# Review of robust multivariate statistical methods in high dimension ☆

Peter Filzmoser[a,*], Valentin Todorov[b]

[a]*Department of Statistics and Probability Theory, Vienna University of Technology, Wiedner Hauptstr. 8-10, 1040 Vienna, Austria*
[b]*United Nations Industrial Development Organization (UNIDO), Vienna International Centre, P.O. Box 300, A-1400 Vienna, Austria*

## Abstract

General ideas of robust statistics, and specifically robust statistical methods for calibration and dimension reduction are discussed. The emphasis is on analyzing high-dimensional data. The discussed methods are applied using the packages **chemometrics** and **rrcov** of the statistical software environment R. It is demonstrated how the functions can be applied to real high-dimensional data from chemometrics, and how the results can be interpreted.

*Keywords:* robustness, multivariate analysis, PLS, PCA, validation, diagnostics

## 1. Introduction

Statistical methods are usually based on model assumptions, like normal distribution of the underlying data, or independence of the observations. In practice, however, such assumptions may be violated and invalid for the data set that needs to be analyzed. Practical data can include outliers, they can be plagued with heavy-tailed distributions, and they can have other problems such that strict model assumptions are not fulfilled. It is then questionable

☆The views expressed herein are those of the authors and do not necessarily reflect the views of the United Nations Industrial Development Organization.

*Corresponding author

*Email addresses:* `p.filzmoser@tuwien.ac.at` (Peter Filzmoser ), `v.todorov@unido.org` (Valentin Todorov)

if the application of “classical” statistical methods relying on these assumptions lead to valid results, or if the results are even misleading. Especially for high-dimensional data this issue is difficult, because it is practically impossible to “see” data outliers with exploratory data analysis techniques, or to actually check the model assumptions. It is then also difficult to judge on the consequences of deviations from idealized model assumptions.

Robust statistics attempts to provide solutions to the above mentioned problems. Robust statistical methods aim to give reliable results even if the strict model assumptions that are needed for the classical counterparts are not fulfilled. The formal approach to robustness also allows to characterize the robustness properties of a statistical estimator. Important concepts in this context are the influence function or the breakdown point [see, e.g., 1]. The influence function assesses an estimator with respect to infinitesimal contamination. The breakdown point, on the other hand, deals with the problem of large contamination. It characterizes the smallest amount of contamination that can cause an estimator to yield arbitrary values. Robust estimators have a positive breakdown point, meaning that a certain part of the data could be “outliers”, and the estimator gives still useful results. The arithmetic mean, for example, has a breakdown point of zero, because moving even a single observation towards plus or minus infinity would lead to a meaningless result. The median has a breakdown point of one half, since even an arbitrary shift of (approximately) half of the data can still not have a disastrous effect on the result–in contrary, usually we observe only a small change of the result. Consequently, robust statistical estimators focus on the homogeneous data majority, but not on the minority formed by deviating outlying data points. After applying robust estimators, deviating observations can be identified as outliers, because the deviation of each observation to an estimated value or model is now reliable.

A further important concept is the statistical (asymtptotic) efficiency of an estimator. It depends on the considered data distribution and on the (asymptotic) variance of the estimator [e.g. 2]. It can be shown that the efficiency is in the interval $[0, 1]$, where 1 refers to a highly efficient estimator. For example, under normal distribution, the arithmetic mean has an efficiency of 1, whereas the median only achieves a value of about 0.64. In other words, for obtaining the same precision of the location estimation, we need about one third more data for the median than for the mean.

Outliers in multivariate data are not necessarily values that are extreme along one coordinate. Figure 1 shows an example in two dimensions, where the data are elliptically symmetric around the origin, and one observation (marked by +) is deviating from the data structure. This data point is neither extreme along $x_1$ nor along $x_2$. Not a univariate but a multivariate treatment is necessary in order to identify this observation as outlier. In addition to the data points, Figure 1 shows a so-called tolerance ellipse that is supposed to include the "inner" 97.5% observations of a bivariate normal distribution. The parameters for constructing the tolerance ellipse are the robustly estimated location and covariance. Since the ellipse reveals the shape of the data structure, the multivariate outlier immediately becomes visible. In this example, also classical estimates would be able to identify this outlier, but in other situations the outliers themselves could affect the classical estimates, and thus the appearance of the ellipse. In the worst case, the ellipse could get inflated such that the outlier is even within the ellipse. This unwanted effect is called *masking effect*. Moreover, if other non-outlying observations are falling outside the ellipse, they would be erroneously declared as outliers. This phenomenon is called *swamping effect* [see, e.g., 3]. The use of robust estimators can avoid such problems.

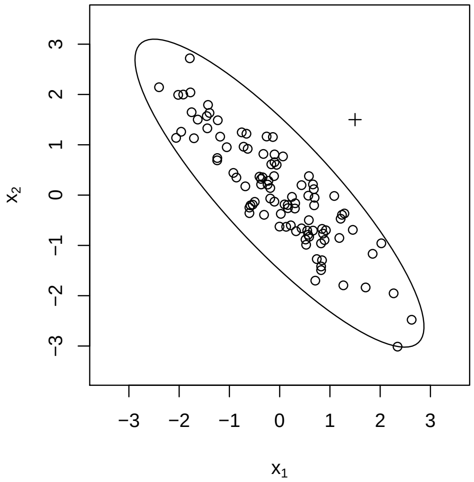

Figure 1: Two dimensional data set with one outlier, and tolerance ellipse revealing the outlier.

Since robust statistics is a rapidly developing field, it would be impossible to mention and describe all the major aspects. We refer to the book Maronna et al. [2] which contains also more recent important areas of robustness, and to Frosch-Møller et al. [4] for a review of robust multivariate methods. Here we focus on the main ideas of robust statistics, and explain these concepts for robust regression and principal component analysis. For both subjects we pay attention to dealing with high-dimensional data. A further important aspect is the application of the methods using the software environment R [5]. The freely available statistical software has gained highest importance not only in the "world of statistics" but also in many other fields. It includes the latest developments of statistical methods in form of documented functions, and offers plenty of possibilities also for robust estimation.

## 2. Basic concepts of robust linear regression

Consider the multiple linear regression model

$$y_i = \boldsymbol{x}_i^t \boldsymbol{\beta} + \varepsilon_i \quad \text{for } i = 1, \ldots, n \tag{1}$$

with $n$ observations $y_i$ of the response and of the explanatory variables $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})^t$ (an intercept term is included by setting $x_{i1} = 1$), the vector of regression coefficients $\boldsymbol{\beta}$, and the error term $\varepsilon_i$. The most widely used estimator for the regression coefficients is the least-squares (LS) estimator, which is defined as

$$\widehat{\boldsymbol{\beta}}_{LS} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \left( y_i - \boldsymbol{x}_i^t \boldsymbol{\beta} \right)^2 . \tag{2}$$

The solution $\widehat{\boldsymbol{\beta}}_{LS}$ is thus obtained by minimizing the sum of squared deviations from the values $y_i$ to a projection of $\boldsymbol{x}_i$ on any $p$-dimensional vector $\boldsymbol{\beta}$. The LS estimator is known as the best linear unbiased estimator (BLUE) if the errors $\varepsilon_1, \ldots, \varepsilon_n$ are independent identically distributed according to $N(0, \sigma^2)$, with the same residual variance $\sigma^2$ [see, e.g., 6]. Naturally, if these strict assumptions are violated, the LS estimator loses its good properties, and another estimator could be preferable.

Figure 2 (left) illustrates the problem of violations from model assumptions for LS regression with one predictor and one response variable. The majority of data points follows a linear trend, and for these observations also the above assumptions seem to be valid. However, a group of points, named "vertical outliers", is deviating in the direction of the response variable, and another group called “leverage points” forms outliers along the explanatory variable. Vertical outliers usually have a smaller effect on the LS estimation, but leverage points can attract (“lever”) the LS regression line that would be valid for the data majority. This effect is shown in Figure 2 (right), where the LS estimation using all data points is not useful at all, neither for the data points following the linear trend, nor for predicting outliers. The line denoted by “robust fit” follows the linear trend formed by the data majority. Ideally it would correspond to the LS regression line computed from the data subset without vertical outliers and leverage points. However, in real problems–and especially if more than one predictor variable is involved–it would be difficult or even impossible to judge which observations form vertical outliers or leverage points, even though the latter have to be identified in the $p$-dimensional space of the explanatory variables. Thus an automatic procedure is desirable that allows to downweight outlying observations in an appropriate manner.

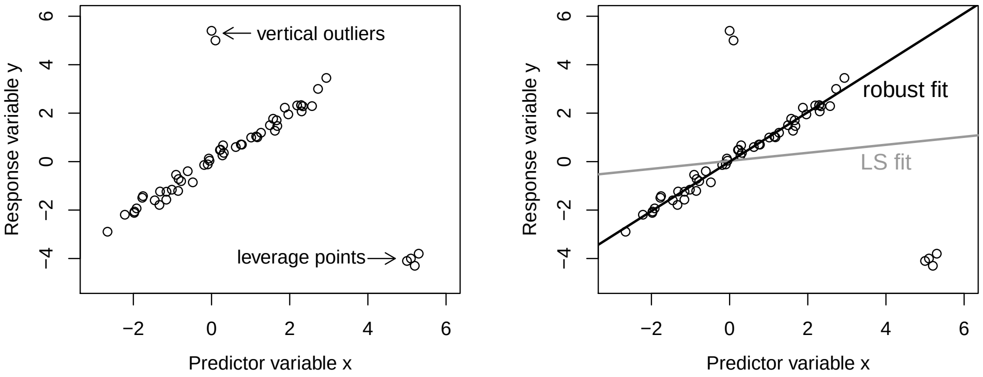

Figure 2: Linear regression problem with vertical outliers and leverage points (left); result of LS regression and robust regression (right).

