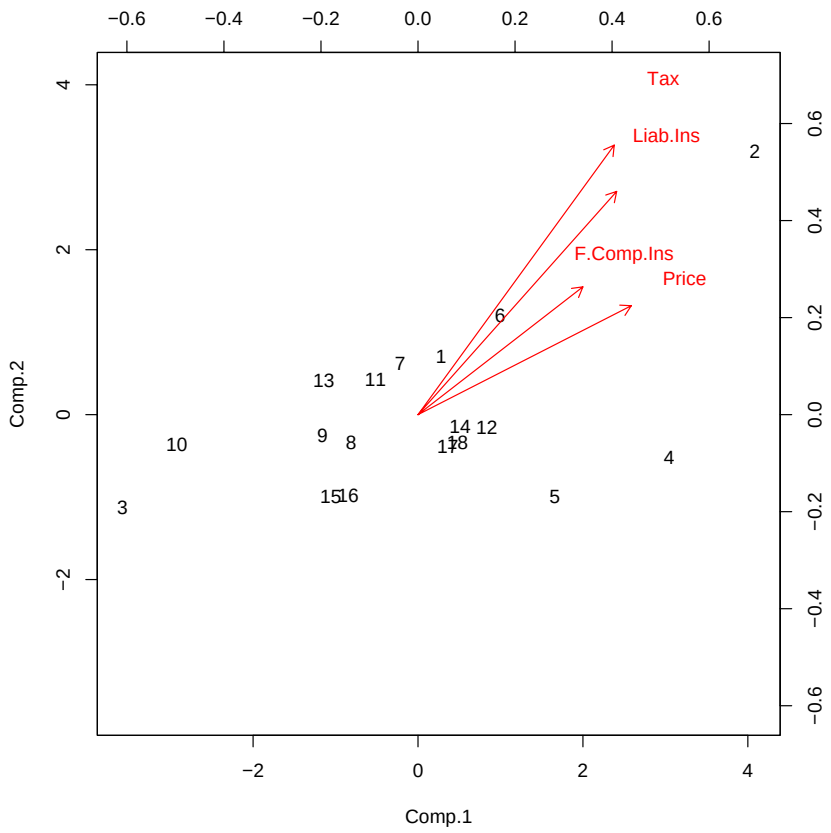


Figure 3.2: Biplot of PLS2 for **Y**–data of oeamtc

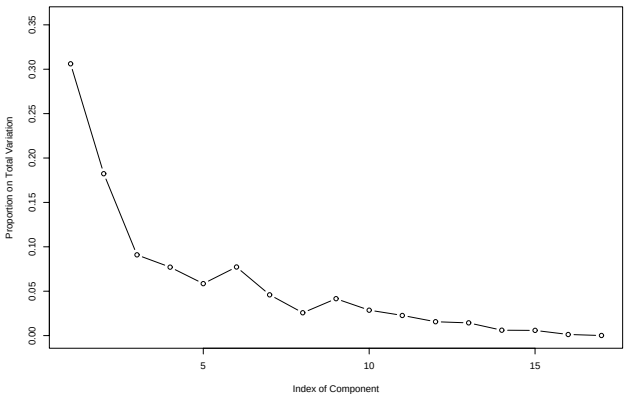


Figure 3.3: Screeplot of PLS2 components for **X**–data of oeamtc

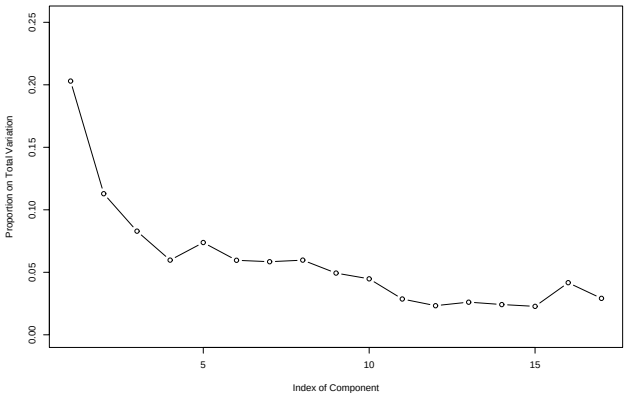


Figure 3.4: Screeplot of PLS2 components for $\mathbf{Y}$–data of oeamtc

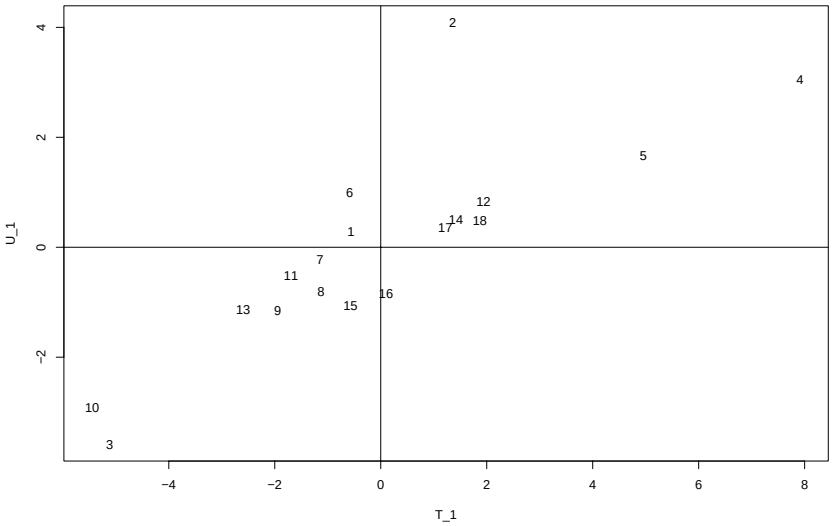


Figure 3.5: First PLS2 components for oeamtc

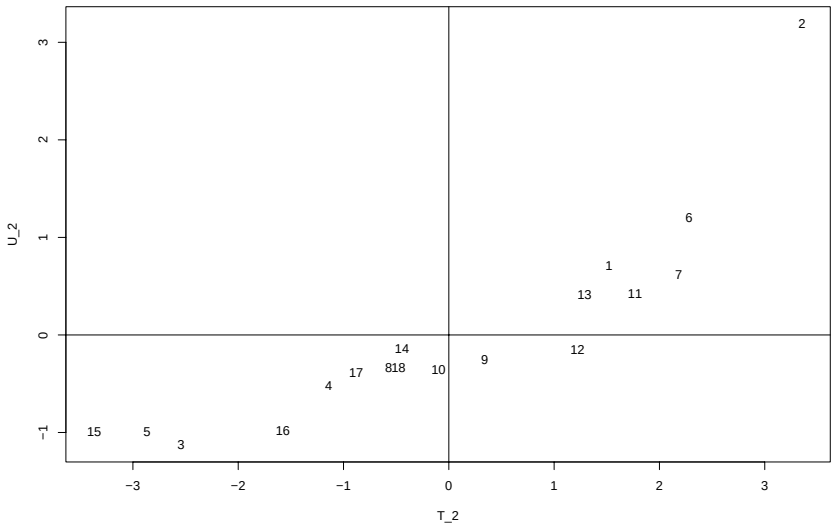


Figure 3.6: Second PLS2 components for oeamtc

houses the Fiat Stilo (1) and Alfa Romeo 147 (6). Most of the cars under survey can be found in the third quadrant, lower consumption and size mixed with lower price. The extremest ones are–as can be expected–the smallest cars, Renault Clio (3) and Seat Arosa (10). Still in this part are the compact cars such as Skoda Fabia (13), Ford Focus (7), VW Golf (9). On the boundary to bigger size, still with lower price, is only the Citroen Picasso (16).

Figure 3.6 shows the second PLS components $\mathbf{t}_2$ against $\mathbf{u}_2$. In the first quadrant, standing for high speed and high taxes, the Mercedes C220 (2) is the clear favorite, followed with a big gap by Alfa Romeo 147 (6), Ford Focus (7) and Fiat Stilo (1). Fortunately, the second quadrant (high taxes and low speed) is completely empty which, however, reflects the policy of insurance companies. The third quadrant stands for low speed and low tax. The minimum is met by Renault Clio (3), Renault Scenic (15) and Hyundai Trajet (5). The majority of the rest of the cars can also be found in this part. An interesting point is that the second small car, the Seat Arosa (10), seems to have quite a good highest speed, unlike his companion regarding size and price, the Renault Clio (3). Finally in the fourth quadrant, high speed and low taxes, are two representatives, the VW Golf (9) and the Ford Mondeo (12).

PLS2 Applied on the oeamtc Data Set Without Mercedes C220 (2)

As the Mercedes C220 (2) appears to be an outlier according to plots just presented we did another survey without it. The results can be found in Figures 3.7 to 3.12. The components for the $\mathbf{X}$–data show no great changes, but when we look at the $\mathbf{Y}$–data we see that there is a better distinction between the first and second component. This observation goes on in the plots of the first and second components against each other. The plot of the first components shows no great changes whereas in that of the second a clearer distinction regarding speed and tax can be observed. Regarding the importance of the components we observe slightly

increasing importance in the components of the **X**–data (2 components: 50%; 6: 80%; 9: 92%) whereas the importance of the components of the **Y**–data decreases (2 components: 29%; 6: 54%; 9: 70%).

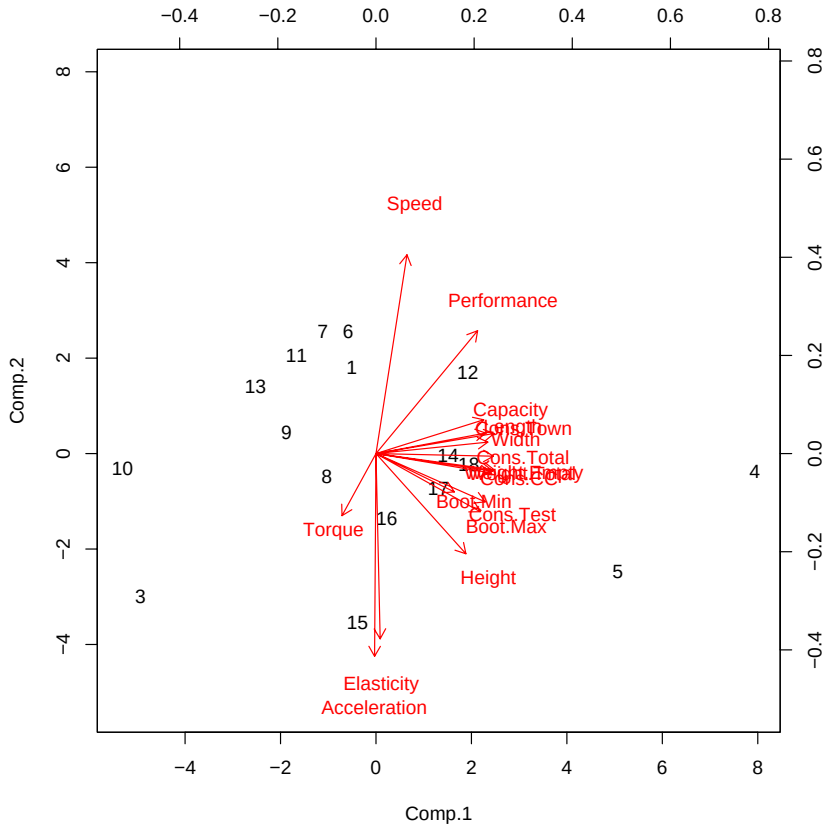


Figure 3.7: Biplot of PLS2 for **X**–data of oeamtc without Mercedes C220 (2)

Prediction for the oeamtc Data Set Without Mercedes C220 (2)

We now want to present the results of prediction for PLS2 as described in Section 3.1.3 on the oeamtc data set without Mercedes C220 (2).

