Partial Robust M-Regression

Sven Serneels[1] * Christophe Croux[2] Peter Filzmoser[3]

Pierre J. Van Espen[1]

[1]Department of Chemistry, University of Antwerp, Belgium

[2] Department of Applied Economics, K.U. Leuven, Belgium

[3] Department of Statistics and Probability Theory,

Vienna University of Technology, Austria

*Correspondence to S. Serneels, Departement Scheikunde, Universiteit Antwerpen, Universiteitsplein 1, 2610 Antwerpen (Belgium) E-mail: sven.serneels@ua.ac.be. Tel.: +32/3/8202378; Fax.: +32/3/8202376

Partial Robust M-Regression

Abstract

Partial Least Squares (PLS) is a standard statistical method in chemometrics. It can be considered as an incomplete, or "partial", version of the Least Squares estimator of regression, applicable when high or perfect multicollinearity is present in the predictor variables. The Least Squares estimator is well-known to be an optimal estimator for regression, but only when the error terms are normally distributed. In absence of normality, and in particular when outliers are in the data set, other more robust regression estimators have better properties. In this paper a "partial" version of M-regression estimators will be defined. If an appropriate weighting scheme is chosen, partial M-estimators become entirely robust to any type of outlying points, and are called Partial Robust M-estimators. It is shown that partial robust M-regression outperforms existing methods for robust PLS regression in terms of statistical precision and computational speed, while keeping good robustness properties. The method is applied to a data set consisting of EPXMA spectra of archæological glass vessels. This data set contains several outliers, and the advantages of partial robust M-regression are illustrated. Applying partial robust M-regression yields much smaller prediction errors for noisy calibration samples than PLS. On the other hand, if the data follow perfectly well a normal model, the loss in efficiency to be paid for is very small.

Keywords: calibration, partial least squares, M-estimators, prediction, outliers, robustness, spectrometric quantization.

1 Introduction

Partial Least Squares (PLS) [1] is a very widely used statistical tool. Its major benefit over other techniques is mainly apparent if applied to datasets consisting of blocks of variables of which at least one is subject to problems such as multicollinearity or the number of variables exceeding the number of observations at hand. For this type of datasets, PLS yields stable estimates which

can be applied to unveil a presumed latent structure in the data (the so called PLS approach), and to predict one of the blocks of variables from the other block (PLS regression). PLS has been developed keeping these objectives in mind and has become a major tool for data analysis. In chemometrics its use has become standard, for example for predicting the concentration of a chemical substance in UV-VIS or infrared spectrometry. In this paper the focus is on PLS regression, where a dependent variable needs to be predicted by a set of prediction variables. The underlying idea is that PLS summarizes the often high-dimensional predictor variables into a smaller set of uncorrelated, so-called *latent* variables, which have a maximal covariance to the predictand. Regressing the dependent variable on this set of latent variables is more stable, and this explains why PLS can be successful.

In regression analysis the Least Squares (LS) estimator has several optimality properties. One of them is that LS is the maximum likelihood estimator if the error terms follow a normal distribution. Hence LS is the most efficient estimator at regression models with normal error terms. However, there is no guarantee at all that normality applies: for example, the distribution of the error term may have heavy tails. For such data, the LS procedure may loose much of its power. Huber [2] introduced the class of M-estimators, having the property that they have good efficiency properties over a wide range of error-distributions. In this paper a *Partial M-estimator* (PM) will be proposed. Indeed, there is no reason why only a "partial" version of LS should exist, and not "partial" versions of other regression estimators. It will be shown that for many types of error distributions the partial M-estimator outperforms the partial LS-estimator in terms of efficiency, resulting in smaller prediction errors. An exception is the normal model, where there is a very small loss of precision for the PM-estimator.

Another important advantage of partial M-regression is its robustness with respect to outliers. The PLS method is known to be very sensible to outlying observations, which are typically expected to be present in experimental data. This drawback of classical partial least squares regression has been heeded by several authors who propose different ways to construct a robust

version of partial least squares regression. The first authors to propose a robustified PLS were Wakeling and MacFie [3] who replace all least squares regressions in the PLS algorithm by robust regressions. As the latter demand a high computational cost, the same statement holds for their entire method. Cummins and Andrews [4] propose a closely related technique, called *Iteratively Reweighted Partial Least Squares* (IRPLS) which is no longer prone to a high computational effort. However, the method has recently been criticized by Hubert and Vanden Branden [5] because it is non-resistant to leverage points (i.e. outliers in the space of predictor variables). This statement justifies their proposal of their own method, Robust SIMPLS (RSIMPLS), which requires more computation time, but is resistant to all types of outliers. The approach taken in this paper is conceptually different: instead of *robust partial* LS, a *partial robust* regression estimator is proposed. With an appropriately chosen weighing scheme, where leverage points are downweighted, partial M-regression becomes as robust as previously proposed methods and will be called *Partial Robust M* (PRM) regression.