The estimation of the regression parameters, as well as inference statistics and diagnostics, take into account the residuals, which are defined as $r_i(\boldsymbol{\beta}) = y_i - \boldsymbol{x}_i^t\boldsymbol{\beta}$, for $i = 1, \ldots, n$, and depend on the parameter vector $\boldsymbol{\beta}$. For the estimated regression parameters from Figure 2 (right) we can inspect the residuals in Figure 3. For the LS fit (left) the residuals of the outliers are large, but also residuals of the regular observations following the linear trend are increasing towards the boundary points. This will make it difficult to distinguish outliers from regular points later on. The distribution of the residuals may even look like a normal distribution. In contrast, for the robust fit (right) the residuals of the regular observations are very small, and only the residuals of the outliers are large, making it easy to identify them. Particularly the residuals from the leverage points are large, and it is thus easy to see that within the LS criterion (2) their square would lead to very large values that increase the objective function. An alternative solution (given by the LS fit) finds a compromise that can also reduce the value of the objective function.

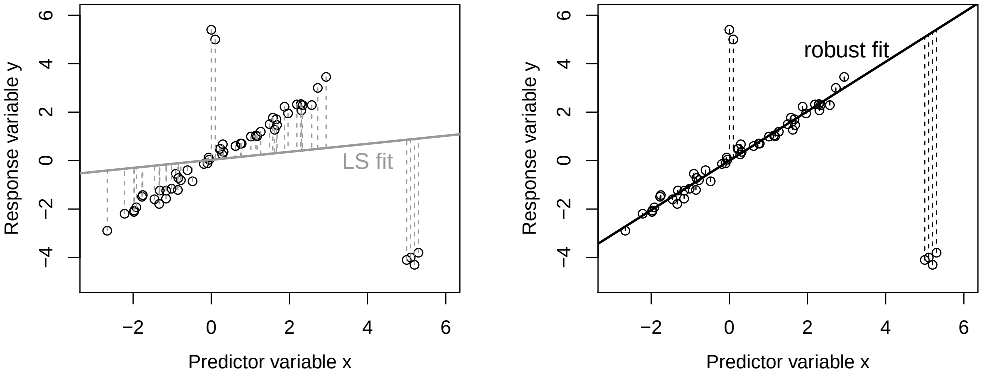

Figure 3: LS regression with residuals (left), and robust regression with residuals (right).

The performance of a regression model can be evaluated using an appropriate performance measure like the *standard error of prediction* (SEP)

$$\mathrm{SEP} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \hat{y}_i - \mathrm{bias})^2} \quad \text{with} \quad \mathrm{bias} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i), \qquad (3)$$

or the *root mean squared error* (RMSE)

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}. \qquad (4)$$

Here, $\hat{y}_i = \boldsymbol{x}_i^t \hat{\boldsymbol{\beta}}$ are the predicted values of the response variable, using the estimated regression parameters $\hat{\boldsymbol{\beta}}$ [see, e.g., 7]. When evaluating the performance measures for the example data set above (i.e. we are using the original "calibration" data), the SEP and RMSE values are shown in Table 1. Both measures lead to comparable results, but the values for the LS fit is smaller than that for the robust fit. Accordingly, the practitioner would prefer LS regression. However, note that the performance measures in (3) and (4) are not robust against outliers, because each observation gets the same contribution in the formulas. The influence of outliers to the performance measures can be reduced by trimming for example the 10% of the largest contributions. The results for the example are shown in Table 1, and they lead to a contrary picture: the values of the criteria for the robust fit are much smaller than for the LS fit. This is also visible in Figure 3, where an exclusion of 10% of the largest residuals would clearly give preference to the robust fit. The trimming value 10% is rather subjective here, and it could be taken higher if the data quality is worse.

A regression model is usually used for the prediction of new test data. Suppose we have given for the above example test data following the linear trend formed by the majority of data points in Figure 3, without any outliers. There are, however, still two different models, and only the robust model will lead to small residuals (prediction errors). A check for normal distribution of the test set residuals can be done by a Q-Q plot [see, e.g., 7], and it may look as shown in Figure 4. These plots indeed suggest that the assumption of normal distribution is valid for both models. However, the scale of the residuals from both fits is very different.

The above performance measures can also be evaluated for test data, and the results are shown in Table 1. Here the robust fit leads to much smaller values than the LS model. Note that since the test data contain no outliers, the SEP and the RMSE as well as their 10% trimmed versions are comparable and lead to the same conclusions.

## 3. Methods for robust linear regression

One of the main problems of the non-robustness of LS regression is the square in the objective function (2), and that large values would dominate

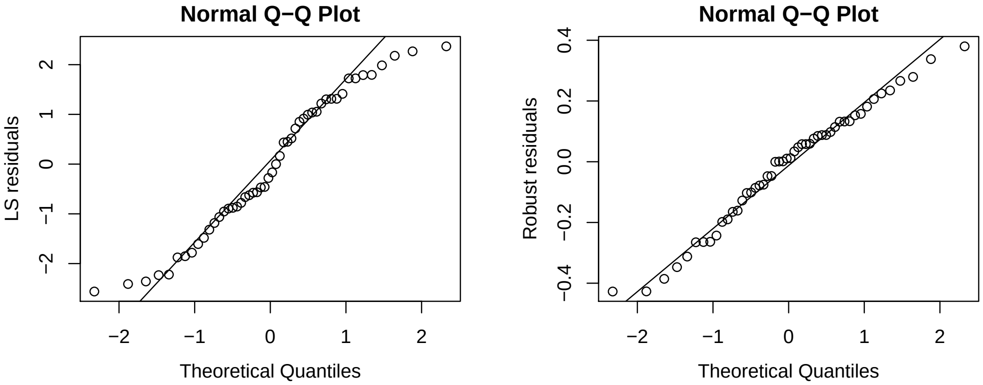

Figure 4: Q-Q plots of outlier-free test set residuals from LS regression (left) and robust regression (right).

Table 1: Performance measures for the calibration data and an outlier-free test data set, see Figures 2 and 4.

| | Calibration data | | Test data | |
|---|---|---|---|---|
| | LS fit | robust fit | LS fit | robust fit |
| SEP | 2.09 | 2.66 | 1.42 | 0.20 |
| RMSE | 2.07 | 2.68 | 1.41 | 0.20 |
| SEP 10% | 1.14 | 0.18 | 1.12 | 0.14 |
| RMSE 10% | 1.13 | 0.18 | 1.10 | 0.14 |

the sum. An idea for a more robust estimation of the regression parameters is to use an *M estimator* for regression [8, 9], defined as

$$\widehat{\boldsymbol{\beta}}_M = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho\left(y_i - \boldsymbol{x}_i^t \boldsymbol{\beta}\right) = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho\left(r_i(\boldsymbol{\beta})\right). \tag{5}$$

The function $\rho$ can be seen as a loss function applied to the residuals. Clearly, for LS regression $\rho(r) = r^2$, and thus the LS criterion (2) is a special case of the criterion (5). The idea is to downweight large (absolute) residuals. Figure 5 shows two choices of the $\rho$ function: the left picture is the quadratic function, corresponding to the LS criterion, where large (absolute) residuals can become very dominating in the used criterion for obtaining the regression estimates. In the right picture the residuals within $[-c, c]$ (for a constant $c$)

have the same contribution to the criterion as before, but for larger (absolute) values it remains bounded. The choice of the $\rho$ function will also affect the properties of the resulting regression estimator [see 2, for more details].

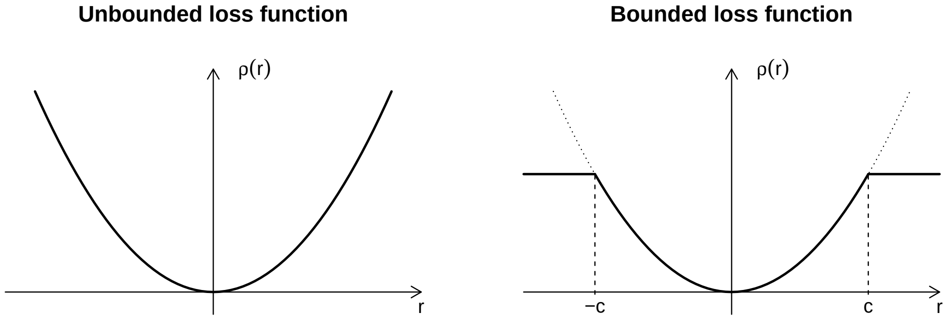

Figure 5: Different choices of the $\rho$ function for the M estimator. Left: quadratic function as used for LS regression; right: bounded $\rho$ function according to Huber [10].