In order to receive comparable results we took the first three cars from the oeamtc data set without Mercedes C220 (Fiat Stilo (1), Renault Cio (3), Chrysler Voyager (4)) and compared the predicted results to the ones introduced above. Figure 3.13 plots the true against the predicted values of the scores **T**. According to these results the prediction is quite good.

In Figure 3.14 we see the plot of true against predicted **Y**–values. Also these predictions represent the characteristics of the data set. The deviations which are visible result from the fact that each of the three chosen cars has specific distinctions from the whole data set: The Renault Clio (3) is one of the smallest cars, the Chrysler Voyager (4) is the biggest one and the Fiat Stilo (1) shows some innovations regarding the "surrounding" characteristics.

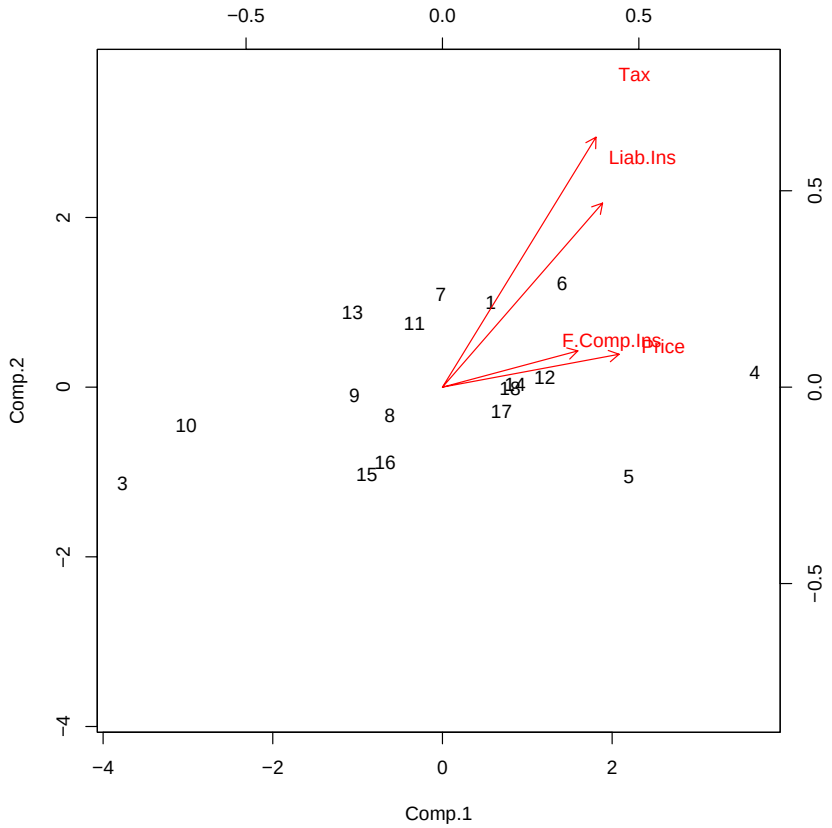


Figure 3.8: Biplot of PLS2 for **Y**–data of oeamtc without Mercedes C220 (2)

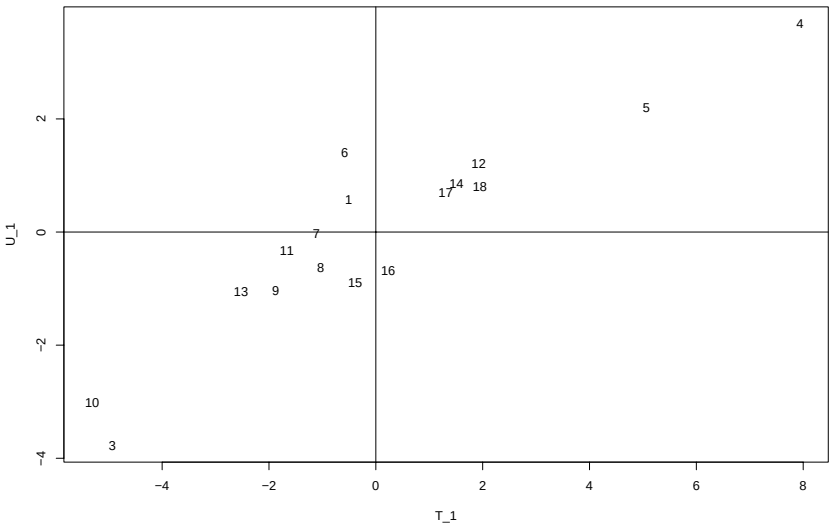


Figure 3.9: First PLS2 components for oeamtc without Mercedes C220 (2)

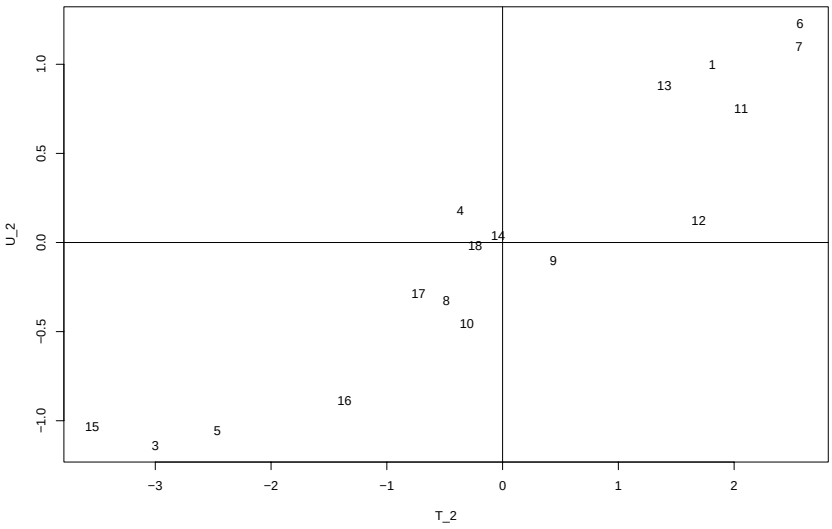


Figure 3.10: Second PLS2 components for oeamtc without Mercedes C220 (2)

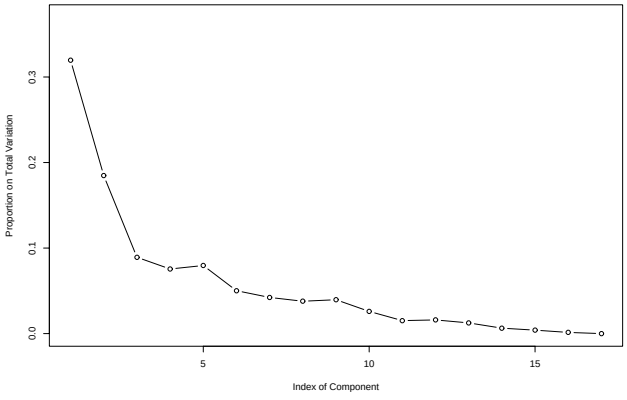


Figure 3.11: Screeplot of PLS2 components for **X**–data of oeamtc without Mercedes C220 (2)

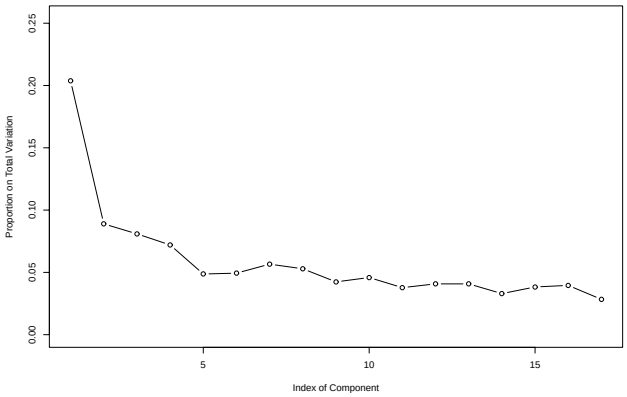


Figure 3.12: Screeplot of PLS2 components for $\mathbf{Y}$–data of oeamtc without Mercedes C220 (2)

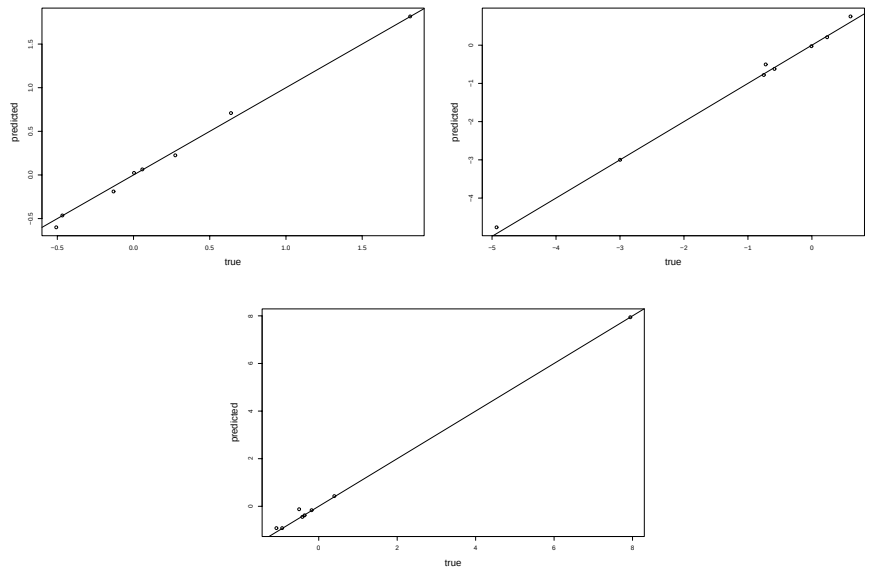


Figure 3.13: Comparison of true and predicted scores $\mathbf{T}$ of Fiat Stilo (1) (upper left), Renault Clio (3) (upper right) and Chrysler Voyager (4) (lower)