The paper is organized as follows. Section 2 introduces partial M-estimators and partial robust M-estimators. Section 3 details an algorithm to compute them. It turns out that partial (robust) M-regression can be computed by a modification of the IRPLS algorithm of [4]. Partial robust M inherits the speed of computation of the IRPLS approach, but is robust to all types of outliers. Section 4 presents a simulation study, showing that partial robust M-regression outperforms its main competitors both in terms of statistical efficiency and in terms of computational cost. In Section 5 an application of partial robust M-regression to X-ray analysis of glass samples is presented, and Section 6 concludes.

2 Partial M-regression

2.1 M-estimators of regression

Before defining partial M-regression estimators, we set up the notation and review the definition of M-estimators in the standard regression setting. Let $\mathbf{X}$ be the data matrix of size $n \times p$ containing the predictor variables in its columns, and let $\boldsymbol{y}$ be the data vector of size $n \times 1$ containing the dependent variable. The i-th rows of $\mathbf{X}$ and $\boldsymbol{y}$, containing the information on the i-th sample, are denoted by $\boldsymbol{x}_i$ and y_i, respectively. Consider the regression model

$$y_i = \boldsymbol{x}_i\boldsymbol{\beta} + \varepsilon_i, \tag{1}$$

where the unknown regression parameter $\boldsymbol{\beta}$ is a column vector of length p, and the error terms are denoted by ε_i, for $1 \leq i \leq n$. The Least Squares (LS) estimator of $\boldsymbol{\beta}$ is defined as

$$\hat{\boldsymbol{\beta}}_{LS} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i\boldsymbol{\beta})^2, \tag{2}$$

and is known to be the optimal estimator (in the sense of having the smallest variance and being unbiased) if the error terms ε_i follow a normal distribution. However, if the error terms come from other distributions, e.g. heavy-tailed distributions, then LS looses its optimality and other types of estimators perform better. The most well-known robust estimators are M-estimators, obtained by replacing the squares in (2) by a more general *loss function* ρ:

$$\hat{\boldsymbol{\beta}}_{M} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho(y_i - \boldsymbol{x}_i\boldsymbol{\beta}). \tag{3}$$

The loss function ρ needs to be symmetric and non-decreasing. By taking $\rho(u) = u^2$, the LS-estimator appears as a special case. To give less importance to large residuals one chooses a bounded loss function ρ, resulting in more robust estimator than LS. Let $r_i = y_i - \boldsymbol{x}_i\boldsymbol{\beta}$ denote the residual in the objective function of (3) and define the weight attached to observation i as

$$w_i^r = \rho(r_i)/r_i^2. \tag{4}$$

Then one can rewrite (3) now as

$$\hat{\boldsymbol{\beta}}_M = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} w_i^r (y_i - \boldsymbol{x}_i \boldsymbol{\beta})^2. \tag{5}$$

In the above definition, the M-estimator is expressed as a weighted LS-estimator, but with weights depending on $\boldsymbol{\beta}$. This formulation allows the M-estimator to be computed with an iteratively reweighted least squares algorithm [6]. More information on M-estimators of regression can be found in [2]. A more applied textbook on robust regression methods is [7].

M-estimators of regression have been critiqued since they only give protection against *vertical outliers,* i.e. outliers in the error terms. Another type of outliers are *leverage points*, being observations $\boldsymbol{x}_i$ in the predictor space far away from the big majority of the data. To accommodate for these leverage points, the weight in (4) will be multiplied by a second weight w_i^x:

$$\hat{\boldsymbol{\beta}}_{RM} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} w_i^r \; w_i^x (y_i - \boldsymbol{x}_i \boldsymbol{\beta})^2. \tag{6}$$

Observations close to the centre of the data cloud in the predictor space will receive a weight w_i^x close to or equal to one, while leverage points will get a weight close to zero. The precise definition of the weights will be given in Section 3. If weights are used for both residuals and leverage points, as in (6), then we speak about a Robust M-estimator (RM). A robust M-estimators gives protection against both vertical outliers and leverage points.