The criterion (5) has the disadvantage that it is not "regression equivariant", meaning that rescaling the response variable with a multiplicative factor $h$ does not lead to regression coefficients $h\widehat{\boldsymbol{\beta}}_M$. Therefore, the general definition of the M estimator for regression is

$$\widehat{\boldsymbol{\beta}}_M = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho\left(\frac{r_i(\boldsymbol{\beta})}{\hat{\sigma}}\right). \tag{6}$$

$\hat{\sigma}$ is a robust scale estimator of the residuals which, however, depends on the unknown regression coefficients $\boldsymbol{\beta}$. For finding a solution, Equation (6) can be differentiated, which gives

$$\sum_{i=1}^{n} \psi\left(\frac{r_i(\boldsymbol{\beta})}{\hat{\sigma}}\right) \boldsymbol{x}_i = 0, \tag{7}$$

with $\psi = \rho'$. Putting $W(r) = \psi(r)/r$ allows to write (7) as

$$\sum_{i=1}^{n} w_i \left(y_i - \boldsymbol{x}_i^t \boldsymbol{\beta}\right) \boldsymbol{x}_i = 0, \tag{8}$$

with $w_i = W(r_i(\boldsymbol{\beta})/\hat{\sigma})$. This shows that the problem can be written in terms of the normal equations with weights for the observations. For the solution one can use an iterative scheme called *iteratively reweighted least-squares* (IRLS), but the problem is that usually many local minima exist. Therefore it is crucial to initialize the procedure with a good (i.e. robust) starting value $\hat{\boldsymbol{\beta}}_0$.

A further problem is how to obtain $\hat{\sigma}$, and how to define a robust scale estimator. One possibility is to use an *M estimator of scale*, defined as the solution $\sigma$ of the equation

$$\frac{1}{n}\sum_{i=1}^{n}\rho\left(\frac{r_i(\boldsymbol{\beta})}{\sigma}\right) = \delta, \tag{9}$$

for a certain $\rho$ function and a constant $\delta$ [see 8, 9]. Using a weight function $W_\sigma(z) = \rho(z)/z^2$, Equation (9) can be rewritten as

$$\sigma^2 = \frac{1}{n\delta}\sum_{i=1}^{n} w_i r_i^2(\boldsymbol{\beta}), \tag{10}$$

with weights $w_i = W_\sigma(r_i(\boldsymbol{\beta})/\sigma)$. By taking an appropriate starting value $\sigma_0$, the estimator $\hat{\sigma}$ can be obtained by an iterative procedure [2]. Using this robust scale estimator for the problem

$$\widehat{\boldsymbol{\beta}}_S = \underset{\boldsymbol{\beta}}{\operatorname{argmin}}\, \hat{\sigma}\left(r_1(\boldsymbol{\beta}), \ldots, r_n(\boldsymbol{\beta})\right) \tag{11}$$

results in the *regression S estimator* $\widehat{\boldsymbol{\beta}}_S$. The S estimator achieves the maximum breakdown point, but has low efficiency [2]. A combination of both, robustness and controllable efficiency, can be obtained by the following procedure:

- compute an initial estimator $\hat{\boldsymbol{\beta}}_0$, using a regression S estimator,
- compute a robust scale $\hat{\sigma}$ of the residuals $r_i(\hat{\boldsymbol{\beta}}_0)$, using Equation (9) for the M estimator of scale,
- take an M estimator of regression to obtain the MM estimator of regression as

  $$\widehat{\boldsymbol{\beta}}_{MM} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n}\rho\left(\frac{r_i(\boldsymbol{\beta})}{\hat{\sigma}}\right) \tag{12}$$

  as a local solution within an iterative algorithm, starting from $\hat{\boldsymbol{\beta}}_0$.

The MM estimator of regression has the breakdown point of $\hat{\boldsymbol{\beta}}_0$ and an efficiency that can be controlled by tuning parameters (e.g. 0.85). For more details we refer to Maronna et al. [2].

*Example: MM regression in R*

As an example we consider the milk data set, available in the R package **robustbase** as data set `milk`. For 86 containers of milk several characteristics were analyzed. We consider as the response variable (named X4) the casein content and as explanatory variable the cheese dry substance measured in the laboratory (named X6). We only use one explanatory variable in order to visualize the resulting regression lines. The R code for generating the left plot in Figure 6 is shown below. Since MM regression boils down to computing appropriate weights for the observations, the right plot of Figure 6 visualizes these weights as symbol size of the observations: the smaller the weight, the larger the symbol. It can be seen that the clear outliers receive very low weight, but also that points being not so far from the MM regression line are downweighted.

```
> library(robustbase)
> data(milk)
> plot(X4~X6,data=milk,xlab="Casein content",ylab="Dry substance")
> reslm <- lm(X4~X6, data=milk)          # LS regression
> reslmrob <- lmrob(X4~X6, data=milk)    # MM regression
> abline(reslm,lty=2)                    # plot LS regression line
> abline(reslmrob,lty=1)                 # plot MM regression line
> legend("topright",leg=c("LS","MM"),lty=c(2,1))
```

Details of the MM estimation can be seen by the following command:

```
> summary(reslmrob)
```

The resulting (shortened) output is shown in Table 2. The structure of the output is similar to that of LS regression, including information of the residuals and robust inference statistics. The weights for the observations resulting from MM regression are also summarized, and this information was already displayed in Figure 6 (right). Note that robust regression methods, like the well-known Least Trimmed Squares (LTS) regression [see 11] assign

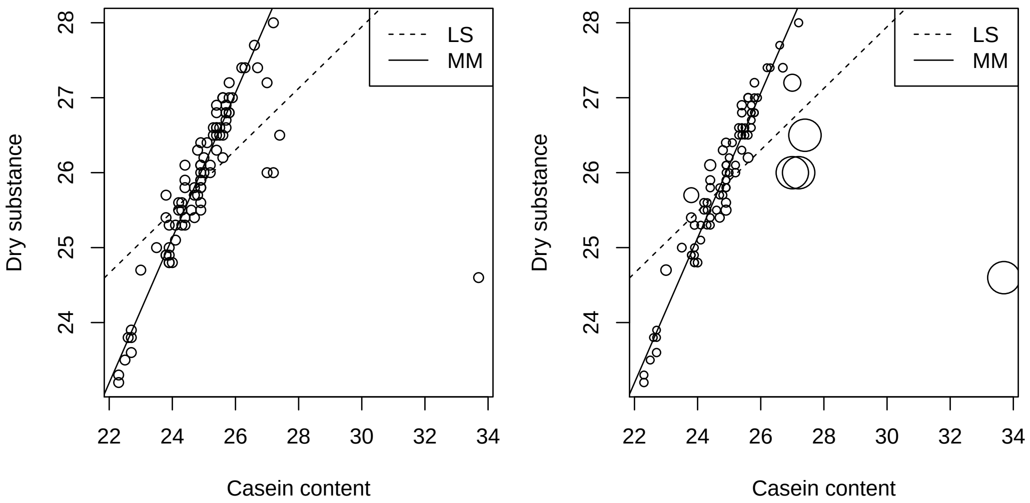

Figure 6: LS and MM regression for the `milk` data set. The symbol size in the right panel refers to the weight received from MM regression: a large symbol means a small weight, which is assigned to an outlying observation.

weights of zero or one to the observations, which is still robust but leads to (considerably) lower efficiency than MM regression [see 2].

Regression is typically applied in situations where the number of observations is much larger than the number of variables. LS regression becomes numerically instable if the number of regressor variables comes close to the number of observations because of singularity problems. MM regression has even stricter constraints on the number of variables relative to the number of observation. In any case, typical applications in chemometrics have many more variables than observations, and they cannot be handled with these regression methods. In the next section, however, we will discuss a robust regression method that can handle the case $n < p$.

## 4. Robust linear regression in high dimension

Regression problems become challenging when the number $p$ of explanatory variables exceeds the number of observations. Methods for dealing with this kind of problems are described for instance in Hastie et al. [12]. Only in recent years, robust counterparts to such methods were introduced, like Khan et al. [13] for robust linear model selection, or approaches as proposed in [14] for robustifying partial least squares (PLS) regression [see also 7]. Here we

Table 2: Part of the summary of MM regression for the milk data set.

```
Weighted Residuals:
     Min       1Q   Median       3Q      Max
-9.91375 -0.19793 -0.03457  0.12813  0.76380

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.91138    0.81009   2.359   0.0206 *
Casein       0.96743    0.03248  29.787   <2e-16 ***
---
Robust residual standard error: 0.2568
Convergence in 11 IRWLS iterations

Robustness weights:
 4 observations c(1,2,41,70) are outliers with |weight|=0 (<0.0012)
 7 weights are ~= 1. The remaining 75 ones are summarized as
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.2722  0.8642  0.9532  0.9028  0.9858  0.9989
```

focus on a robust PLS method called *partial robust M* (PRM) regression [15]. The main idea is to use an M estimator for regression (see Section 3) not on the complete but only for a partial information of the explanatory variables. This partial information is obtained via so-called latent variables that need to be extracted in a robust manner.