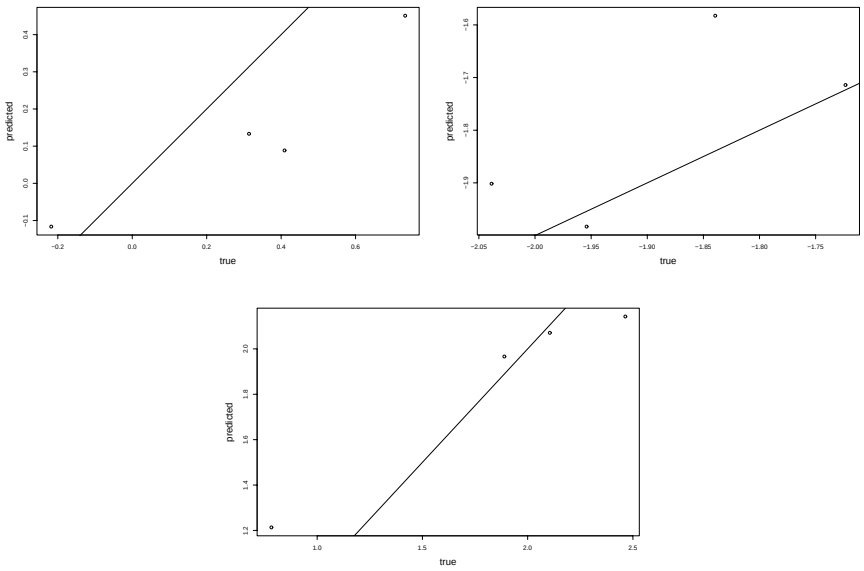


Figure 3.14: Comparison of true and predicted **Y**–values of Fiat Stilo (1) (upper left), Renault Clio (3) (upper right) and Chrysler Voyager (4) (lower)

3.3.2 Example for PLS1: The euro86 Data Set

We will now study the results for the euro86 data set when applying the PLS1 algorithm. First we have to divide the data set into a **X**– and **y**–part. For the **X**–data matrix we chose the following variables: number of women in the age to give birth, number of inhabitants per doctor, baby underweigth, percentage of women, infant mortality, population growth and calories supplied per day. As **y**–variable female life expectation was selected. Note that male life expectation was not taken into account for reasons of redundancy as we have seen in Section 2.2 that female and male life expectation are variables influencing the components in nearly the same way. Additionally, as we could see that Albania is an outlier when studying the results of NIPALS on the euro86 data set, we focused on the reduced data set.

Figure 3.15 shows the biplot of the first and second component of the **X**–data.

The first component is strongly (negative) influenced by infant mortality and baby underweight. But also the percentage of women, the population growth and supplied calories per day are a main part of this component. The number of women in the age to give birth is the main part of the second component. The number of inhabitants per doctor and calorie supply also have great influence. Furthermore all of the other variables except the percentage of women more or less determine the second component. In the part of the biplot describing high infant mortality and population growth (i.e., the negative part of the first component) we find most eastern countries (ro, h, su, ddr, bg, cs), combined with baby underweight we also see yu, pl and p. The western countries all have low values for infant mortality, baby underweight and percentage of women. They can be divided into countries with a higher number of women in the age to give birth and inhabitants per doctor, mostly the northern countries (nl, dk, n, sf, s, gb), Germany (d), Italy (i) and Austria (a), and those with lower

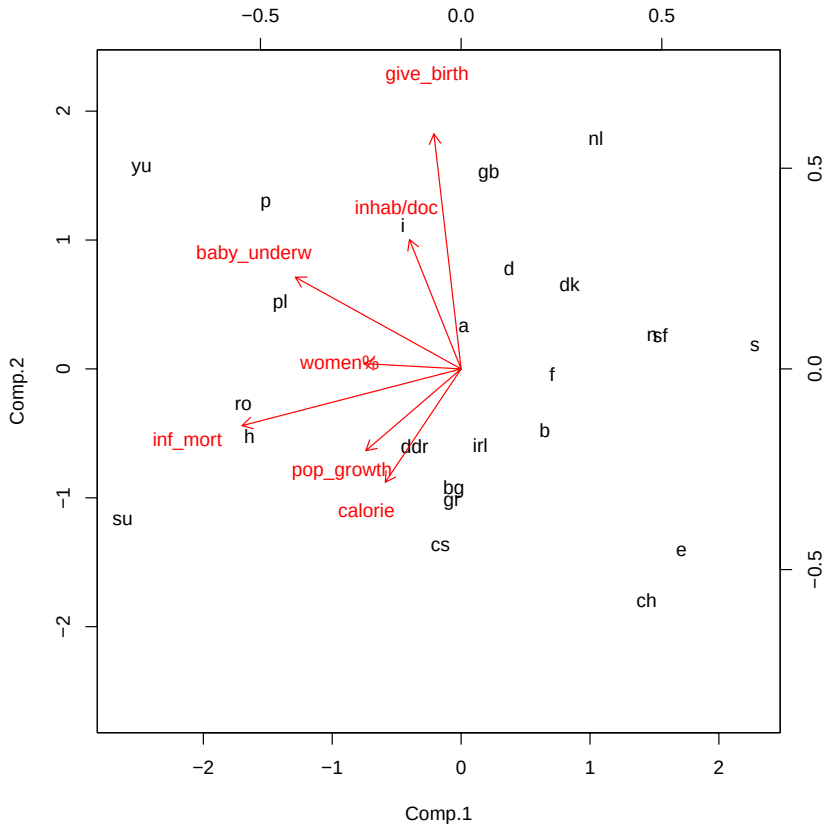


Figure 3.15: Biplot of PLS1 for **X**–data of euro86 without Albania (al)

numbers in these variables (b, ch, e, irl). France (f) seems to be neutral with regard to the second component. The biplot for the **y**–data (Figure 3.16) naturally shows dependencies of both the first and the second component on the variable female life expectation. The relation between the variable and the component seems to be equally strong in both cases. Countries with high female life expectation are i, gr, f, nl, ch, n and s whereas those with low female life expectation are mainly the eastern countries such as ro, su, cs, bg, yu and h, but also irl.

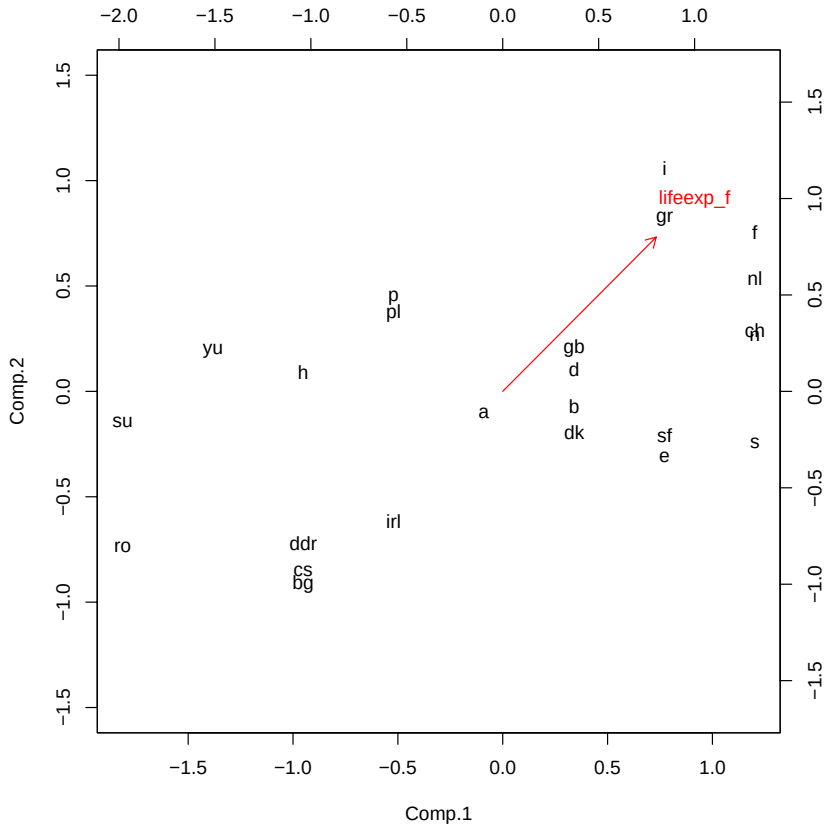


Figure 3.16: Biplot of PLS1 for **y**–data of euro86 without Albania (al)

The screeplots for the **X**– and **y**–data (Figures 3.17 and 3.18) again show the proportion of the total variation explained by the calculated components. In the case of the **X**–data the first and second component explain 35% of the total variation. 3 components would already explain 54% and if we took 6 components into account we would fulfill the rule of thumb and get 90%. The corresponding percentage of explanation of the total variance of the **y**–part is 40% for 2, 52% for 3 and 88% for 6 components.

Figures 3.19 and 3.20 graphically represent the combination of **X**– and **y**–data.

The plot for the first PLS components (Figure 3.19) makes a clear distinction between countries with high infant mortality, baby underweight and population growth together with low female life expectation and those clearly opposed. This is nearly a division into

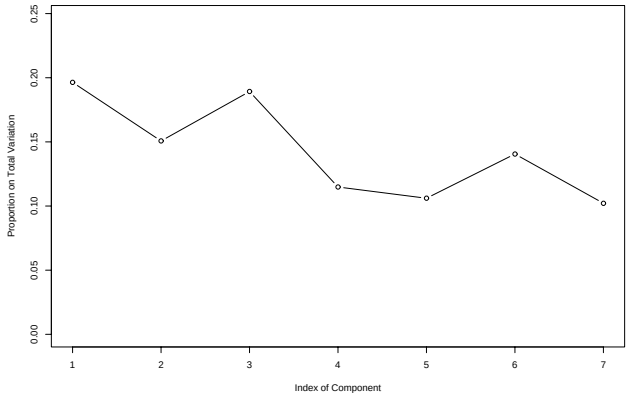


Figure 3.17: Screeplot of PLS1 components for **X**–data of euro86 without Albania (al)

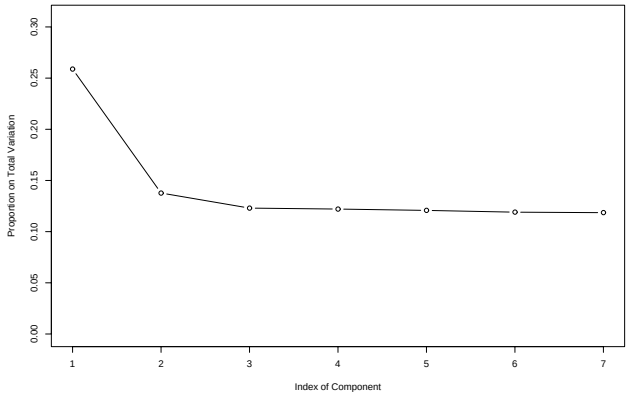


Figure 3.18: Screeplot of PLS1 components for **y**–data of euro86 without Albania (al)

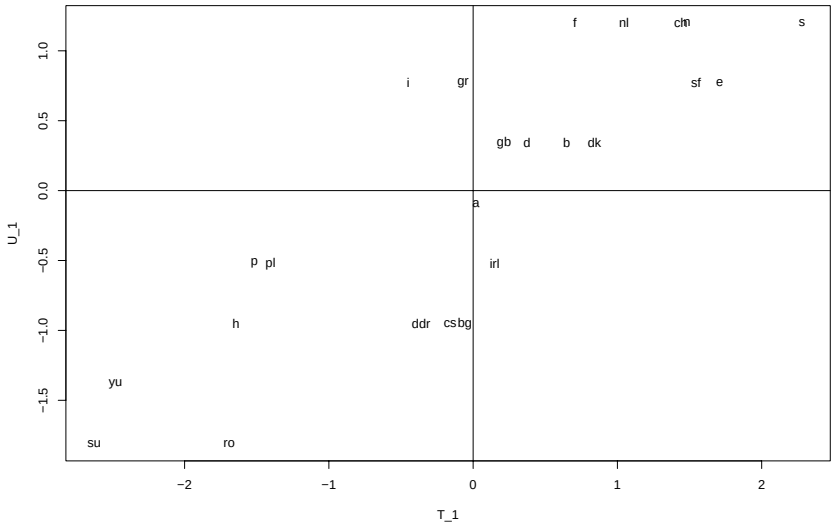


Figure 3.19: First PLS1 components for euro86 without Albania (al)