2.2 Partial Robust M-estimators (PRM)

If the number of predictors p is large relative to the sample size n, a more convenient model is the *latent variables regression model.* The idea is that it is sufficient to regress the dependent variable on a limited number of h latent variables. The values of these latent variables are put together in the score matrix $\mathbf{T}$, of size $n \times h$, having as rows the vectors $\boldsymbol{t}_i$, with $1 \leq i \leq n$. The latent regression model is then given by:

$$y_i = \boldsymbol{t}_i \boldsymbol{\gamma} + \varepsilon_i. \tag{7}$$

Since the dimension of $\boldsymbol{\gamma}$ is low, namely h, the vector $\boldsymbol{\gamma}$ can be estimated as before by regressing the dependent variable on the latent variables by means of a robust M-estimator. The main difference is that the weights w_i^r are now computed from the residuals $r_i = y_i - \boldsymbol{t}_i\boldsymbol{\gamma}$, and the weights w_i^x for downweighting leverage points will be computed from the scores $\boldsymbol{t}_i$, instead of from the original explicative variables. If one only downweights for large residuals, that is $w_i = w_i^r$, the resulting estimator is the Partial M-estimator (PM). On the other hand, if one wants to protect against presence of possible leverage points, weights need to be taken as

$$w_i = w_i^r \; w_i^x,$$

and the resulting estimator will be called the Partial Robust M-estimator (PRM).

A remaining issue is to obtain the score matrix $\mathbf{T}$, which is not directly observed. Herefore the following scheme will be used. Loading vectors $\boldsymbol{a}_k$, for $k = 1, \ldots, h$ are obtained in a sequential manner as

$$\boldsymbol{a}_k = \underset{\boldsymbol{a}}{\operatorname{argmax}} \operatorname{Cov}_W(\boldsymbol{y}, \mathbf{X}\boldsymbol{a}) \tag{8a}$$

under the constraint that

$$\|\boldsymbol{a}\| = 1 \quad \text{and} \quad \operatorname{Cov}_W(\mathbf{X}\boldsymbol{a}, \mathbf{X}\boldsymbol{a}_j) = 0 \quad \text{for} \quad 1 \leq j < k. \tag{8b}$$

Above, $\operatorname{Cov}_W(\boldsymbol{y}, \boldsymbol{u})$, with $\boldsymbol{u}$ another vector of length n, stands for a weighted covariance

$$\operatorname{Cov}_W(\boldsymbol{y}, \boldsymbol{u}) = \frac{1}{n} \sum_{i=1}^{n} w_i y_i u_i.$$

After determining the loading vectors, they are collected in the columns of a $p \times h$ matrix $\mathbf{A}$, and the score matrix is given by $\mathbf{T} = \mathbf{X}\mathbf{A}$. Once $\hat{\boldsymbol{\gamma}}$ is obtained, the final estimate for $\boldsymbol{\beta}$ follows immediately as $\hat{\boldsymbol{\beta}} = \mathbf{A}\hat{\boldsymbol{\gamma}}$.

Note that PLS appears as a special case if all weights w_i are taken equal, resulting in a non-robust estimator. If we pretend the weights to be fixed, then it is not hard to see that $\hat{\boldsymbol{\gamma}}$ is nothing else but the PLS-estimator computed from the weighted observations $(\sqrt{w_i}\boldsymbol{x}_i, \sqrt{w_i}y_i)$. But the weights in the above definitions depend on unknown quantities and are not fixed. The idea is to

use an appropriate starting value for the weights, from which a first approximation of the estimator $\hat{\gamma}$ can be computed using PLS with fixed weights. Then the weights are recomputed using the preliminary parameter estimates, and a second approximation of $\hat{\gamma}$ is obtained by again applying weighted PLS. Afterwards the weights w_i are recomputed, and one continues the iteration process. Hence, the Iterative Reweighted Partial Least Squares (IRPLS) algorithm can be used to compute $\hat{\gamma}$ and more details are presented in the next Section.

3 Algorithm

Implementation of partial robust M-regression boils down to writing an iterative reweighted partial least squares algorithm. It will be crucial to use robust starting values and carefully chosen weights. These two aspects were overlooked in the original IRPLS paper [4]. Their method is not resistant to leverage points [5], because the weights they use *only* depend on the residuals after each step. For partial robust M-regression, the weights also have to depend on the scores, hereby correcting for leverage points if present in the predictor space. Another crucial difference is that the starting values for the IRPLS algorithm suggested in [4] are non-robust. Since the objective function defining the partial (robust) M-estimator may have local minima, starting from non-robust initial values induces the risk of converging to a local minimum corresponding to a non-robust estimate.