The linear regression model from (1) can be written in matrix notation as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{13}$$

where the vector $\boldsymbol{y}$ contains the $n$ observations of the response variable, the $n \times p$ matrix $\boldsymbol{X}$ has in its rows the observations $\boldsymbol{x}_i$, and $\boldsymbol{\varepsilon}$ includes all the error terms $\varepsilon_i$. In PLS regression we switch to a latent variable model

$$\boldsymbol{y} = \boldsymbol{T}\boldsymbol{c} + \boldsymbol{\delta}, \tag{14}$$

with the score matrix $\boldsymbol{T}$ of dimension $n \times a$, regression coefficients $\boldsymbol{c}$, and error terms $\boldsymbol{\delta}$. The number $a$ of latent variables is smaller than $p$, and usually even much smaller. The latent variables $\boldsymbol{w}_j$ (and thus the columns of $\boldsymbol{T}$) are obtained sequentially, for $j = 1, \dots, a$, by

$$\boldsymbol{w}_j = \underset{\boldsymbol{w}}{\operatorname{argmax}} \operatorname{Cov}(\boldsymbol{y}, \boldsymbol{X}\boldsymbol{w}) \tag{15}$$

under the constraints

$$\|\boldsymbol{w}\| = 1 \quad \text{and} \quad \operatorname{Cov}(\boldsymbol{X}\boldsymbol{w}, \boldsymbol{X}\boldsymbol{w}_k) = 0 \text{ for } 1 \leq k < j. \tag{16}$$

The score matrix is obtained by $\boldsymbol{T} = \boldsymbol{X}\boldsymbol{W}$, where $\boldsymbol{W}$ contains in its columns the latent variables $\boldsymbol{w}_j$.

For classical PLS, "Cov" in (15) and (16) is taken as the sample covariance matrix. In the robust case, the estimation of the covariance needs to be robustified. For PRM this is done by M estimation. According to Equation (6), the model (14) can be written as

$$\widehat{\boldsymbol{c}} = \underset{\boldsymbol{c}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho\left(\frac{y_i - \boldsymbol{t}_i^t \boldsymbol{c}}{\hat{\sigma}}\right), \tag{17}$$

where $\boldsymbol{t}_i$ denotes the $i$th row of $\boldsymbol{T}$, and $\hat{\sigma}$ is an estimator of the residual scale. As it has been shown in Equation (8), the solution boils down to deriving weights $w_i$ for each observation. These weights can be used to robustify the estimation of the covariance. Accordingly, (15) and (16) can be written as

$$\boldsymbol{w}_j = \underset{\boldsymbol{w}}{\operatorname{argmax}} \frac{1}{n} \sum_{i=1}^{n} w_i y_i (\boldsymbol{x}_i^t \boldsymbol{w}) \tag{18}$$

under the constraints

$$\|\boldsymbol{w}\| = 1 \quad \text{and} \quad \frac{1}{n} \sum_{i=1}^{n} w_i (\boldsymbol{x}_i^t \boldsymbol{w})(\boldsymbol{x}_i^t \boldsymbol{w}_k) = 0 \text{ for } 1 \leq k < j, \tag{19}$$

assuming that response and explanatory variables have been (robustly) centered. The final solution can be found within an iterative scheme, and it is again crucial to have good starting values [see 15, for details]. This procedure has the advantage that it yields robust solutions that are fast to compute.

*Example: PRM regression in R*

We use a data set included in the R package **chemometrics**, containing the concentration of glucose and ethanol (in g/L) for $n = 166$ alcoholic fermentation mashes of different feedstock (rye, wheat and corn) [see 16]. For the mashes the first derivatives of near infrared spectroscopy (NIR) absorbance values at 1115-2285 nm are available, leading to 235 explanatory variables. Glucose will be used as the response variable. The data are prepared in R by:

```
> library(chemometrics)
> data(NIR)
> X <- NIR$xNIR
> y <- NIR$yGlcEtOH$Glucose
```

For choosing an optimal number of PRM components, 10-fold cross-validation (CV) is used for a maximum of $a = 40$ components (this high number is only taken for illustrative purposes):

```
> res.prmcv <- prm_cv(X, y, a = 40)
```

This command also generates a plot for the optimal number of components, which is shown in Figure 7. The SEP value, see (3), and its 20% trimmed version are used as performance measures. The indicated intervals correspond to mean plus/minus one standard error of the 20% trimmed SEP values resulting from 10-fold CV. The optimal number of components is selected as the lowest number whose prediction error mean is below the minimal prediction error mean plus one standard error [see 7]. Here, 20 components are selected, leading to a prediction error of 5.12 (±0.9).

Using the computed 20 PRM components, the predicted values of the response and the residuals can be computed. Plots of measured versus predicted values, and predicted values versus residuals can be visualized with

```
> plotprm(res.prmcv, y)
```

(not shown here). The regression coefficients, weights, scores, and loadings for the optimal number of components can be obtained by:

```
> prm(X, y, a = res.prmcv$optcomp)
```

A more careful and detailed model selection can be done with repeated double cross-validation (rdCV) [see 17, 18, for details]. The procedure is rather time consuming. Within an "inner loop", $k$-fold CV is used to determine an optimal number of components, which then is applied to a "test set" resulting from an "outer loop". The procedure is repeated a number of times. rdCV with 20 replications is run by

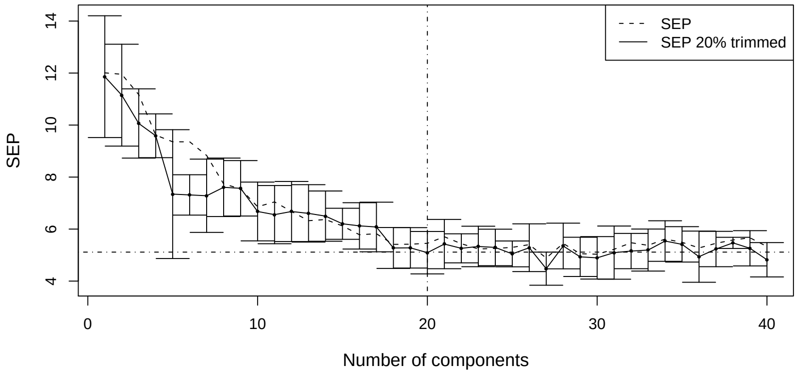

Figure 7: Output of function `prm_cv`, computing PRM regression models with 1 to 40 components. Dashed line: mean of SEP values from CV. Solid part: mean and standard deviation of 20% trimmed SEP values from CV. Vertical and horizontal line correspond to the optimal number of components (after standard-error-rule) and the according 20% trimmed SEP mean, respectively.

```
> res.prmdcv <- prm_dcv(X, y, a = 40, repl = 20)
```

but it requires about four hours on a standard PC (compared to about four minutes for single CV for PRM). The frequencies of the optimal numbers of components can be seen by

```
> plotcompprm(res.prmdcv)
```

and they are shown in Figure 8. There is a clear peak at 20 components, meaning that a model with 20 components has been optimal in most of the experiments within rdCV. Note that here we obtain the same result as for single CV.

In a next plot the prediction performance measure, the 20% trimmed SEP, is shown:

```
> plotSEPprm(res.prmdcv, res.prmdcv$afinal, y, X)
```

The result of executing the above command is shown in Figure 9. The gray lines correspond to the results of the 20 repetitions of the double CV scheme,

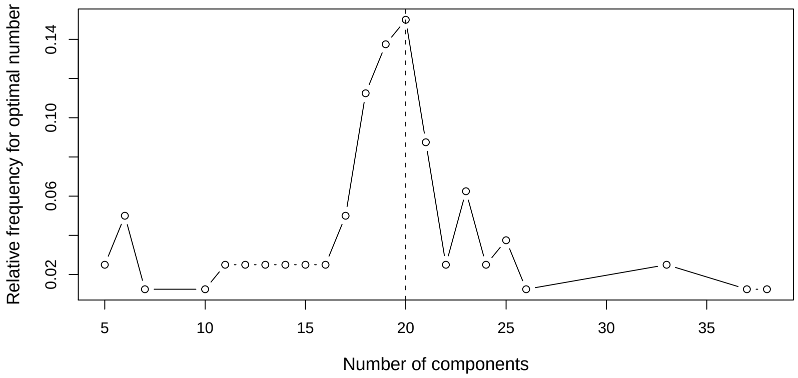

Figure 8: Output of `plotcompprm` for rdCV of PRM. The optimal number of components is indicated by the vertical dashed line.

while the black line represents the single CV result. Obviously, single CV is much more optimistic than rdCV. The estimated prediction error for 20 components is 5.86 (±0.9).

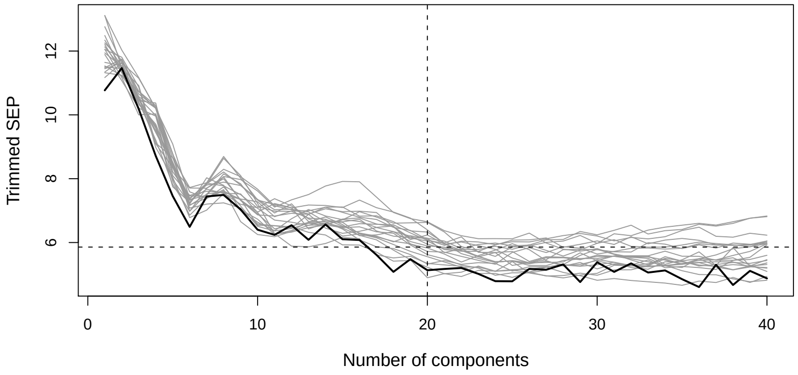

Figure 9: Output of `plotSEPprmdcv` for PRM. The gray lines result from repeated double CV, the black line from single CV.

Using the optimal number of 20 components, predictions and residuals can be computed. However, for rdCV there are predictions and residuals available for each replication (we used 20 replications). The diagnostic plot

```
> plotpredprm(res.prmdcv, res.prmdcv$afinal, y, X)
```

shows the predicted versus measured response values, see Figure 10. The left picture is the prediction from a single CV, while in the right picture the resulting predictions from rdCV are shown. The latter plot gives a clearer picture of the prediction uncertainty.