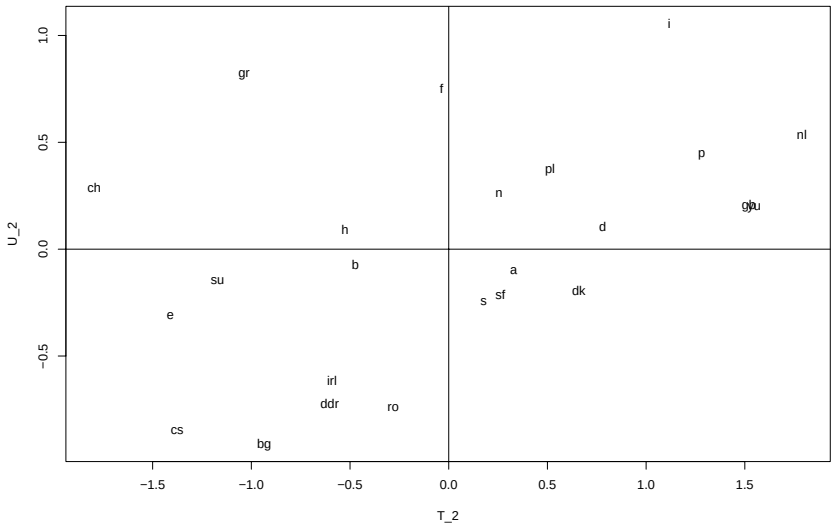


Figure 3.20: Second PLS1 components for euro86 without Albania (al)

eastern (su, ro, yu, h, pl, ddr, cs and bg) and western countries (f, nl, ch, s, sf, e, gb, d, b, dk, i and gr). Only the western countries Austria (a), Ireland (irl) and Portugal (p) show tendencies to higher infant mortality, baby underweight, population growth and low female life expectation.

In the first quadrant of the plot of the second components (Figure 3.20) we find countries with a high percentage of women in the age to give birth and high female life expectation such as i, nl, p, pl, gb, yu, n and d. The second quadrant shows those countries where a low percentage of women in the age to give birth dominates (ch, gr, f and h). In the third quadrant we find countries with higher infant mortality, population growth and portion of calories per day together with a low female life expectation: cs, bg, ddr, ro, irl, e, su, b. Finally in the fourth quadrant there are countries with slightly higher numbers of women in the age to give birth and inhabitants per doctor behaving neutral regarding female life expectation, namely, a, sf, s and dk.

Chapter 4

Robust Partial Least Squares

Generally, the aim of robust statistics is to handle data as in the classical way but additionally to identify and treat the appearance of outliers. Outliers can occur everywhere and at any time by various reasons.

Thinking of chemometrics, simple examples of outliers would be measurement errors, produced by a room temperature too high for the instrument or different calibrations of the recording instrument when analysing the same substances. Looking at the PLS literature there are some attempts for robustification of the PLS method which will be described in the sections below.

Still there are also critical voices from Martens and Naes (1989): They plead for not using robust methods as they fear a loss of information from simple deletion of the outlying measurements. They prefer any program to give a warning if an outlier occurs so that the user is informed and may handle the data analysis differently, adjusted to the new situation.

At this point we should stress that good robust statistical methods do not delete or ignore outliers, but treat them separately in the analysis. After the robust fit of the model outliers can be detected by their large residuals.

The problem with robustifying PLS is how to keep the orthogonality of the scores $\mathbf{T}$.
There are two main approaches to the robustification of PLS. The first idea is to use the Iteratively Reweighted Least Squares method (IRLS). In this method each data point is assigned with a certain weight (between 0 and 1), depending on its distance from the "middle" of the data set relative to the spread of the data. Measurements which are far away from the "middle" of the data set gain less weight than those in the "middle" of it. There are different weight functions, based on the median and MAD (median of absolute deviations from the median), for example. This IRLS principle can be applied to each regression of the PLS algorithm in order to make every regression solution robust. Still it is an interesting question if it is necessary to calculate each regression by means of IRLS or if it is sufficient to apply IRLS to a chosen few. Wakeling and Macfie (1992) and Cummins and Andrews (1995) introduce two similar methods of robustification of PLS with the help of IRLS. Griep et al. (1995) compare the method presented in Wakeling and Macfie (1992) to two other methods for robust PLS by robustification of single regressions within the PLS algorithm.

The second approach can be found in Gil and Romera (1998). Instead of applying a robust regression method the regression coefficients are estimated with the help of a robust covariance matrix.

In literature algorithms where only a part of the equations is robustified are called *semi-robust* methods.

Now we will present these methods in detail.

4.1 Robustification by Iterative Reweighting

First we want to shortly introduce the reweighted least squares method (cf. Rousseeuw and Leroy, 1987) and then–as an extension–the IRLS (Iteratively Reweighted Least Squares) algorithm as lined out by Phillips and Eyring (1983).

Let us consider a simple linear regression equation without intercept of the form

$$\mathbf{y} = \mathbf{x}\beta + \boldsymbol{\varepsilon}$$

where both $\mathbf{x}$ and $\mathbf{y}$ are vectors and the scalar β is the regression coefficient which should be estimated. The usual least squares solution for β would be

$$\widehat{\beta} = \frac{\mathbf{x}^\top \mathbf{y}}{\mathbf{x}^\top \mathbf{x}}.$$

In reweighted least squares this solution is used to build a weight function which leads to a robust regression solution.

Calculate the residual vector

$$\mathbf{r} = \mathbf{y} - \mathbf{x}\widehat{\beta}.$$

With the help of

$$S = \text{median}(|r_i|) \quad , i = 1, \ldots, n$$

we get the median standardized residuals

$$\tilde{\mathbf{r}} = \frac{\mathbf{r}}{S}. \tag{4.1}$$

With these residuals from Equation (4.1) a weight matrix $\boldsymbol{\Omega} = diag(w_{11}, \ldots, w_{nn})$ is calculated. A weight function often used is the biweight function defined by

$$w_{ii} = \left\{ \begin{array}{ll} \left[1 - (\tilde{r}_i/c)^2\right]^2 & \text{for } |\tilde{r}_i| \leq c \\ 0 & \text{for } |\tilde{r}_i| > c \end{array} \right\} \tag{4.2}$$

where c is a sensitivity factor which is chosen according to each problem.

These weights are assigned to the data giving

$$\begin{aligned} \tilde{\mathbf{y}} &= \boldsymbol{\Omega}^{1/2}\mathbf{y} \\ \tilde{\mathbf{x}} &= \boldsymbol{\Omega}^{1/2}\mathbf{x} \end{aligned}$$

where $\mathbf{\Omega}^{1/2}$ stands for the elementwise square-roots of $\mathbf{\Omega}$. Hence we get a new model for the weighted regression equation

$$\tilde{\mathbf{y}} = \tilde{\mathbf{x}}\tilde{\beta} + \tilde{\boldsymbol{\varepsilon}}$$

equal to

$$\mathbf{\Omega}^{1/2}\mathbf{y} = (\mathbf{\Omega}^{1/2}\mathbf{x})\tilde{\beta} + \tilde{\boldsymbol{\varepsilon}}$$

with the resulting estimation of the regression coefficient

$$\widehat{\tilde{\beta}} = \frac{\tilde{\mathbf{x}}^\top\tilde{\mathbf{y}}}{\tilde{\mathbf{x}}^\top\tilde{\mathbf{x}}} = \frac{\mathbf{x}^\top\mathbf{\Omega}\mathbf{y}}{\mathbf{x}^\top\mathbf{\Omega}\mathbf{x}}.$$

This reweighted least squares method is the basis for the iteratively reweighted least squares (IRLS) method. In IRLS successive iterations are wrapped around this reweighted least squares method. The newly gained $\widehat{\tilde{\beta}}$ is used as $\widehat{\beta}$ in the calculation of the residual vector $\mathbf{r}$ to perform the next step. This procedure stops when $\widehat{\beta}$ "converges", i.e., the norm of the difference between $\widehat{\beta}$ and the newly calculated $\widehat{\tilde{\beta}}$ is smaller than a certain constant.

4.1.1 RPLS–Robust Partial Least Squares

The first to suggest a robust PLS procedure based on the IRLS approach were Wakeling and Macfie (1992). In their paper they propose to use two weight matrices, one for the data matrix $\mathbf{X}$, the other for $\mathbf{Y}$. Within the iteration there has to be a weight matrix for every column of $\mathbf{X}$, i.e., p matrices, and also for every column of $\mathbf{Y}$, i.e., q matrices. Additionally these matrices have to be reweighted in each iteration.

Wakeling and Macfie (1992) called their method *RPLS (Robust Partial Least Squares)*. We will now present RPLS in detail.

For the robustification of the PLS2 algorithm two matrices $\mathbf{M}_{h-1} = [\mathbf{m}_{1(h-1)}, \ldots, \mathbf{m}_{p(h-1)}]$ and $\mathbf{N}_{h-1} = [\mathbf{n}_{1(h-1)}, \ldots, \mathbf{n}_{q(h-1)}]$ corresponding to the dimensions of $\mathbf{E}_{h-1}$ and $\mathbf{F}_{h-1}$ are introduced. As starting values all elements are set to unity. These two matrices form weight vectors for each variable.

RPLS now has the following iterative parts.

Form $\mathbf{\Phi}^*_{i(h-1)} = diag(m_{1i(h-1)}, m_{2i(h-1)}, \ldots, m_{ni(h-1)}), i = 1, \ldots, p$, the diagonal matrix of starting weights for the reweighted regression of $\mathbf{e}_{i(h-1)}$ (which is the ith column of $\mathbf{E}_{h-1}$) on $\mathbf{u}_h$. Apply the iteratively reweighted regression algorithm leading to the updated sample weights $\mathbf{\Phi}_{i(h-1)}$. Now replace the solution of Step 1 in the classical PLS2 algorithm with the following robust solution for $\mathbf{w}_h = (w_{1(h)}, \ldots, w_{p(h)})$:

$$w_{i(h)} = \frac{\mathbf{e}^\top_{i(h-1)}\mathbf{\Phi}_{i(h-1)}\mathbf{u}_h}{\mathbf{u}_h^\top\mathbf{\Phi}_{i(h-1)}\mathbf{u}_h}$$

and replace the matrix $\mathbf{M}_{h-1}$ by

$$\mathbf{m}_{i(h-1)} = \mathbf{\Phi}^{1/2}_{i(h-1)}\mathbf{1}$$

where $\mathbf{1}$ is a vector of length n with all elements set to unity.