In our implementation of the algorithm the weights w_i^r in (4) have been computed as

$$w_i^r = f(\frac{r_i}{\hat{\sigma}}, c) \tag{9}$$

with $\hat{\sigma}$ an estimate of residual scale and

$$f(z, c) = \frac{1}{\left(1 + \left|\frac{z}{c}\right|\right)^2}, \tag{10}$$

where c is a *t*uning constant, taken as $c = 4$. The weight function f is called the "Fair" function, and is one of several possible weight functions cited in the original IRPLS paper [4]. Of course, other weight functions could be taken, and we do not claim any optimality properties for the choice (10). However, many numerical experiments indicated that the Fair function combined with the

proposed value for the tuning constant yields a good compromise between robustness and statistical efficiency. If the tuning constant c increases to infinity, then the weight function becomes more and more flat, and the PRM-estimator will resemble more and more PLS.

Note that the weights (9) are computed from standardized residuals. The advantage of scaling the residuals is that the weights are unaltered under scalar multiplication of all residuals, making the regression procedure *scale equivariant*. A simple and robust choice for $\hat{\sigma}$ is the *Median Absolute Deviation*:

$$\hat{\sigma} = \mathrm{MAD}(r_1, \ldots, r_n) = \underset{i}{\mathrm{median}}\, |r_i - \underset{j}{\mathrm{median}}\, r_j|.$$

The weights w_i^x measuring the leverage of each score vector $\boldsymbol{t}_i$ are computed as

$$w_i^x = f\left(\frac{\|\boldsymbol{t}_i - \mathrm{med}_{L1}(\mathbf{T})\|}{\mathrm{median}_i\, \|\boldsymbol{t}_i - \mathrm{med}_{L1}(\mathbf{T})\,\|}, c\right), \tag{11}$$

where $\|\cdot\|$ stands for the Euclidean norm. Here $\mathrm{med}_{L1}(\mathbf{T})$ denotes the L_1-median computed from the collection of score vectors $\{t_1, \ldots, t_n\}$; it is a robust estimator of the center of the data cloud of the h-dimensional score vectors. This L_1-median is a multivariate version of the sample median, also called the is a spatial median, and it can be computed very quickly [8]. Another possibility is to estimate the multivariate median by computing the coordinatewise or componentwise median. The advantage using the L_1-median is that it will make the whole partial regression procedure *orthogonally equivariant* (see Remark 2 below).

The routine for partial robust M-regression consists now of the following steps:

1. Compute robust starting values for the weights $w_i = w_i^x\ w_i^r$. For the residual weights we use formula (9) with $r_i = y_i - \mathrm{median}_j\, y_j$ and for the leverage weights we take formula (11) with the score vectors replaced by $\boldsymbol{x}_i$, for $1 \leq i \leq n$.

2. Perform an PLS regression analysis on the (re)weighted data matrices $\check{\mathbf{X}}$ and $\check{\boldsymbol{y}}$ obtained by multiplying each row of $\mathbf{X}$ and $\boldsymbol{y}$ with $\sqrt{w_i}$. PLS is performed by the SIMPLS algorithm [9]. This PLS analysis results then in an update of $\hat{\boldsymbol{\gamma}}$ and of the score matrix $\mathbf{T}$. The latter needs to be corrected for by dividing each row of $\mathbf{T}$ by $\sqrt{w_i}$.

3. Recompute the residuals $r_i = y_i - \boldsymbol{t}_i\hat{\boldsymbol{\gamma}}$ and update the weights $w_i = w_i^r \; w_i^x$ using (9) and (11).

4. Go back to step (2) until convergence of $\hat{\boldsymbol{\gamma}}$. Convergence is achieved whenever the *relative difference* in norm between two consecutive approximations of $\hat{\boldsymbol{\gamma}}$ is smaller than a specified threshold, e.g. 10^{-2}.

5. The final estimate $\hat{\boldsymbol{\beta}}$ is directly obtained from the last weighted PLS step.

Many numerical experiments showed that this iterative procedures is stable and converges quite fast. If software for computing standard PLS is available, then it is easy and fast to program the above algorithm[1].