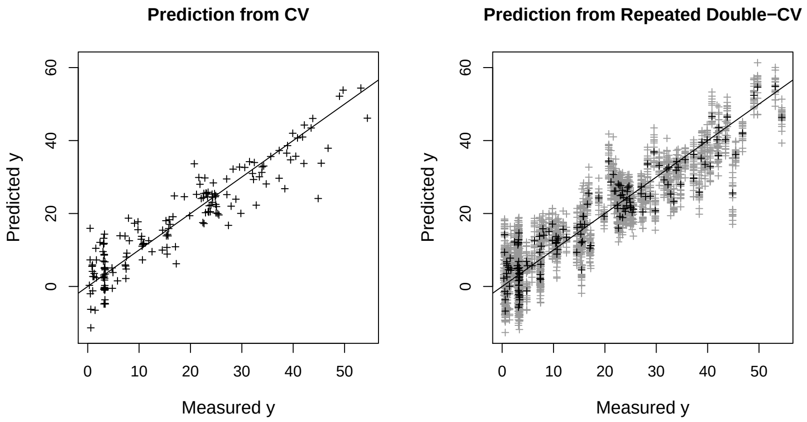

Figure 10: Predicted versus measured response values as output of `predprmdcv` for PRM. The left picture shows the results from single CV, the right picture visualizes the results from repeated double CV.

A similar plot can be generated with

```
> plotresprm(res.prmdcv, res.prmdcv$afinal, y, X)
```

for predicted values versus residuals (not shown here).

The package **chemometrics** has also implemented R functions to perform single CV and rdCV for classical PLS. Using the same parameters for rdCV, classical PLS requires 14 components, and results in a prediction error of 6.52. Thus, although this example data set does not contain huge or visible outliers but probably only slight inhomogeneities, the robust approach leads to an improved prediction model. Further details are provided in the vignette to the package.

## 5. Robust Principal Component Analysis (PCA)

Principal component analysis (PCA) is a widely used technique for dimension reduction achieved by finding a smaller number $k$ of linear combinations of the originally observed $p$ variables and retaining most of the variability of the data. These new variables, referred to as *principal components* are uncorrelated with each other and account for decreasing amount of the total variance, i.e. the first principal component explains the maximum variance in the data, the second principal component explains the maximum variance in the data that has not been explained by the first principal component and so on. Dimension reduction by PCA is mainly used for: (i) visualization of multivariate data by scatter plots (in a lower dimensional space); (ii) transformation of highly correlated variables into a smaller set of uncorrelated variables which can be used by other methods (e.g. multiple or multivariate regression); (iii) combination of several variables characterizing a given process into a single or a few *characteristic* variables or *indicators*.

The classical approach to PCA measures the variability through the empirical variance and is essentially based on computation of eigenvalues and eigenvectors of the sample covariance or correlation matrix. Therefore the results may be extremely sensitive to the presence of even a few atypical observations in the data. The outliers could artificially increase the variance in an otherwise uninformative direction and this direction will be determined as a PC direction. These discrepancies will carry over to any subsequent analysis and to any graphical display related to the principal components such as the biplot.

The following example shown in Figure 11 illustrates the effect of outliers on classical PCA. We generate $n = 60$ observations of two variables $x_1$ and $x_2$ from a bivariate normal distribution with zero means, variances of 1, and a correlation between the two variables 0.8. The sample correlation of the generated data set is 0.84. We sort the data by the first coordinate $x_1$ and modify the first four observations with smallest $x_1$ and the last four with largest $x_1$ by interchanging their first coordinates. Thus (less than) 15% of outliers are introduced which are undistinguishable on the univariate plots of the data. However, the sample correlation changes even its sign and becomes -0.05. The upper left panel (a) of Figure 11 shows a scatter plot of the clean data with the first principal component PC1, and the upper right panel (b) shows the same for the altered data. We see that the first principal component is tilted by the outliers in an almost perpendicular direction. The lower left panel (c) shows the plot of the scores on the two classical principal components. Most of the outliers lie within the 97.5% tolerance ellipse and thus they are influential on the classical covariance estimation. The lower right panel (d) shows the same plot based on robust estimates. We see that the estimate of the center remains the same as the classical one (as should be expected since we have not changed the values of the single variables) but the outliers are clearly separated by the 97.5% tolerance ellipse. In terms of dimension reduction, the results from classical PCA are not useful, because neither PC1 nor PC2 follow the main data structure but they are affected by the outliers. In contrast, robust PC1 indicates the direction of the data majority, and robust PC2 reveals the outliers.

Consider an $n \times p$ data matrix $\boldsymbol{X}$. Further, $\boldsymbol{m}$ denotes the (robust) center of the data and $\mathbf{1}$ is a column vector with all $n$ components equal to 1. We are looking for linear combinations $\boldsymbol{t}_j$ that result from a projection of the centered data on a direction $\boldsymbol{p}_j$,

$$\boldsymbol{t}_j = (\boldsymbol{X} - \mathbf{1}\boldsymbol{m}^t)\boldsymbol{p}_j \tag{20}$$

such that

$$\boldsymbol{p}_j = \underset{\boldsymbol{p}}{\operatorname{argmax}} \operatorname{Var}(\boldsymbol{X}\boldsymbol{p}) \tag{21}$$

subject to $\|\boldsymbol{p}_j\| = 1$ and $\operatorname{Cov}(\boldsymbol{X}\boldsymbol{p}_j, \boldsymbol{X}\boldsymbol{p}_l) = 0$ for $l < j$ and $j = 1, \ldots, k$ with $k \leq \min(n, p)$. The solutions of these maximization problems are obtained by solving a Lagrangian problem, and the result is that the principal components of $\boldsymbol{X}$ are the eigenvectors of the covariance matrix $\operatorname{Cov}(\boldsymbol{X})$, and the variances are the corresponding eigenvalues $l_j = \operatorname{Var}(\boldsymbol{X}\boldsymbol{p}_j)$. Classical PCA is obtained if the sample covariance matrix $\boldsymbol{S} = \frac{1}{n-1}\sum_{i=1}^{n}(\boldsymbol{x}_i - \boldsymbol{m})(\boldsymbol{x}_i - \boldsymbol{m})^t$ is used for “Cov”, with $\boldsymbol{m}$ being the arithmetic mean vector. PCA based on robust covariance estimation will be discussed in Section 5.1. Here, not only “Cov” but also the data center $\boldsymbol{m}$ need to be estimated robustly. Usually the eigenvectors are sorted in a decreasing order of the eigenvalues and hence the first $k$ principal components are the most important ones in terms of explained variance. A more general and usually recommended algorithm for PCA is *singular value decomposition* (SVD), for further details see Jolliffe [19]. However, SVD is not straightforward to robustify, see Croux et al. [20]. Finally, the vectors $\boldsymbol{t}_j$ are collected as columns in the $n \times k$ *scores* matrix $\boldsymbol{T}$, and the vectors $\boldsymbol{p}_j$ as columns in the *loadings* matrix $\boldsymbol{P}$. The eigenvalues $l_j$

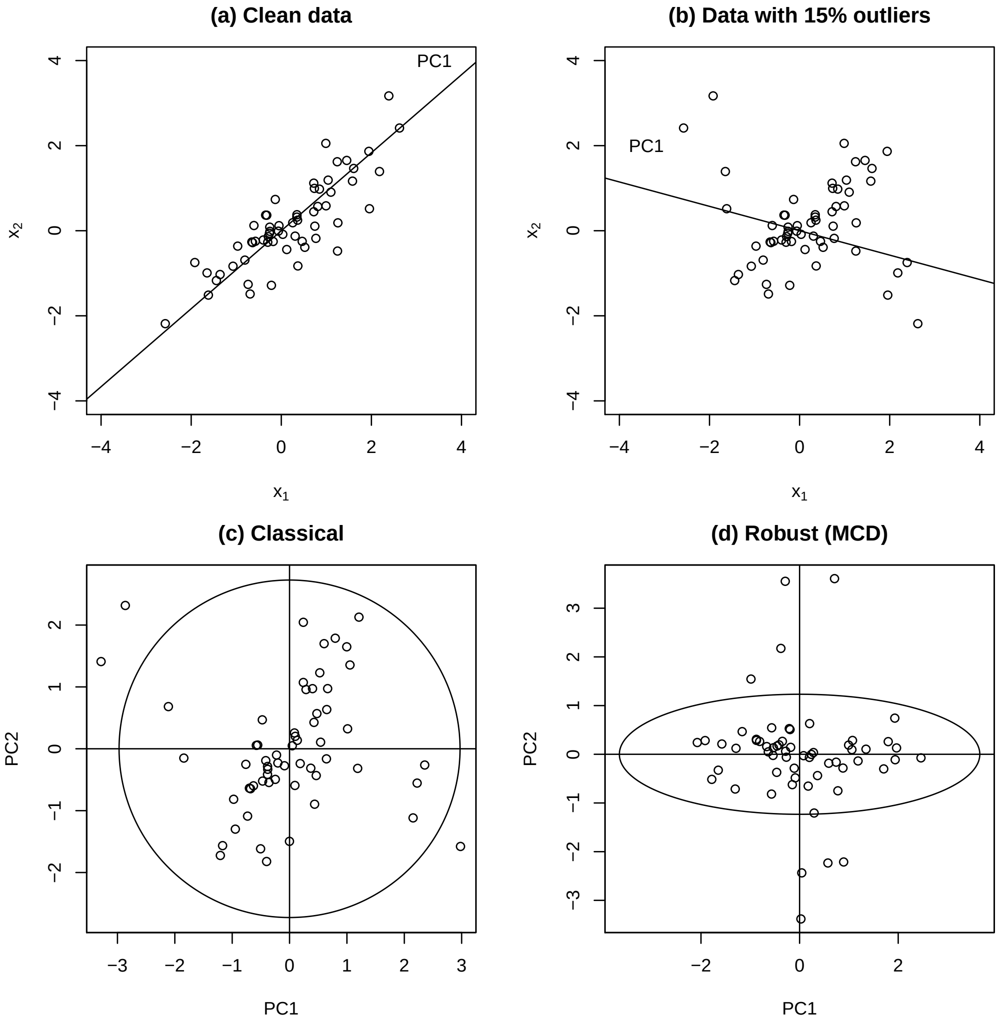

Figure 11: Plot of the principal components of the generated data: the upper two panels show scatter plots of the clean (a) and the altered (b) data with the first principal component. The lower two panels show plots of the scores obtained by classical (c) and robust (d) PCA together with the corresponding 97.5% tolerance ellipses.