Normalize $\mathbf{w}_h$ as in Step 2 of the classical PLS2 algorithm.

In Step 3 substitute $\mathbf{E}_{h-1}$ with $\mathbf{E}_{h-1} \bigotimes \mathbf{M}_{h-1}$ where $\bigotimes$ represents the elementwise multiplication of two matrices of equal size and perform the regression giving the following least squares solution:

$$\mathbf{t}_h = (\mathbf{E}_{h-1} \bigotimes \mathbf{M}_{h-1})\mathbf{w}_h.$$

In analogy to Step 1 change Step 4 in the following way.

Form $\mathbf{\Gamma}^*_{j(h-1)} = diag(n_{1j(h-1)}, n_{2j(h-1)}, \ldots, n_{nj(h-1)}), j = 1, \ldots, q$, the diagonal matrix of starting weights for the robust regression of $\mathbf{f}_{j(h-1)}$ (which is the jth columnof $\mathbf{F}_{h-1}$) on $\mathbf{t}_h$. Apply the iteratively reweighted regression algorithm leading to the updated sample weights $\mathbf{\Gamma}_{j(h-1)}$. Now replace the solution of Step 4 in the classical PLS2 algorithm with the following robust solution for $\mathbf{q}_h = (q_{1(h)}, \ldots, q_{q(h)})$:

$$q_{j(h)} = \frac{\mathbf{f}^\top_{j(h-1)} \mathbf{\Gamma}_{j(h-1)} \mathbf{t}_h}{\mathbf{t}^\top_h \mathbf{\Gamma}_{j(h-1)} \mathbf{t}_h}$$

and replace the matrix $\mathbf{N}_{h-1}$ by

$$\mathbf{n}_{j(h-1)} = \mathbf{\Gamma}^{1/2}_{j(h-1)} \mathbf{1}$$

where $\mathbf{1}$ is a vector of length n with all elements set to unity.

In Step 5 normalize $\mathbf{q}_h$ as in the classical PLS2 algorithm.

In Step 6 substitute $\mathbf{F}_{h-1}$ with $\mathbf{F}_{h-1} \bigotimes \mathbf{N}_{h-1}$ and again perform the regression yielding the least squares solution:

$$\mathbf{u}_h = (\mathbf{F}_{h-1} \bigotimes \mathbf{N}_{h-1})\mathbf{q}_h.$$

The convergence check of Step 7 is performed in exactly the same way as in the classical PLS2 algorithm. Wakeling and Macfie (1992) propose to calculate the final values for $\mathbf{t}_h$ and $\mathbf{u}_h$ from the unweighted data by calculating

$$\begin{aligned} \mathbf{t}_h &= \mathbf{E}_{h-1}\mathbf{w}_h \\ \mathbf{u}_h &= \mathbf{F}_{h-1}\mathbf{q}_h. \end{aligned}$$

They don't make any comment about the calculation of $\mathbf{p}_h$ in Step 8 and of b_h in Step 11 so we suppose they leave Steps 8 and 11 unchanged.

The data matrices are updated in the same way as in the classical PLS2 algorithm (Step 12).

A point of criticism, already stated by Wakeling and Macfie (1992) themselves, is the large amount of computation time necessary to gain new weight matrices in every iteration.

Another crucial fact is that IRLS is *no* robust regression estimation as the weights, which form the main part of IRLS, are deduced with the help of ordinary least squares so that they may be influenced by outliers, especially by bad leverage points, i.e., outliers in x–direction.

4.1.2 IRPLS

Cummins and Andrews (1995) present a slightly different version of a robust PLS procedure, also based on the ideas of IRLS. This method has strong similarities to the RPLS method proposed by Wakeling and Macfie (1992) which are definitely lined out in the paper of Cummins and Andrews (1995).

The first difference to the method of Wakeling and Macfie (1992) is the usage of a slightly different weight function. Instead of calculating the median absolute deviation of the residuals from 0, namely

$$S = \text{median}(|r_i|) \quad , i = 1, \ldots, n,$$

they use the MAD (Median Absolute Deviation), i.e., the median absolute deviation of the residuals from their median:

$$\tilde{S} = \text{MAD}(r_i) \quad , i = 1, \ldots, n.$$

Additionally they suggest to adjust $\tilde{S}$ by the multiplicative factor 1.4826 to make it consistent for the standard deviation at the Gaussian model. They found convergence of the adjusted MAD to the standard deviation in a performance study carried out on random normal samples of size 50, 500 and 5000, whereas the unadjusted MAD converges very slowly.

They also suggest the usage of other weight functions than the biweight function but always replace S by $\tilde{S}$. For all these weight functions they make proposals of default values for the sensitivity factor c. For further information on the choice of the weight function and their corresponding sensitivity factor confer Cummins and Andrews (1995).

For the algorithmic part Cummins and Andrews (1995) use the *Tripos* software product *SYBYL* together with the macro package of the software, called *SPL (SYBYL programming language)*. In this software package a PLS algorithm is already implemented, which is also able to perform *weighted PLS*, but we found no further information on the details of this implemented algorithm.

Cummins and Andrews (1995) now suggest the following robustified version of the PLS2 algorithm, called IRPLS (Iteratively Reweighted Partial Least Squares), based on the implementation of PLS in *SYBYL*:

Step 0: Choose a weight function.
Step 1: Perform an ordinary PLS regression analysis.
Step 2: Pass the regression residuals (**Y**–block errors) from Step 1 into the weight function.
Step 3: Perform a weighted PLS regression analysis using the weights just obtained.
Step 4: Pass the residuals from Step 3 into the weight function choosen.
Step 5: If a convergence criterion is met, stop; else go to Step 3.

Another change is done by Cummins and Andrews (1995): They pass the predicted residuals instead of the ordinary ones into the weight funcion. They found their algorithm to give better results by this change.

Gil and Romera (1998), who present a summary of the existing robust PLS procedures aside introducing a new one themselves, make a classification of the two methods presented yet by the application of the weight function:

They call the RPLS method of Wakeling and Macfie (1992) a method of Internal Iterative Reweighting, also called PLSIR by Gil and Romera (1998), for only applying the reweighted regression model within each iteration.

The proposal of IRPLS by Cummins and Andrews (1995) is classified as External Iterative Reweighted, called IRPLS by Gil and Romera (1998), because of the calculation of weights after performing a PLS regression each time and their usage for the next PLS regression.

4.2 Robust Covariance Matrix Estimation

The idea of robust covariance estimation instead of using iteratively reweighted least squares (IRLS) is introduced in Gil and Romera (1998).

They only present their method for the PLS1 algorithm where–as we have shown in Section 3.2–many simplifications come to pass. It would be interesting if and how an extension of the robust covariance matrix to the PLS2 algorithm is possible.

They propose three variations of their main idea: PLSR, PLSR2 and PLSMR. They call PLSR and PLSR2 partial robustification methods whereas PLSMR is a global robustification approach.

Gil and Romera (1998) start with the assumption that the combined data $[\mathbf{f}_0, \mathbf{E}_0]$ come from a joint distibution with mean zero and a population covariance matrix

$$\boldsymbol{\Sigma}_{[\mathbf{f}_0,\mathbf{E}_0]} = \begin{pmatrix} \sigma^2_{\mathbf{f}_0} & \boldsymbol{\delta}^\top_{\mathbf{f}_0,\mathbf{E}_0} \\ \boldsymbol{\delta}_{\mathbf{f}_0,\mathbf{E}_0} & \boldsymbol{\Sigma}_{\mathbf{E}_0} \end{pmatrix}.$$

According to Helland (1990) the population loading vectors $\mathbf{w}_h$ are proportional to

$$\boldsymbol{\delta}_{\mathbf{f}_0,\mathbf{E}_0} - \boldsymbol{\Sigma}_{\mathbf{E}_0}\mathbf{W}_{h-1}(\mathbf{W}^\top_{h-1}\boldsymbol{\Sigma}_{\mathbf{E}_0}\mathbf{W}_{h-1})^{-1}\mathbf{W}^\top_{h-1}\boldsymbol{\delta}_{\mathbf{f}_0,\mathbf{E}_0}$$

with $\mathbf{W}_{h-1} = [\mathbf{w}_1, \ldots, \mathbf{w}_{h-1}]$ and a starting value of $\mathbf{w}_1$ proportional to $\boldsymbol{\delta}_{\mathbf{f}_0,\mathbf{E}_0}$. With this defintion every $\mathbf{w}_h$ only depends on the previous $\mathbf{w}_1, \ldots, \mathbf{w}_{h-1}$, $\boldsymbol{\delta}_{\mathbf{f}_0,\mathbf{E}_0}$ and $\boldsymbol{\Sigma}_{\mathbf{E}_0}$. They consider these relations an alternative definition of the PLS1 algorithm.

PLSR

The basic idea in PLSR is that the data $[\mathbf{f}_0, \mathbf{E}_0]$ have a sample covariance matrix $\mathbf{V}$ of dimension $(p+1) \times (p+1)$:

$$\mathbf{V} = \widehat{\boldsymbol{\Sigma}}_{[\mathbf{f}_0,\mathbf{E}_0]} = \begin{pmatrix} \widehat{\sigma}^2_{\mathbf{f}_0} & \widehat{\boldsymbol{\delta}}^\top_{\mathbf{f}_0,\mathbf{E}_0} \\ \widehat{\boldsymbol{\delta}}_{\mathbf{f}_0,\mathbf{E}_0} & \widehat{\boldsymbol{\Sigma}}_{\mathbf{E}_0} \end{pmatrix}.$$

The idea is to calculate a robust version of the matrix $\mathbf{V}$ and then take the part corresponding to the mixed covariances, after normalization, as $\mathbf{w}_1$. The values $\mathbf{w}_h$ in the following iterations are calculated analogous with replacement of $\mathbf{E}_0$ and $\mathbf{f}_0$ by $\mathbf{E}_{h-1}$ and $\mathbf{f}_{h-1}$, respectively.

This algorithm has the disadvantage that only $\mathbf{w}_1$ is estimated robustly as the succeeding values $\mathbf{w}_h, h = 2, \ldots, k$, are calculated from the residuals $\mathbf{E}_{h-1}$ and $\mathbf{f}_{h-1}$ which are gained by ordinary least squares regressions.

PLSR2

In contrast to PLSR Gil and Romera (1998) use a vector in PLSR2 built from robust estimations $\mathbf{V}_i$ of the covariance matrices of $\mathbf{f}_0$ and each $\mathbf{e}_{(0)i}, i = 1, \ldots, p$, with $\mathbf{E}_0 = [\mathbf{e}_{(0)i}]$:

$$\mathbf{V}_i = \widehat{\mathbf{\Sigma}}_{[\mathbf{f}_0, \mathbf{e}_{(0)i}]} = \begin{pmatrix} \widehat{\sigma}^2_{\mathbf{f}_0} & \widehat{\delta}_{\mathbf{f}_0, \mathbf{e}_{(0)i}} \\ \widehat{\delta}_{\mathbf{f}_0, \mathbf{e}_{(0)i}} & \widehat{\sigma}_{\mathbf{e}_{(0)i}} \end{pmatrix}.$$

The vector $\mathbf{w}_1$ then is equal to the normalization of the vector $(\widehat{\delta}_{\mathbf{f}_0, \mathbf{e}_{(0)i}}, \ldots, \widehat{\delta}_{\mathbf{f}_0, \mathbf{e}_{(0)i}})$. For the following iterations $\mathbf{w}_h$ is obtained by replacing $\mathbf{f}_0$ and $\mathbf{E}_0$ by $\mathbf{f}_{h-1}$ and $\mathbf{E}_{h-1}$, respectively.