Remark 1: If the number of explanatory variables is large compared to the number of observations ($p > n$), computation can be speed up by carrying out a preliminary singular value decomposition (SVD) on the the data matrix $\mathbf{X}^T$ (similar as in [5]). Write $\mathbf{X}^T = \mathbf{VSU}^T$, where $\mathbf{S}$ is a diagonal matrix whose diagonal elements are the n singular values of $\mathbf{X}$, and $\mathbf{U}$ an $n \times n$ orthogonal matrix. Instead of carrying out the iteration scheme presented above on $\mathbf{X}$, apply it on the reduced data matrix $\tilde{\mathbf{X}} = \mathbf{US}$, having size $n \times n$. The resulting Partial M-regression estimate $\tilde{\boldsymbol{\beta}}$ needs then to be back-transformed into $\hat{\boldsymbol{\beta}} = \mathbf{V}\tilde{\boldsymbol{\beta}}$. This estimate is mathematically equivalent to the estimate obtained by applying the algorithm directly on the full matrix $\mathbf{X}$.

Remark 2: As already mentioned above, the PM and PRM estimators are scale and orthogonally equivariant. This means that if the estimator is computed from a transformed response vector $c\,\boldsymbol{y}$ and data matrix $\mathbf{X\Gamma}$, with c any non-zero scalar and $\mathbf{\Gamma}$ any orthogonal matrix, then the property

$$\hat{\boldsymbol{\beta}}(\mathbf{X\Gamma}, c\boldsymbol{y}) = \mathbf{\Gamma}^T \hat{\boldsymbol{\beta}}(\mathbf{X}, \boldsymbol{y}) \; c \tag{12}$$

holds. The starting estimate used in the iterative algorithm already fulfills the above property, as does every subsequent approximation, resulting in a fully scale equivariant and orthogonally equivariant procedure. Note that the PM and PRM, as PLS and RSIMPLS, are *not* affine and

[1] Computer code written in Matlab is available at http://www.chemometrix.ua.ac.be/dl/prm.php

regression equivariant in the sense of Rousseeuw and Leroy [7]. Partial regression is a technique allowing only for equivariance property (12), which PM and PRM do verify .

4 Simulation study

4.1 Statistical properties

In this Section we illustrate the statistical properties of partial (robust) M-regression in comparison with PLS and the recently proposed RSIMPLS algorithm of [5]. First the efficiency of the estimators is investigated. Let $\mathbf{X}$ be a predictor data matrix of size $n \times p$, where there is perfect collinearity between the variables: $\mathbf{X} = \mathbf{T}\mathbf{B}^T$ with $\mathbf{T}$ a score matrix of size $n \times h$ and $\mathbf{B}$ a matrix of size $p \times h$, both filled with random standard normal numbers. Then $m_{\text{rep}} = 1000$ samples of size n are generated according to

$$y_i = \boldsymbol{x}_i\boldsymbol{\beta}_0 + \varepsilon_i = \boldsymbol{t}_i(\mathbf{B}^T\boldsymbol{\beta}_0) + \varepsilon_i, \tag{13}$$

for $1 \leq i \leq n$ and with $\boldsymbol{\beta}_0$ the true regression parameter, with components randomly drawn from a normal with mean zero and standard deviation 0.001. Model (13) can be considered as a latent variable regression model (7). From every sample the estimate $\hat{\boldsymbol{\beta}}^j$ is computed for $1 \leq j \leq m_{\text{rep}}$ for the appropriate value of h. A simulated value of the Mean Squared Error (MSE) of the estimator $\hat{\boldsymbol{\beta}}$ is then given by

$$\text{MSE} = \frac{1}{m_{\text{rep}}} \sum_{j=1}^{m_{\text{rep}}} \|\hat{\boldsymbol{\beta}}^j - \boldsymbol{\beta}_0\|^2,$$

and is a measure of the precision of the estimator.

In Table 1 results are reported for three different sets of values (n, p, h). In the first case, the data matrix has size 24×6, but in the two other sampling schemes we have more variables than observations, i.e. $\mathbf{X}$ has size 15×60, respectively 10×100. The simulation study has been repeated for various distributions of the error term ε_i: the standard normal, the Laplace, Student t-distributions with 5 and 2 degrees of freedom, and two heavy-tailed distributions, namely the Cauchy and Slash distributions. (The latter is defined as a standard normal divided by a uniform

distribution on [0,1]).