are arranged in the diagonal of the $k \times k$ diagonal matrix $\boldsymbol{\Lambda}$. This allows to represent the covariance matrix as

$$\mathrm{Cov}(\boldsymbol{X}) = \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}^t. \tag{22}$$

The original $\boldsymbol{X}$ matrix can be reconstructed from the scores $\boldsymbol{T}$ in the original coordinate system (using $k$ principal components) preserving the main structure of the data:

$$\tilde{\boldsymbol{X}} = \mathbf{1}\boldsymbol{m}^t + \boldsymbol{T}\boldsymbol{P}^t + \boldsymbol{E}, \tag{23}$$

where the error or residual matrix $\boldsymbol{E}$ will be zero if all principal components are used.

PCA was probably the first multivariate technique subjected to robustification, either by simply computing the eigenvalues and eigenvectors of a robust estimate of the covariance matrix or directly by estimating each principal component in a robust manner. The different approaches to robust PCA are presented in the next sections and examples are given how these robust analyses can be carried out in R. Details about the methods and algorithms can be found in the corresponding references.

### *5.1. PCA based on robust covariance matrix estimation*

The most straightforward and intuitive method to obtain robust PCA is to replace the classical estimates of location and covariance by their robust analogues. In the earlier works, M estimators of location and scatter were used for this purpose [see 21, 22] but these estimators have the disadvantage of low breakdown point in high dimensions. To cope with this problem, the MVE estimator [23] and the MCD estimator [24] were used. Croux and Haesbroeck [25] investigated the properties of the MCD estimator and computed its influence function and efficiency.

The package **stats** in base R contains the function `princomp()` which performs a principal components analysis on a given numeric data matrix and returns the results as an object of S3 class `princomp`. This function has a parameter `covmat` which can take a covariance matrix, or a covariance list as returned by `cov.wt`, and if supplied, it is used rather than the covariance matrix of the input data. This allows to obtain robust principal components by supplying the covariance matrix computed by `cov.mve` or `cov.mcd` from the package **MASS**. Much easier and more flexible is the interface provided in package **rrcov**. The essential value added of this package, apart from implementing many new robust multivariate methods, is the unification of the interfaces by leveraging the object orientation provided by the S4 classes and methods. The function `PcaCov()` computes robust PCA by replacing the classical covariance matrix with one of the robust covariance estimators available in the framework—MCD, OGK, MVE, M, MM, S or Stahel-Donoho [for details, see 26], i.e., the parameter `cov.control` can be any object of a class derived from the base class `CovControl`. This control class will be used to compute a robust estimate of the covariance matrix. If this parameter is omitted, MCD will be used by default. Of course any newly developed estimator following the concepts of the framework can be used as input to the function `PcaCov()`.

```
> library(rrcov)
> pc <- PcaCov(X)             # X is the input data matrix
> P  <- getLoadings(pc)       # robust PCA loadings
> T  <- getScores(pc)         # robust PCA scores
```

Unfortunately, while highly robust and very intuitive, this method is limited to relatively low-dimensional data. For example, the MCD estimator is defined as the mean and covariance matrix of the $h$ observations whose covariance matrix has smallest determinant (here $h$ is roughly $n/2$). If $h \geq p$, where $p$ is the number of variables, the MCD is not defined since the determinant of any $h$-subset will have determinant zero. This condition could be weakened by setting $h$ to a larger value, e.g. $h = 0.75n$, but in any case we must have $p \leq n$. An additional limitation is the fact that all high breakdown point estimators of location and covariance matrix (with the appropriate equivariance properties) are computationally intensive and even the best available algorithms cannot handle very large data sets with dimensions in the number of thousands as they are often encountered in chemometrics.

### *5.2. PCA based on projection pursuit*

The second approach to robust PCA uses *projection pursuit* (PP) and calculates directly the robust estimates of the eigenvalues and eigenvectors without passing by a robust covariance estimation. Directions are sought for, which maximize the variance of the data projected onto them. The advantage of this approach is that the principal components can be computed sequentially, and that one can stop after $k$ components have been extracted. Thus, this approach is appealing for high-dimensional data, in particular for problems with $p > n$.

Using the empirical variance in the maximization problem would lead to classical PCA, and robust scale estimators result in robust PCA. Such a method was first introduced by Li and Chen [27] using an M estimator of scale, see Equation (9). They showed that the PCA estimates inherit the robustness properties of the scale estimator. Unfortunately, in spite of the good statistical properties of the method, the algorithm they proposed was too complicated to be used in practice. A more tractable algorithm in these lines was proposed by Croux and Ruiz-Gazen [28], and they also completed the theoretical results of Li and Chen [27] by computing the influence functions of the estimators of the eigenvalues, eigenvectors and associated covariance matrix as well as by computing the asymptotic variances. This algorithm centers the data matrix with the spatial median or $L_1$–median, which is fast to compute and has a 50% breakdown point [see, e.g., 29]. When solving the maximization problem the algorithm does not investigate all possible directions but considers only those defined by a data point and the robust center of the data. The robust variance estimate is computed for the data points projected on these $n$ directions and the direction corresponding to the maximum of the variance is the searched approximation of the first principal component. After that the search continues in the same way in the space orthogonal to the first component. An improved version of this algorithm, being more precise especially for high-dimensional data, was proposed by Croux et al. [30]. The space of all possible directions is scanned more thoroughly. This is done by restricting the search for an optimal direction on a regular grid in a plane.

The PCA projection pursuit algorithms Croux and Ruiz-Gazen [28] and Croux et al. [30] are represented in R by the classes `PcaProj` and `PcaGrid`, respectively. Their generating functions provide simple wrappers around the original functions from the package **pcaPP** and return objects of the corresponding class, derived from `PcaRobust`.

```
> pc <- PcaGrid(X, k=2, scale=mad)
>      # k=2 PCs are computed, MAD is the robust scale measure
> P  <- getLoadings(pc)        # robust PCA loadings
> T  <- getScores(pc)          # robust PCA scores
```

### *5.3. The method ROBPCA*

This robust PCA method proposed by Hubert et al. [31] tries to combine the advantages of both approaches—PCA based on a robust covariance matrix and PCA based on projection pursuit. A brief description of the algorithm follows, for details see the relevant references [32]. After robustly centering the data, an SVD is applied to express the information in the $n$-dimensional space (useful if $p > n$). Then for each observation a measure of “outlyingness” is computed. The $h$ data points with smallest outlyingness measure are used to compute the robust covariance matrix and to select the number $k$ of principal components to retain. With an eigendecomposition of this covariance matrix, the space spanned by the first $k$ eigenvectors is used to project all data points. Finally, location and covariance of the projected data are computed using the reweighted MCD estimator, and the eigenvectors of this scatter matrix yield the robust principal components.

The algorithm ROBCA is implemented in the R package (rrcov) as the function `PcaHubert()`.

### *5.4. Spherical principal components*

The spherical principal components procedure was first proposed by Locantore et al. [33] as a method for functional data analysis. The idea is to perform classical PCA on the data, projected onto a unit sphere. The estimates of the eigenvectors are consistent if the data are elliptically distributed [see 34] and the procedure is extremely fast. Although not much is known about the efficiency of this method, the simulations of Maronna [35] show that it has very good performance. If each coordinate of the data is normalized using some kind of robust scale, like for example the MAD, and then spherical principal component analysis is applied, we obtain “elliptical PCA”, but unfortunately this procedure is not consistent. To compute the PCA estimates by the ROBPCA method in R, the function `PcaLocantore()` is used.