In this method important characteristics such as affine equivariance for linear transformations and efficiency, among others, are lost.

PLSMR

In this modfication (MR stands for matrix robust) Gil and Romera (1998) use the robust estimate from the covariance matrix in PLSR (see Equation (4.2)). As in PLSR they choose $\mathbf{w}_1$ as the normalization of the vector $\widehat{\boldsymbol{\delta}}_{[\mathbf{f}_0, \mathbf{E}_0]}$ but then they calculate the succeeding values $\mathbf{w}_h$ robustly, namely proportional to

$$\widehat{\boldsymbol{\delta}}_{[\mathbf{f}_0, \mathbf{E}_0]} - \widehat{\mathbf{\Sigma}}_{\mathbf{E}_0} \mathbf{W}_{h-1} (\mathbf{W}_{h-1}^\top \widehat{\mathbf{\Sigma}}_{\mathbf{E}_0} \mathbf{W}_{h-1})^{-1} \mathbf{W}_{h-1}^\top \widehat{\boldsymbol{\delta}}_{[\mathbf{f}_0, \mathbf{E}_0]}.$$

In PLSMR every $\mathbf{w}_h, h = 1, \ldots, k$, now really is a robust estimate but still this method is only semi-robust as all other estimations are carried out with ordinary least squares.

As methods for the robustification of covariance matrices they propose the Stahel–Donoho estimator (cf. Stahel, 1981; Donoho, 1982) and the minimum volume ellipsoid estimator (MVE) of Rousseeuw (1985). Based on surveys of Maronna and Yohai (1995) they prefer the Stahel–Donoho estimator.

According to simulation studies Gil and Romera (1998) think PLSMR to behave best compared to classical PLS, PLSR, PLSR2, IRPLS and PLSIR.

4.3 Other Proposals

Griep et al. (1995) introduce two other methods for the robustification of PLS. These methods are only lined out for PLS1 and an enhancement to PLS2 seems to be difficult.

They start from the simplified version of PLS1 as introduced in Section 3.2, without looking for orthogonal scores $\mathbf{T}$. Their idea is to perform a robust regression in Step 1

$$\mathbf{w}_h^\top = \frac{\mathbf{f}_{h-1}^\top \mathbf{E}_{h-1}}{\mathbf{f}_{h-1}^\top \mathbf{f}_{h-1}}.$$

They suggest to apply the robust estimators LMS (cf. Rousseeuw, 1984) for an algorihtmic implementation, and Siegel's Repeated Median (Siegel, 1982).

Let us consider a linear regression equation of the form

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

where both $\mathbf{X}$ is a matrix of dimension $n \times p$, $\mathbf{y}$ is a n-dimensional vector and the p-dimensional vector $\boldsymbol{\beta}$ contains the regression coefficients which should be estimated.

The LMS (Least Median of Squares) estimator is similar to the classical LS (Least Sum of Squares) estimator with the important feature that the sum is replaced by the median. The residuals between true and estimated y–values are given by

$$\mathbf{r}(\boldsymbol{\beta}) = \mathbf{y} - \hat{\mathbf{y}} = \mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}.$$

Then the LMS estimator is defined as

$$\widehat{\boldsymbol{\beta}} = \text{argmin}_{\boldsymbol{\beta}} \text{median}_i [r_i(\boldsymbol{\beta})]^2.$$

Unfortunately, the LMS estimator has disadvantages concerning asymptotic efficiency but for a detailed study of the properties of the LMS estimator we refer to Rousseeuw and Leroy (1987).

For Siegel's Repeated Median we consider a subset of p observations

$$(\mathbf{x}_{i_1}, y_{i_1}), \ldots, (\mathbf{x}_{i_p}, y_{i_p}).$$

First the parameter vector fitting these points exactly is computed. The jth coordinate of this parameter vector is denoted by $\beta_j(i_1, \ldots, i_p)$. The coordinatewise definition of the repeated median regression estimator now is

$$\widehat{\beta}_j = \text{median}_{i_1}(\ldots(\text{median}_{i_{(p-1)}}(\text{median}_{i_p}\beta_j(i_1, \ldots, i_p)))\ldots).$$

The disadvantage of this estimator is the large amount of computation time as all subsets of p observations have to be computed.

Griep et al. (1995) compare these two methods to the IRLS method introduced by Wakeling and Macfie (1992) and find IRLS still to perform best. This may result on one hand from

the properties of the robust regression estimators used and on the other hand from the application of these robust regression estimates to only the first regression equation in PLS1. For further surveys we suggest to study the behaviour of other robust regression estimates (cf., e.g., Rousseeuw and Leroy, 1987) and the differences occuring when applying robust regression estimates to all regression equations. An example for another choice of a robust regression estimator would be the LTS estimator (Rousseeuw, 1984) who has higher efficiency and a fast algorithm (Rousseeuw and Van Driessen, 1998).

Chapter 5

Outlook

In this work we presented the classical NIPALS and PLS algorithm and attempts for robustification of PLS. Future work will focus on further development and analysis of robust PLS algorithms.

As it seems to us there will be two main approaches (as could be seen in Chapter 4): Robustification of the different steps of the PLS algorithm or robust estimation of the covariance matrix describing the PLS model.

In this context a more detailed survey of the mathematical properties and their importance for applications will be necessary as these properties will have to be maintained in robustified versions of the algorithm. Also modified versions of the PLS algorithm may be interesting in this context, e.g., the SIMPLS method proposed by De Jong (1993).

An interesting question concerning the robustification based on the different steps of the PLS algorithm still is the performance of semi-robust methods compared to that of completely robust methods, i.e., wether it makes sense to robustify each regression solution or the robustification of some of them is sufficient as suggested by Wakeling and Macfie (1992).

It will also be important for practical applications to focus on the robustification of the PLS2 algorithm instead of PLS1.

Additionally to the introduction of NIPALS, PLS and the attempts of robustification presented in the chapters above we also made a short internet recherche concerning PLS. Apart from links about literature we also found links to statistical software packages where PLS is already implemented. A short summary of the most interesting links is given below.

http://disc-nt.cba.uh.edu/chin/PLSINTRO.HTM: Introduction to PLS by Wynne W. Chin with many citations of PLS literature, contains link to homepage of Dr. Jack McArdle.

http://kiptron.psyc.virginia.edu/disclaimer.html: Homepage of Dr. Jack McArdle, possibility to download *PLS-PC 1.8 for DOS* developed by J.-B. Lohmöller.

http://www.statsoftinc.com: Aside the base package of *STATISTICA* there is an add-on called *STATISTICA Advanced Linear/Non-Linear Models* including a whole part only on PLS.

http://www.statsoftinc.com/textbook/stpls.html: Description of NIPALS, SIMPLS (cf. De Jong, 1993), PLS and cross-validation.

http://www.tripos.com/software/sybyl.html: Software *SYBYL* from *TRIPOS* as used in Cummins and Andrews (1995).

http://www.gsm.uci.edu/~joelwest/SEM/PLS.html: List of links with short descriptions of books, articles, software and application.

http://www.sas.com: PLS procedure in *SAS/STAT* software, .pdf document with examples of PLS.

http://www.galactic.com/products/: Chemometrics software, especially spectroscopy, *GRAMS* with add-on package *PLSplus/IQ*, containing PLS1 and PLS2.

http://www.eigenvektor.com/PLS_Toolbox.html: Add-on package *PLS_Toolbox 2.1* for *MATLAB*, also containing PCA, PCR (principal components regression), Ridge Regression, Continuum Regression and nonlinear PLS Regression.

http://www.gsu.edu/~mkteer/relmeth.html: Page for Structural Equation Modelling (SEM), PLS is mentioned as related method to SEM.

http://cran.r-project.org: A PLS package for *R* is under development.

This short list–aside the many articles that can be found in the bibliography–shows how much work has already been done in the field of PLS but also gives a slight impression of what could still be done in future works.

Appendix A

Data

A.1 The districts Data Set

The districts data set is taken from a survey carried out at the Institute of Statistics and Probability Theory at the Vienna University of Technology in 1996 (cf. Rauth and Sedlacek, 1996). The aim of this project was to find the (economic) characteristics of the Austrian regions, similar to those leading to grants from the European Union. By doing so a data set emerged consisting of 17 variables and 99 observations.

The variables describe such characteristics as percentage of children, adults and seniors, employees in industry, trade, tourism, service and agriculture, unemployed, level of education (university, secondary school, primary school), (im-)migration, not daily commuters, commuters to another district, mountain farmers and tourist stays. A value greater than 1 in variable (im-)migration marks a region with immigration whereas a value less than 1 shows migration. The variable names used as abbreviations in graphical representations are summarized in Table A.1.

The observations are the corresponding numbers from the Austrian political districts. These political districts include big cities (Vienna, Graz, Klagenfurt, Salzburg, Linz, Eisenstadt, Innsbruck), regions of high industrial development (Gmunden, Hallein, Dornbirn, Bregenz, ...) as well as touristic regions (Rust, Kitzbühel, Reutte, ...) and agricultural regions (Hollabrunn, Tulln, Zwettl, Hartberg, ...). A complete listing of all political districts of Austria is given in Table A.2.

A.2 The euro86 Data Set

The euro86 data set was collected from statistical year books giving an overview of the populational characteristics of the European countries around 1986.

For the 25 European countries (the observations) 9 variables were collected. In detail they are the average growth of population from 1986 to 2000, the percentage of women in the age to

give birth (1985), percentage of women per 100 men (1985), life expectation of women (1986), life expectation of men (1986), infant mortality (1986), inhabitants per doctor (1981), daily provided calories per person (1985) and percentage of infants born underweighted (1984). The variable names used in the various plots are reprinted in Table A.3. A list of the abbreviations of the European countries (observations) is given in Table A.4.

One of the European countries, namely Albania, can easily be detected as an outlier. Compared with the median of the European values it has a 9 times higher population growth, 4 times the infant mortality, 4.5 times the number of inhabitants per doctor and 20 percent less calories per person provided daily.

A.3 The oeamtc Data Set

The oeamtc data set was collected from the journal of the Austrian Automobile Association (abbreviation: OEAMTC) describing 18 cars consuming diesel.

As a help for purchasing decisions 21 characteristics were provided: price (in Euro), tax (which in Austria depends on the performance; in Euro), the costs of a standard liability insurance (in Euro) and a full comprehensive insurance (in Euro), capacity (in ccm), performance (in hp) and torque (in rpm) of the motor, length (in mm), width (in mm), height (in mm), weight of the empty car (in kg), highest possible total weight (in kg), minimum size of the boot (in l), maximum size of the boot (in l), highest speed (in km/h), acceleration (in sec), elasticity (in sec), consumption cross-country (in l), consumption in town (in l), consumption total (in l) and test consumption (in l). The abbreviations for the variable names are given in Table A.5.