[Table 1 about here]

From Table 1, it is seen that PLS has the smallest MSE for normal error terms, confirming its optimality at this model. The loss in efficiency when using PM or PRM is, however, very limited at the normal model. For RSIMPLS the loss in efficiency is much more significant here, the MSE at normal errors being more than twice as high as for PLS. When deviating from the normal model, one sees that PLS immediately looses its optimality: at all other considered distributions PM and PRM perform better. For example, if the error terms follow a t_5, a distribution quite close to the normal one, then predictions made with PRM will be more precise as with PLS. Even worse, at error distributions with heavy tails, the PLS estimator breaks down and the MSEs goes beyond any bound, while this does not occur for the other estimators. The Cauchy and Slash distribution may generate quite large error terms, which could be considered as vertical outliers. The non-robustness of PLS with respect to such vertical outliers is then clearly visible from Table 1.

Comparing PM and PRM with its robust competitor RSIMPLS gives a clear outcome: PM and PRM are more precise at all considered distributions and sampling schemes. Note in particular the bad performance of RSIMPLS at the third sampling scheme, where the sample size is low. There is of course always some arbitrariness in a simulation study, but there is clear simulation evidence for good efficiency properties of PM and PRM. An explanation for this is that PM and PRM are downweighting outliers in a smooth way, resulting in stable estimators. The RSIMPLS method is reminiscent to the Minimum Covariance Determinant estimator, the latter being known to have quite a low efficiency [10].

The efficiencies for PM and PRM are very comparable across all considered settings. The reason for this is that the observations were generated by equation (13), keeping the design matrix $\mathbf{X}$ fixed and allowing only for possible outliers in the error terms. Since both PM and PRM deal

with outlying residuals in the same way, one cannot expect much difference between them. To mark the difference between both estimators one needs to resort to a different sampling scheme. We consider model (13) with normal errors, $n = 100, p = 5, h = 1$, but where in 10% of the cases, the components of the observations $\boldsymbol{x}_i$ are coming from a $N(5, 0.2)$ instead of a $N(0, 1)$. In this way, bad leverage points are induced in 10% of all samples. The MSEs are reported in Table 2. It is readily seen that PM breaks down and results in a huge MSE in comparison to PRM.

Observe the similar performance of PRM with respect to RSIMPLS from Table 2. We want to emphasize, however, that sampling schemes with extreme outliers are not very realistic in practice. We do believe that extreme outliers are most of the time detected before a chemometrical analysis is carried out. Moreover, extreme leverage points can easily be detected by a robust principal components analysis, as in [11] or [12]. Small deviations from the normal model are almost never detected, and are therefore the more relevant setting to consider.

[Table 2 about here]

Conclusions to state here are that the partial robust M-estimation procedure (i) barely loses efficiency at the normal model when no outliers are present (ii) at all other considered error distributions performs better and most often much better than PLS in terms of statistical precision (iii) can withstand both vertical outliers and leverage points. In the next Section we will show that PRM is also a computationally efficient method.

4.2 Computational properties

A major reason not to use the first version of robust PLS ever to be proposed [3] was its high computational cost. This is caused by the fact that these authors plugged in robust regression estimators wherever regression is needed in the PLS algorithm. As robust multiple regression methods themselves are often time-consuming, the same applies to the entire method. Attempts were later made to overcome this drawback, such as to plug in simple robust regressions where

the multiple regression could be replaced by a sequence of simple bivariate regressions and to keep the non-robust least squares estimator in the other case, hence obtaining a computationally efficient but only semi-robust estimator [13]. As already mentioned, a similar observation can be made about IRPLS of [4]: it is fast to compute, but not resistant to all types of outliers. In this subsection the computation times of the PRM method proposed in this paper is compared with another fully robust PLS procedures, the RSIMPLS [5] method, and it will be shown that the computational speed of the PRM is far superior.

The computational complexity of PRM-regression and RSIMPLS are studied by measuring the CPU computation times needed for a single run of their Matlab (The MathWorks, Inc.) default implementations. A sequence of data sets is generated from (13) with normal error terms and 3 latent variables. First the number of explicative variables is kept fixed at $p = 20$. In Figure 1, the computation times for both PRM and RSIMPLS are plotted against an increasing number n of observations.

[Figures 1 and 2 about here]

Computation times for RSIMPLS are substantially higher than for PRM. Figure 1 indicates that both computation times increase linearly with the number of observations, but for PRM they increase at a lower rate. Even for a sample size of 1000 in 20 dimensions, the computation time for PRM remains below 2 seconds.