### *5.5. Visualization of PCA results and diagnostic plots*

The results of all PCA methods implemented in the R package (rrcov) can be visualized using exactly the same plotting functions. The *screeplot*, comparing the variances of the principal components [see, e.g., 7] can be visualized by

```
> screeplot(pc, type="lines")
```

where `pc` is the result of a PCA method. The *biplot* [36] represents both the observations and variables in the plane of (the first) two principal components allowing the visualization of the magnitude and sign of each variable’s contribution to these principal components. This plot is generated by:

```
> biplot(pc)
```

Besides scatter plots of the first few principal components that allow to reveal the multivariate structure of the data and to discover data groups and structures, a *diagnostic plot* is especially useful for identifying outlying observations. The diagnostic plot is based on the *score distances* and *orthogonal distances* computed for each observation. Note that in chemometrics, the score distance is also known under the name Hotelling's $T^2$, and the orthogonal distance under the abbreviation $Q$. The *score distance* is defined by

$$SD_i = \sqrt{\sum_{j=1}^{k} \frac{t_{ij}^2}{l_j}}, \qquad i = 1, \ldots, n, \tag{24}$$

where $t_{ij}$ are the elements of the score matrix $\boldsymbol{T}$. It measures the distance of each observation to the subspace spanned by the first $k$ principal components. The *orthogonal distance* is defined by

$$OD_i = ||\boldsymbol{x}_i - \boldsymbol{m} - \boldsymbol{P}\boldsymbol{t}_i||, \qquad i = 1, \ldots, n \tag{25}$$

where $\boldsymbol{t}_i$ is the $i$th row of the score matrix $\boldsymbol{T}$. This measure corresponds to the distance of the projection of each observation into the space spanned by the first $k$ principal components. The *diagnostic plot* shows the score versus the orthogonal distance, and indicates with a horizontal and vertical line the cut-off values that allow to distinguish regular observations from the two types of outliers [for details, see 31].

### *5.6. Example: Robust PCA in R*

We consider a data set originating from 180 glass vessels [37]. In total, 1920 characteristics are available for each vessel, coming from an analysis by an electron-probe X-ray micro-analysis. The data set includes four different materials comprising the vessels, and we focus on the material forming the larger group of 145 observations. Columns with MAD equal to zero were removed, resulting in a matrix with 1905 columns. It is known from other studies on this data set [15, 38] that these 145 observations should form two groups, because during the measurement process the detector efficiency has been changed. So, in principle PCA should reveal the two clouds of data points. We assume that the package **rrcov** has been loaded and that `X` contains the data. Depending on the PCA algorithm, $k = 2$ to $k = 4$ components explain more than 90% of the data variability. For reasons of comparability we use $k = 4$ for all methods.

```
> pcC <- PcaClassic(X, k=4)              # classical PCA
> pcG <- PcaGrid(X, k=4)                 # see Section 5.2
> pcH <- PcaHubert(X, k=4,alpha=0.5)     # see Section 5.3
> pcL <- PcaLocantore(X, k=4)            # see Section 5.4
```

Details of the PCA results can be seen with the function `summary()` applied to the result objects.

A scatter plot of the first two PCA scores (first two columns of $\boldsymbol{T}$) can be seen with

```
> rrcov:::pca.scoreplot(pcC, main = "(a) Classical PCA")
> rrcov:::pca.scoreplot(pcG, main = "(b) PCA based on PP")
> rrcov:::pca.scoreplot(pcH, main = "(c) ROBPCA")
> rrcov:::pca.scoreplot(pcL, main = "(d) Spherical PCA")
```

and the results are shown in Figure 12. All PCA methods show the two data groups, but additionally several other inhomogeneities are visible. PCA based on PP (b) clearly shows three data groups.

The diagnostic plots mentioned in Section 5.5 are shown for the resulting PCA objects by

```
> plot(pcC, main = "(a) Classical PCA")
> plot(pcG, main = "(b) PCA based on PP")
> plot(pcH, main = "(c) ROBPCA")
> plot(pcL, main = "(d) Spherical PCA")
```

which gives the plots in Figure 13. The symbols used in the plots refer to the information from other studies [15, 38] if an observation was identified as outlier (×) or not (○). It can be seen that classical PCA (a) shows some outliers, but also regular observations are declared as outliers. In addition, there is no clear grouping structure visible. The robust PCA methods all show two groups and in addition some deviating data points. PCA based on PP using the algorithm of Croux et al. [30] clearly flags the group with the different detector efficiency as outliers in terms of both the orthogonal and the score distance. ROBPCA finds almost the same answer, but the contrast between the two groups is not as clear as for PP-based PCA. For the result of spherical PCA, the score distance is reliable but the orthogonal distance is misleading.

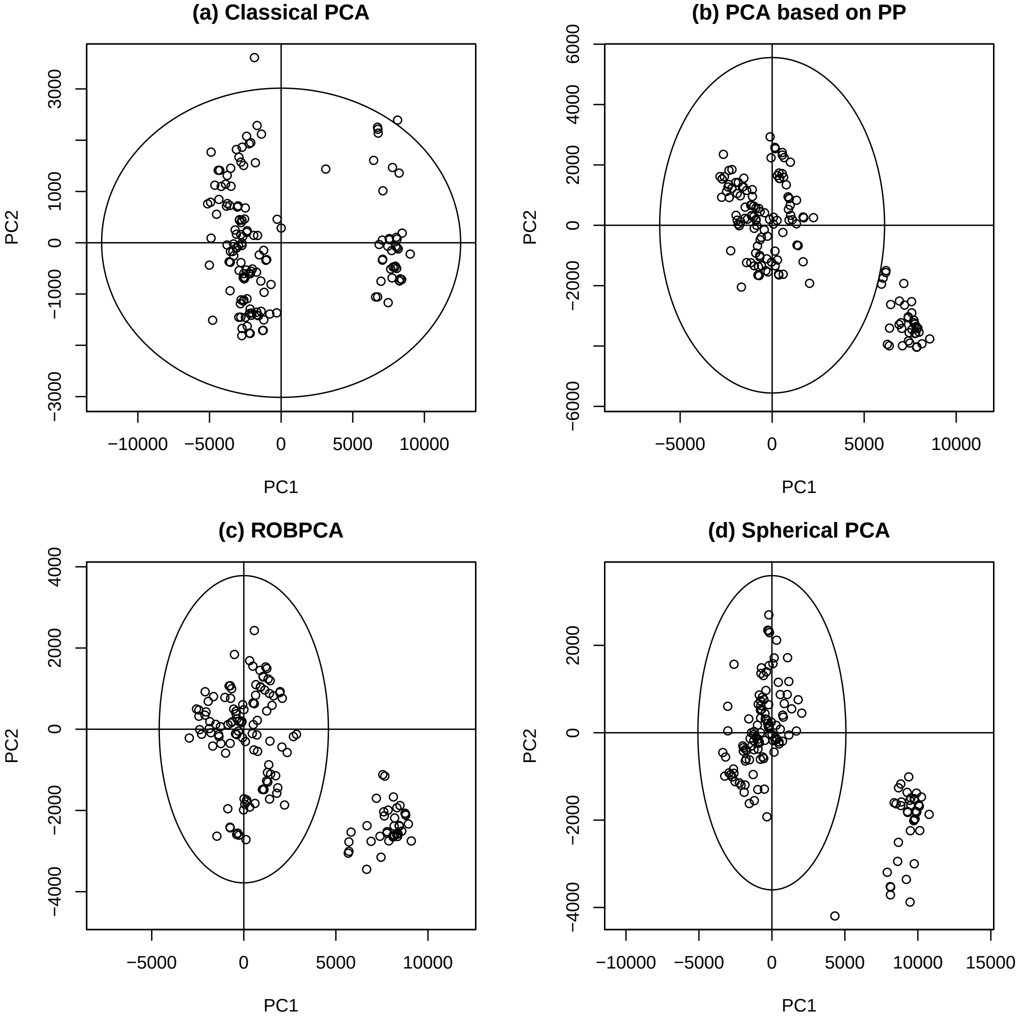

Figure 12: Plots of the first two PCA scores for classical PCA (a), and robust PCA based on projection pursuit (b), for the ROBCA algorithm (c), and for spherical PCA (d).

## 6. Conclusions

Robust statistical methods are available for various purposes needed especially in chemometrics: for dimension reduction, for modeling and model evaluation, for outlier detection, etc. Robust methods focus on modeling the data majority, and they downweight deviating or outlying observations. Downweighting with weights of zero and one would correspond to omitting outly-