Different types of cars were under survey, namely from micro to maxi over compact, also including limousines and vans. However, there are no SUVs (sports utility vehicle), sports or cross-country cars. So a wide range of cars is examined, sometimes with only one respresentative, instead of one special class of cars. The exact names of all cars can be found in Table A.6.

Still some cars appear as outliers as their characteristics in at least one variable is very different from the rest of the examined cars, e.g., the only van in this data set (Chrysler Voyager) has a maximal boot volume of 4700l whereas the median is 1252.5l. Another example would be the smallest car (Seat Arosa) with respect to the minimal boot volume or the cross-country consumption.

This data set is useable for regression (and therefore for PLS) if we divide it into a $\mathbf{Y}$–data matrix consisting of the variables price, tax, liability insurance and full comprehensive insurance and a $\mathbf{X}$–data matrix consisting of the rest of the variables. The underlying thought of this partition is that the features of the car sum up its price, whereas taxes and the insurances rely on the performance and the price, also.

Table A.1: Variables of districts Data Set

Symbol	Explanation
Children	percentage of children (beyond 15 years) in the resident population
Adult+Youth	percentage of youth and adult people (15 to 60 years) in the resident population
Old	percentage of older and old people (over 60 years) in the resident population
Industry	portion of employees in the manufacturing industry, in the industry, and in the building trade, in percent of the total employees
Trade	portion of employees in the trade in percent of the total employees
Tourism	portion of employees in the lodging and catering trade in percent of the total employees
Service	portion of employees in service in percent of the total employees (this measurement includes credit and insurancy trade, personal, social, and public service, traffic and news service)
Agriculture	portion of employees in agriculture and forest in percent of the total employees
Unemployed	portion of unemployed people in percent of the resident population
University	portion of people with university education in percent of the resident population
Sec.School	portion of people with secondary school in percent of the resident population
Prim.School	portion of people with primary school, technical college, or apprenticeship in percent of the resident population
Im/migration	number of total employees in workshop places divided by the number of the total employees of the resident population (This number should express the economic situation of the district, and it shows if there is immigration or migration.)
Comm.n.daily	portion of commuters commuting not daily, in percent of the total employees of the resident population
Comm.distr.	portion of commuters commuting to another district, in percent of the total employees of the resident population
Mountain-farm	portion of mountain farms in percent of all farms
Tourist-stays	number of overnight stays per year in the tourism

Table A.2: Observations of districts Data Set

AM	Amstetten	LL	Linz (Land)
B	Bregenz	LZ	Lienz
BL	Bruck an der Leitha	MA	Mattersburg
BM	Bruck an der Mur	MD	Mödling
BN	Baden	ME	Melk
BR	Braunau am Inn	MI	Mistelbach
BZ	Bludenz	MU	Murau
DL	Deutschlandsberg	MZ	Mürzzuschlag
DO	Dornbirn	NK	Neunkirchen
E	Eisenstadt (Stadt)	NS	Neusiedl am See
EF	Eferding	OP	Oberpullendorf
EU	Eisenstadt-Umgebung	OW	Oberwart
FB	Feldbach	P	Sankt Pölten (Stadt)
FE	Feldkirchen	PE	Perg
FF	Fürstenfeld	PL	Sankt Pölten (Land)
FK	Feldkirch	RA	Radkersburg
FR	Freistadt	RE	Reutte
G	Graz (Stadt)	RI	Ried im Innkreis
GD	Gmünd	RO	Rohrbach
GF	Gänserndorf	RT	Rust (Stadt)
GM	Gmunden	S	Salzburg (Stadt)
GR	Grieskirchen	SB	Scheibbs
GS	Güssing	SD	Schärding
GU	Graz-Umgebung	SE	Steyr (Land)
HA	Hallein	SL	Salzburg-Umgebung
HB	Hartberg	SP	Spittal an der Drau
HE	Hermagor	SR	Steyr (Stadt)
HL	Hollabrunn	SV	Sankt Veit an der Glan
HO	Horn	SZ	Schwaz
I	Innsbruck (Stadt)	TA	Tamsweg
IL	Innsbruck (Land)	TU	Tulln
IM	Imst	UU	Urfahr-Umgebung
JE	Jennersdorf	VB	Vöcklabruck
JO	Sankt Johann im Pongau	VI	Villach (Stadt)
JU	Judenburg	VK	Völkermarkt
K	Klagenfurt (Stadt)	VL	Villach (Land)
KB	Kitzbühel	VO	Voitsberg
KI	Kirchdorf an der Krems	W	Wien
KL	Klagenfurt (Land)	WB	Wiener Neustadt (Land)
KN	Knittelfeld	WE	Wels (Stadt)
KO	Korneuburg	WL	Wels (Land)
KR	Krems (Land)	WN	Wiener Neustadt (Stadt)
KS	Krems an der Donau (Stadt)	WO	Wolfsberg
KU	Kufstein	WT	Waidhofen an der Thaya
L	Linz (Stadt)	WU	Wien-Umgebung
LA	Landeck	WY	Waidhofen an der Ybbs (Stadt)
LB	Leibnitz	WZ	Weiz
LE	Leoben	ZE	Zell am See
LF	Lilienfeld	ZT	Zwettl
LI	Liezen		

Table A.3: Variables of euro86 Data Set

1	pop_growth	4	lifeexp_f	7	inhab/doc
2	give_birth	5	lifeexp_m	8	calorie
3	women%	6	inf_mort	9	baby_underw

Table A.4: Observations of euro86 Data Set

a	Austria	h	Hungary
al	Albania	i	Italy
b	Belgium	irl	Ireland
bg	Bulgaria	n	Norway
ch	Switzerland	nl	The Netherlands
cs	Czechoslovakia	p	Portugal
d	Western Germany	pl	Poland
ddr	Eastern Germany	ro	Romania
dk	Denmark	s	Sweden
e	Spain	sf	Finland
f	France	su	Sovjet Union
gb	Great Britain	yu	Yugoslavia
gr	Greece		

Table A.5: Variables of oeamtc Data Set

1	Price	8	Length	15	Speed
2	Tax	9	Width	16	Acceleration
3	Liab.Ins	10	Height	17	Elasticity
4	F.C.Ins	11	Weight.Empty	18	Cons.CC
5	Capacity	12	Weight.Total	19	Cons.Town
6	Performance	13	Boot.Min	20	Cons.Total
7	Torque	14	Boot.Max	21	Cons.Test

Table A.6: Observations of oeamtc Data Set

1	Fiat Stilo	10	Seat Arosa
2	Mercedes C220 CDI S.C.	11	Renault Megane 1.9 dCi
3	Renault Clio 1.5 dCi	12	Ford Mondeo Traveller 2.0 TDdi Ghia
4	Chrysler Voyager CRD	13	Skoda Fabia 1.9 TDI Elegance
5	Hyundai Trajet 2.0 CRDi	14	Fiat Multipla JTD ELX
6	Alfa Romeo 147 JTD	15	Renault Scenic 1.9 DTI
7	Ford Focus 1.8 TDCI	16	Citroen Picasso 2.0 HDI
8	Peugeot 307 XT	17	Opel Zafira 2.0 DTI 16V
9	VW Golf Rabbit TDI	18	Nissan Almera Tino Lux.

Appendix B

Listing of the Algorithms

A complete listing of the algorithms (NIPALS, PLS2 and prediction for PLS2) as implemented in R is displayed in this section. The versions of the algorithms are the ones preferred and used in this work. Other PLS or NIPALS algorithms differ slightly concerning the normalization or calculation of residuals. From the aspect of programming techniques there is no claim of optimality by the author. The programming language itself is R-specific.

B.1 The NIPALS Algorithm

Within the NIPALS algorithm only mean-centering of the data matrix is performed. Should it be necessary to scale the data, this has to be done in advance.

```
function (X,k,it=10,tol=.0001)
♯ fct nipals calculates the principal components
♯ of a given data matrix X according to
♯ the NIPALS algorithm (Wold, 1966).
♯ X...data matrix, k...number of components,
♯ it...maximal number of iterations per component,
♯ tol...precision tolerance for calculation of components
{
X <- scale(X,center=TRUE,scale=FALSE) ♯ mean-centering of data matrix X
nr <- 0
T <- NULL
P <- NULL
for (h in 1:k){
th <- X[,1] ♯ starting value for th is 1st column of X
ende <- FALSE
♯ 3 inner steps of NIPALS algorithm
while (!ende){
nr <- nr+1
ph <- t((t(th)%*%X) * as.vector(1/(t(th)%*%th))) ♯ LS regression for ph
```

```
ph <- ph * as.vector(1/sqrt(t(ph)%*%ph)) ♯ normalization of ph
thnew <- t(t(ph)%*%t(X)) ♯ LS regression for th
prec <- t(th-thnew)%*%(th-thnew) ♯ calculate precision
th <- thnew ♯ refresh th in any case
♯ check convergence of th
if (prec <= tol) {
ende <- TRUE
}
else if (it <= nr) { ♯ too many iterations
ende <- TRUE
cat(``\ nWARNING! Iteration stop in h='',h,`` without convergence!\ n\ n'')
}
}
X <- X-(th%*%t(ph)) ♯ calculate new X
T <- cbind(T,th) ♯ build matrix T
P <- cbind(P,ph) ♯ build matrix P
nr <- 0
}
return(T,P)
}
```

B.2 PLS Algorithms

B.2.1 The PLS2 Algorithm

The PLS2 algorithm is the main algorithm in this work and was implemented in R as follows:

```
function (X,Y,k,it=10,tol=.0001)
♯ fct pls2 calculates the partial least squares model for multidimensional Y
♯ according to the PLS2 algorithm.
♯ X...X-data matrix, Y...Y-data matrix, k...number of components,
♯ it...maximal number of iterations per component,
♯ at least 2 iterations have to be performed!
♯ tol...precision tolerance for calculation of components
{
X <- scale(X,center=TRUE,scale=TRUE) ♯ mean-centering, scaling of data matrix X
Y <- scale(Y,center=TRUE,scale=TRUE) ♯ mean-centering, scaling of data matrix Y
T <- NULL
P <- NULL
U <- NULL
Q <- NULL
W <- NULL
B <- NULL
```

```
for (h in 1:k){
nr <- 0
uh <- Y[,1] ♯ starting value for uh is 1st column of Y
ende <- FALSE
♯ inner steps of PLS2 algorithm
while (!ende){
nr <- nr+1
wh <- t((t(uh)%*%X) * as.vector(1/(t(uh)%*%uh))) ♯ LS regression for wh
wh <- wh * as.vector(1/sqrt(t(wh)%*%wh)) ♯ normalization of wh
thnew <- t((t(wh)%*%t(X))) ♯ LS regression for th
qh <- t((t(thnew)%*%Y) * as.vector(1/(t(thnew)%*%thnew))) ♯ LS regression for qh
qh <- qh * as.vector(1/sqrt(t(qh)%*%qh)) ♯ normalization of qh
uh <- t((t(qh)%*%t(Y))) ♯ LS regression for uh
if (nr>1) {
prec <- t(th-thnew)%*%(th-thnew) ♯ calculate precision
}
th <- thnew ♯ refresh th in any case
♯ check convergence of th after 2nd iteration onwards
if (nr>1) {
if (prec <= tol) {
ende <- TRUE
}
else if (it <= nr) { ♯ too many iterations
ende <- TRUE
cat(``\ nWARNING! Iteration stop in h='',h,`` without convergence!\ n \ n'')
}
}
}
ph <- t((t(th)%*%X) * as.vector(1/(t(th)%*%th))) ♯ LS regression for ph
th <- th * as.vector(1/sqrt(t(ph)%*%ph)) ♯ fitting of th
ph <- ph * as.vector(1/sqrt(t(ph)%*%ph)) ♯ normalization of ph
bh <- (t(th)%*%uh) * 1/(t(th)%*%th) ♯ LS regression for bh
X <- X - (th%*%t(ph)) ♯ calculate new X
Y <- Y - (th%*%t(qh)) * as.vector(bh) ♯ calculate new Y
T <- cbind(T,th) ♯ build matrix T
W <- cbind(W,wh) ♯ build matrix W
Q <- cbind(Q,qh) ♯ build matrix Q
U <- cbind(U,uh) ♯ build matrix U
P <- cbind(P,ph) ♯ build matrix P
B <- c(B,bh) ♯ build vector of diagonal elements of matrix B
}
return(T,P,U,Q,W,B)
}
```