In a similar way the number of observations is fixed at $n = 20$ and the number of predictor variables p varies from 10 to 600. The results of this simulation study are pictured in Figure 2, where the CPU computation times are plotted against p. Also in this setup, PRM obviously outperforms RSIMPLS. The computation time seems to be almost constant with respect to the dimension, corroborating the claim that these methods are fit for data sets with high-dimensional regressors. The constancy of the computation time in p is due to the singular value decomposition (see Remark 1 of Section 3), which treats a dataset of size $n \times p$ essentially as $n \times n$ as soon as $p > n$. If the singular value decomposition is omitted, the computation time for $p = 600$ would be about twice

as large here.

From both Figures 1 and 2, it is observed that the curve of the CPU computation times for PRM shows a linear trend, upon which a certain "noise" is superimposed. This is easily understood by the fact that PRM is computed by dint of an iterative reweighting scheme. Different simulated data sets will not require the same number of iterations before convergence is attained, explaining the variability in observed computing time. Similar small fluctuations are observed for the computing time of RSIMPLS, since the latter algorithm uses random search techniques and/or iteration steps.

We conclude that the computation time of PRM is highly competitive and forms no obstacle for its use in practice.

5 Robust calibration for the quantitative analysis of archæological glass vessels

We will now show the benefits of partial robust M-regression in an example. In 1997, in our laboratory at the University of Antwerp, an analysis was performed on 16^{th}-17^{th} century archæological glass vessels. The goal of the study was to learn more about how the beautiful vessels had been produced at the time, and which (trade) connections had existed between the different renowned producers. A first step towards a better understanding of the vessels' origin was, of course, a sound analysis of their chemical constitution. Chemical analysis was performed at the elemental level. Several analytical techniques were applied, which led to an accurate determination of the concentrations of various elements and compounds present in the glass [14].

For the larger part of the analyses, quantification was carried out using methods which bear no relation to the work presented here. However, electron-probe X-ray micro-analysis (EPXMA) was performed for which PLS regression was used to estimate concentrations from the EPXMA spectra. PLS was shown to be reliable and yielded acceptable predictions, as has been reported in [15]. In

that publication the work is presented as a straightforward application of PLS: dividing the set of samples of which the corresponding concentrations are known into a training (calibration) dataset and a validation dataset, computing the PLS vector of regression coefficients for the training set and finally computing the mean-squared error of prediction for the validation data set. However, at a first stage of analysis, PLS did not perform well. Only after it was realized that some spectra in the dataset had been measured with a different detector efficiency, and after identifying these samples and subsequently eliminating them from the data set, PLS quantification gave the good results shown in [15].

In the statistical sense, the spectra which were measured with a different detector efficiency are bad leverage points, i.e. outliers in the $\mathbf{X}$ space whose presence in the data set does harm the calibration. Hence, we are interested to know what *would have happened, if the leverage points would not have been eliminated from the data, and a partial robust M-regression would have been applied.*

Before we can give an answer to the previous question, we first need to give a little information about the data themselves (for the experimental details, we refer the interested reader to [14] and [15]). In [14], it has been shown that in fact the whole data set consisted of four groups, each one corresponding to a different type of glass. This is immediately evident from Figure 3.

[Figure 3 about here]

The different types of glass present in the data set are *sodic*, *potassic*, *potasso-calcic* and *calcic*. Furthermore, it is also obvious from Figure 3, that the majority of the vessels belong to the sodic group. Keeping this in mind, Lemberge *et al.* [15] decided that the training set should contain 20 sodic samples and 5 samples of each of the other types of glass. In our work here, we use a similar design for the training set, but since we do not preliminarily eliminate the outliers, we end up with 5 additional spectra in the sodic group, all being bad leverage points. The reason why only the sodic group is affected by outliers can be explained from a chemical point of view: a decrease in

the detector efficiency function is caused by a contamination layer on the detector's surface. The number of X-ray photons that reach the detector decreases as the thickness of the contamination layer increases. However, highly energetic photons will not be absorbed by the contamination layer. The characteristic energies for Na K_α and Fe K_α photons are 1.02 KeV and 6.4 KeV, respectively. Hence, one may expect the peaks corresponding to iron photons to be affected far less by the lower detector efficiency than the sodium peak.