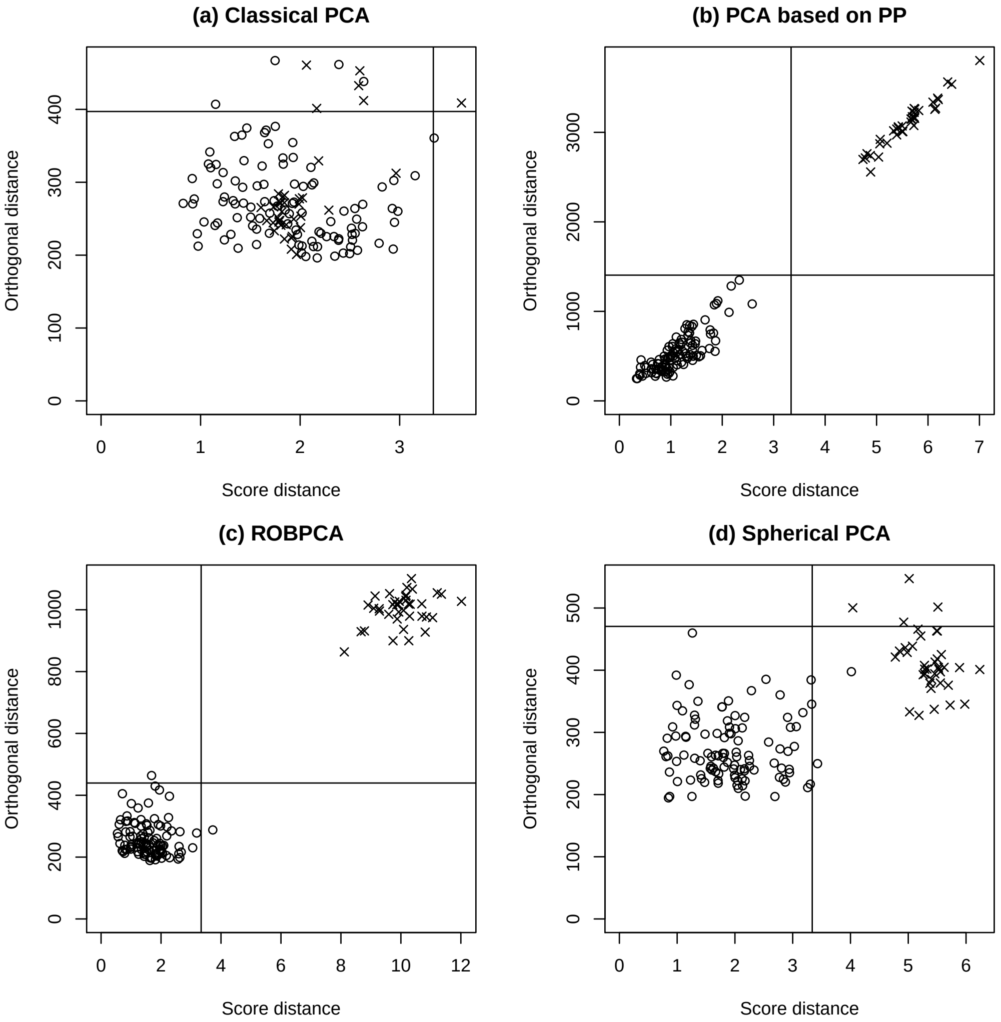

Figure 13: Diagnostic plots for classical PCA (a), and robust PCA based on projection pursuit (b), for the ROBCA algorithm (c), and for spherical PCA (d).

ing observations from the analysis. However, especially for high-dimensional data this would result in a severe loss of information as long as the outliers still contain some valuable information, and thus “intelligent” robust methods adapt the weights according to the outlyingness or inconsistency of the observations. This allows to increase the statistical efficiency of the estimator, leading to more precise results and to better models.

All methods discussed in this contribution are available in the statistical software environment R, freely available at `http://cran.r-project.org/`. The main functions discussed here are implemented in the packages **chemometrics** of Filzmoser and Varmuza [39] and **rrcov** of Todorov [40], which can also be downloaded from the above web page. One aim of this contribution was to demonstrate how the functions can actually be used, which results are produced, and how they can be interpreted. However, they include many more possibilities than mentioned here, and we refer to the package documentations for an overview. Methods for analyzing high-dimensional data are available only recently, and it is work in progress to include functions for other purposes, like discriminant analysis.

### Acknowledgements

We would like to thank two anonymous referees for their valuable comments and suggestions.

### References

[1] F. Hampel, E. Ronchetti, P. Rousseeuw, W. Stahel, Robust Statistics. The Approach Based on Infuence Functions, John Wiley & Sons, 1986.

[2] R. Maronna, D. Martin, V. Yohai, Robust Statistics: Theory and Methods, John Wiley & Sons, New York, 2006.

[3] V. Barnett, T. Lewis, Outliers in Statistical Data, John Wiley & Sons, 1994.

[4] S. Frosch-Møller, J. von Frese, R. Bro, Robust methods for multivariate data analysis, Journal of Chemometrics 19 (2005) 549–563.

[5] R Development Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2010. ISBN 3-900051-07-0.

[6] R. Johnson, D. Wichern, Applied Multivariate Statistical Analysis, Prentice Hall, International, 2002. Fifth edition.

[7] K. Varmuza, P. Filzmoser, Introduction to Multivariate Statistical Analysis in Chemometrics, CRC Press, Boca Raton, FL, 2009.

[8] P. Huber, Robust Statistics, Wiley & Sons, New York, 1981.

[9] P. Huber, E. Ronchetti, Robust Statistics, Wiley & Sons, New York, 2009.

[10] P. Huber, Robust estimation of a location parameter, The Annals of Mathematical Statistics 35 (1964) 73–101.

[11] P. Rousseeuw, A. Leroy, Robust Regression and Outlier Detection, John Wiley & Sons, New York, 2003.

[12] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning. Data Mining, Inference, and Prediction, Springer-Verlag, New York, second edition, 2009.

[13] J. Khan, S. Van Aelst, R. Zamar, Robust linear model selection based on least angle regression, Journal of the American Statistical Association 102 (2007) 1289–1299.

[14] M. Hubert, K. Vanden Branden, Robust methods for partial least squares regression, Journal of Chemometrics 17 (2003) 537–549.

[15] S. Serneels, C. Croux, P. Filzmoser, P. Espen, Partial robust M-regression, Chemometrics and Intelligent Laboratory Systems 79 (2005) 55–64.

[16] B. Liebmann, A. Friedl, K. Varmuza, Determination of glucose and ethanol in bioethanol production by near infrared spectroscopy and chemometrics, Anal. Chim. Acta 642 (2009) 171–178.

[17] P. Filzmoser, B. Liebmann, K. Varmuza, Repeated double cross validation, Journal of Chemometrics 23 (2009) 160–171.

[18] B. Liebmann, P. Filzmoser, K. Varmuza, Robust and classical pls regression compared, Journal of Chemometrics 24 (2010) 111–120.

[19] I. Jolliffe, Principal Component Analysis, Springer, New York, 2002.

[20] C. Croux, P. Filzmoser, G. Pison, P. Rousseeuw, Fitting multiplicative models by robust alternating regressions, Statistics and Computing 13 (2003) 23–36.

[21] S. Devlin, R. Gnanadesikan, J. Kettenring, Robust estimation of dispersion matrices and principal components, Journal of the American Statistical Association 76 (1981) 354–362.

[22] N. Campbell, Procedures in multivariate analysis I: Robust covariance estimation, Applied Statistics 29 (1980) 231–237.

[23] R. Naga, G. Antille, Stability of robust and non-robust principal component analysis, Computational Statistics & Data Analysis 10 (1990) 169–174.

[24] V. Todorov, N. Neykov, P. Neytchev, Stability of (high-breakdown point) robust principal components analysis, in: R. Dutter, W. Grossmann (Eds.), Short Communications in Computational Statistics, COMPSTAT 1994, Physica Verlag, Heidelberg, 1994, pp. 90–92.

[25] C. Croux, G. Haesbroeck, Principal components analysis based on robust estimators of the covariance or correlation matrix: Influence functions and efficiencies, Biometrika 87 (2000) 603–618.

[26] V. Todorov, P. Filzmoser, An object-oriented framework for robust multivariate analysis, Journal of Statistical Software 32 (2009) 1–47.

[27] G. Li, Z. Chen, Projection-pursuit approach to robust dispersion matrices and principal components: Primary theory and Monte Carlo, Journal of the American Statistical Association 80 (1985) 759–766.

[28] C. Croux, A. Ruiz-Gazen, High breakdown estimators for principal components: The projection-pursuit approach revisited, Journal of Multivariate Analysis 95 (2005) 206–226.

[29] H. Fritz, P. Filzmoser, C. Croux, A comparison of algorithms for the multivariate $L_1$-median, Technical Report CS-2010-4, Dept. of Statistics and Probability Theory, Vienna Univ. of Technology, Austria, 2010. Submitted for publication.

[30] C. Croux, P. Filzmoser, M. Oliveira, Algorithms for projection-pursuit robust principal component analysis, Chemometrics and Intelligent Laboratory Systems 87 (2007) 218–225.

[31] M. Hubert, P. Rousseeuw, K. Vanden Branden, ROBPCA: A new approach to robust principal component analysis, Technometrics 47 (2005) 64–79.

[32] M. Hubert, P. Rousseeuw, S. van Aelst, High-breakdown robust multivariate methods, Statistical Science 23 (2008) 92–119.

[33] N. Locantore, J. Marron, D. Simpson, N. Tripoli, J. Zhang, K. Cohen, Robust principal components for functional data, Test 8 (1999) 1–28.

[34] G. Boente, R. Fraiman, Discussion of 'robust principal components for functional data' by locantore et al., Test 8 (1999) 1–28.

[35] R. Maronna, Principal components and orthogonal regression based on robust scales, Technometrics 47 (2005) 264–273.

[36] K. Gabriel, The biplot graphical display of matrices with application to principal component analysis, Biometrika 58 (1971) 453–467.

[37] K. Janssens, I. Deraedt, A. Freddy, J. Veeckman, Composition of 15-$17^{th}$ century archæological glass vessels excavated in Antwerp, Belgium, Mikrochimica Acta 15 (Suppl.) (1998) 253–267.

[38] P. Filzmoser, R. Maronna, M. Werner, Outlier identification in high dimensions, Computational Statistics and Data Analysis (2008) 1694–1711.

[39] P. Filzmoser, K. Varmuza, **chemometrics**: Multivariate Statistical Analysis in Chemometrics, 2010. R package version 1.2.

[40] V. Todorov, **rrcov**: Robust Location and Scatter Estimation and Robust Multivariate Analysis with High Breakdown Point, 2010. R package version 1.2-00.