B.2.2 Prediction with PLS2

Note that the $\mathbf{X}'$–matrix for prediction has to be mean-centered and scaled in advance and with the same values as the mean-centering and scaling was done for $\mathbf{X}$ for the calculation of the PLS2 model parameters.

```
function(X',W,P,Q,B,k)
# fct predpls2 calculates predicted T and predicted Y
# X' ...  data matrix for prediction
# Y, W, P, Q, B ...  parameters calculated by PLS2
# k ...  number of components
{
T <- NULL
for (h in 1:k){
th <- (X' %*% P[,h]) * as.vector(1/(t(P[,h])%*%P[,h]))
X' <- X' - (th %*% t(P[,h]))
if (h==1) Y <- as.vector(B[h]) * (th %*% t(Q[,h]))
else  Y <- Y + as.vector(B[h]) * (th %*% t(Q[,h]))
T <- cbind(T,th)
}
return(T,Y)
}
```

B.2.3 The PLS1 Algorithm

The PLS1 algorithm,i.e., PLS2 for a one-dimensional $\mathbf{y}$–part, was implemented in the following way.

```
function (X,y,k)
# fct pls1 calculates the partial least squares model for onedimensional y
# according to the PLS1 algorithm (Wold, 1966).
# X...X-data matrix, y...y-data vector, k...number of components,
# no iteration, no convergence checks
{
X <- scale(X,center=TRUE,scale=TRUE) # mean-centering and scaling of data matrix X
y <- scale(y,center=TRUE,scale=TRUE) # mean-centering and scaling of data vector y
T <- NULL
P <- NULL
U <- NULL
W <- NULL
q <- NULL
b <- NULL
for (h in 1:k){
wh <- t((t(y)%*%X) * as.vector(1/(t(y)%*%y))) # LS regression for wh
wh <- wh * as.vector(1/sqrt(t(wh)%*%wh)) # normalization of wh
th <- t((t(wh)%*%t(X)) * as.vector(1/(t(wh)%*%wh))) # LS regression for th
```

```
ph <- t((t(th)%*%X) * as.vector(1/(t(th)%*%th))) ♯ LS regression for ph
th <- th * as.vector(1/sqrt(t(ph)%*%ph)) ♯ fitting of th
ph <- ph * as.vector(1/sqrt(t(ph)%*%ph)) ♯ normalization of ph
bh <- (t(th)%*%y) * 1/(t(th)%*%th) ♯ LS regression for bh
X <- X - (th%*%t(ph)) ♯ calculate new X
y <- y - th * as.vector(bh) ♯ calculate new y
T <- cbind(T,th) ♯ build matrix T
W <- cbind(W,wh) ♯ build matrix W
q <- cbind(q,1) ♯ build vector q
P <- cbind(P,ph) ♯ build matrix P
b <- c(b,bh) ♯ build vector b of diagonal elements of matrix B
}
return(T,P,U,W,q,b)
}
```

Bibliography

A. Basilevsky. *Statistical Factor Analysis and Related Methods: Theory and Applications*. Wiley & Sons, New York, 1994.

A.J. Burnham, R. Viveros, and J.F. Macgregor. Frameworks for latent variable multivariate regression. *Journal of Chemometrics*, 10:31–45, 1996.

C. Cassel, P. Hackl, and A.H. Westlund. Robustness of partial least-squares method for estimating latent variable quality structures. *Journal of Applied Statistics*, 26(4):435–446, 1999.

D.J. Cummins and C.W. Andrews. Iteratively reweighted partial least squares: A performance analysis by Monte Carlo simulation. *Journal of Chemometrics*, 9:489–507, 1995.

S. De Jong. SIMPLS: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 18:251–263, 1993.

D.L. Donoho. *Breakdown properties of multivariate location estimators*. PhD thesis, Qualifying paper, Harvard University, Boston, 1982.

S. Duckett and B. Gilbert. *Foundations of Spectroscopy*. Oxford University Press, New York, 2000.

B.S. Everitt and G. Dunn. *Applied Multivariate Data Analysis*. Arnold, London, 2001.

P. Filzmoser. Robust principal component regression. In S. Aivazian, Y. Kharin, and H. Rieder, editors, *Proceedings of the Sixth International Conference on Computer Data Analysis and Modeling*, volume 1, pages 132–137. Belarus State University, Minsk, 2001.

P. Filzmoser. Robust factor analysis methods and applications. In G. Marcoulides and I. Moustaki, editors, *Latent Variable and Latent Structure Models*, pages 153–194. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2002.

P. Filzmoser, C. Dehon, and C. Croux. Outlier resistant estimators for canonical correlation analysis. In J.G. Betlehem and P.G.M. van der Heijden, editors, *COMPSTAT, Proceedings in Computational Statistics*, pages 385–390. Physica-Verlag, Heidelberg, 2000.

I.E. Frank and J.H. Friedman. A statistical view of some chemometrics regression tools. *Technometrics*, 35(2):109–148, 1993.

P.H. Garthwaite. An interpretation of partial least squares. *Journal of the American Statistical Association*, 89(425):122–127, 1994.

P. Geladi and B.R. Kowalski. Partial least–squares regression: A tutorial. *Analytica Chimica Acta*, 185:1–17, 1986.

J.A. Gil and R. Romera. On robust partial least squares (PLS) methods. *Journal of Chemometrics*, 12:365–378, 1998.

J. Gower and D. Hand. *Biplots.* Chapman & Hall, New York, 1996.

M.I. Griep, I.N. Wakeling, P. Vankeerberghen, and D.L. Massart. Comparison of semirobust and robust partial least squares procedures. *Chemometrics and Intelligent Laboratory Systems*, 29:37–50, 1995.

J. Hartung, B. Elpelt und H.-K. Klösener. *Statistik: Lehr- und Handbuch der angewandten Statistik.* Oldenbourg Verlag, München, 1998.

I.S. Helland. Partial least squares regression and statistical methods. *Scandinavian Journal of Statistics*, 17:97–114, 1990.

J.E. Jackson. *A User's Guide to Principal Components.* John Wiley & Sons, New York, 1991.

K.V. Mardia, J.T. Kent, and J.M. Bibby. *Multivariate Analysis.* Acad. Press, London, 1979.

R.A. Maronna and V.J. Yohai. The behaviour of the Stahel-Donoho robust multivariate estimator. *Journal of the American Statistical Association*, 90:330–341, 1995.

H. Martens and T. Naes. *Multivariate Calibration.* John Wiley & Sons, New York, 1989.

M.R. Oliveira and J.A. Branco. Comparison of three methods for robust redundancy analysis. In R. Dutter, P. Filzmoser, U. Gather, and P.J. Rousseeuw, editors, *Developments in Robust Statistics: Proceedings of ICORS 2001*, pages 286–294. Springer Verlag, Heidelberg, 2002.

R.J. Pell. Multiple outlier detection for multivariate calibration using robust statistical techniques. *Chemometrics and Intelligent Laboratory Systems*, 52:87–104, 2000.

G.R. Phillips and E.M. Eyring. Comparison of conventional and robust regression in analysis of chemical data. *Anal. Chem.*, 55(7):1134–1138, 1983.

C.R. Rao and H. Toutenberg. *Linear Models: Least Squares and Alternatives.* Springer, New York, 1995.

H. Rauth and G. Sedlacek. Classification of Austria into homogeneous mathematical regions. Technical Report RIS-1996-7, Dept. of Statistics and Probability Theory, Univ. of Technology, Vienna, 1996. Unpublished. (In German).

P.J. Rousseeuw. Least median of squares regression. *Journal of the American Statistical Association*, 79:871–880, 1984.

P.J. Rousseeuw. Multivariate estimation with high breakdown point. In W. Grossmann, G. Pflug, I. Vincze, and W. Wertz, editors, *Mathematical Statistics and Applications, Vol. B*, pages 283–297. Reidel, Dordrecht, The Netherlands, 1985.

P.J. Rousseeuw and A.M. Leroy. *Robust Regression and Outlier Detection*. Wiley & Sons, New York, 1987.

P.J. Rousseeuw and K. Van Driessen. Computing LTS regression for large data sets. Technical report, University of Antwerp, 1998. Submitted.

A.F. Siegel. Robust regression using repeated medians. *Biometrika*, 69:242–244, 1982.

W.A. Stahel. *Robust estimation: Infinitesimal optimality and covariance matrix estimators*. PhD thesis, ETH, Zürich, 1981.

M. Tenenhaus. *La Régression PLS*. Éditions Technip, Paris, 1998a.

M. Tenenhaus. L'Approche PLS. Avec le concours de la Fondation HEC, Groupe HEC, 78351 Jouy-en-Josas Cedex, France, 1998b.

E.V. Thomas and N. Ge. Development of robust multivariate calibration models. *Technometrics*, 42(2):168–177, 2000.

I.N. Wakeling and H.J.H. Macfie. A robust PLS procedure. *Journal of Chemometrics*, 6: 189–198, 1992.

W. Werther. *Einsatz von Methoden der explorativen Datenanalyse zur Interpretation und Klassifikation von Massenspektren*. PhD thesis, Institut für Allgemeine Chemie der TU Wien, Vienna University of Technology, 1993.

H. Wold. Nonlinear estimation by iterative least square procedures. In F.N. David, editor, *Research Papers in Statistics: Festschrift for Jerzy Neyman*, pages 411–444. Wiley, New York, 1966.

S. Wold, A. Ruhe, H. Wold, and W.J. Dunn. The collinearity problem in linear regression. The partial least squares (PLS) approach to generalized inverses. *SIAM J. Sci. Stat. Comput., Society for Industrial and Applied Mathematics*, 5(3):735–742, 1984.