In the original analysis, univariate PLS calibration was performed for all of the main constituents of the glass. These are sodium oxide, silicium dioxide, potassium oxide, calcium oxide, manganese oxide and iron (III) oxide. Prediction of the sodium oxide concentration by a non-robust method such as PLS will be affected by the outliers, whereas the effect on the prediction of the other compounds, such as iron (III) oxide, should be marginal. The beneficial effect of PRM-regression will be evident for the prediction of sodium oxide, since all of the leverage points belong to the sodic group. For the determination of iron oxide, we expect PLS and PRM to produce comparable results, although one might expect a small increase in root mean squared error of prediction (RMSEP) for PRM due its lower statistical efficiency at the normal model.

We performed PLS and PRM regression on the calibration set described before. We used the number of 8 latent variables for Na_2O and 7 latent variables for Fe_2O_3, as in [15]. We estimated the concentrations of Na_2O and Fe_2O_3 for the validation set, and obtained an overall good agreement between estimated and true concentrations. The root mean squared errors of prediction are given in Table 3. We also state the respective RMSEP's obtained by Lemberge *et al.* posterior to removal of the leverage points from all of the data from the training set. The latter are reported in Table 3 under the heading "cleaned" data, in contrast to the "original" data.

[Table 3 about here]

The result from Table 3 clearly shows a huge increase of the RMSEP of PLS for sodium oxide when bad leverage points are present in the data. The RMSEP of PRM for sodium oxide is

considerably lower than the RMSEP of PLS, due to the robustness of the method. In fact, the RMSEP of PRM for the contaminated data set is still quite comparable to the RMSEP of PLS for the data set from which the outliers had been removed.

For iron (III) oxide, we see that PLS successfully extracts the relevant information and is not affected by the outliers, as the latter are all sodic samples. From our simulation study (Table 1), one could already see that PRM is still very efficient at the normal model and might even be more precise at other distributions, even when no outliers are present in these data. This result is illustrated as well in this practical application. When PRM is applied for the determination of the concentration of iron (III) oxide, we see that the RMSEP of PRM-regression is marginally lower than the RMSEP of PLS.

In general, we conclude from these results that the time-consuming step wherein the spectra with a different detector efficiency were isolated, may well have been skipped if partial robust M-regression had been available at the time. The RMSEP of PRM for the sodic group is still very much acceptable and the RMSEP for the virtually non-contaminated data set for the analysis of the manganese oxide concentration, are comparable.

6 Summary and Outlook

In this article, we have proposed a partial version of M-estimators. When using the appropriate loss-function, partial M-regression inherits the good properties of M-estimators: (i) they are highly efficient at the normal regression model and even more efficient than Least Squares at a variety of other models (ii) they are fast to compute (iii) if weights for leverage points are inserted, resulting in the Partial Robust M-estimator, a procedure robust to all types of outliers is obtained. In the setting of partial regression, adding weights for leverage points is not computationally expensive, since the estimators only need to be scale and orthogonally equivariant.

Partial Robust M-regression is easy to implement, since it can be computed with a variant of an

algorithm already proposed under the name Iteratively Reweighted Partial Least Squares regression [4]. But while IRPLS does not protect against all types of outliers, partial robust M-regression is entirely robust and remains to be practicable for high-dimensional data sets. A simulation study has shown that in terms of computational cost and statistical efficiency, PRM outperforms its main competitor [5] while having both robustness and equivariance properties.

We applied PRM to an EPXMA analysis of archæological glass samples, where some spectra had been measured with a different detector efficiency function. These spectra thus form a group of leverage points in the data set. It has been shown that application of PRM to the entire data set performs comparably to a tedious examination of the detector efficiencies of the respective spectra in order to be able to eliminate the aberrant group of spectra.

Most previous research focused on developing robust versions of Partial Least Squares estimators, while we propose a partial version of a robust estimator. Very recently, a similar point of view was taken by [16], proposing a partial Least Absolute Deviation (PLAD) estimator. The way it is defined in [16], however, results in an estimator being not orthogonally equivariant and not robust with respect to leverage points. Note that by taking the loss function $\rho(u) = |u|$ in definition (3), PLAD results as a special case of partial M-estimators.

Although this article provides both a sound theoretical and practical underpinning for partial robust M-regression, some aspects still have to be explored further in future work. As presented in this paper, the variable $\boldsymbol{y}$ to predict is univariate. An extension to multivariate PRM (which can be seen as a robust alternative to PLS2) seems to be quite straightforward. Also, the number of latent variables was taken to be fixed to h. In practice the number h needs to be selected. Common cross-validation can be applied here: indeed, the last step of the IRPLS algorithm is not only returning a single regression estimate, but also the weighted PLS estimates for the lower order models. Finally, standard errors around the estimates could be obtained by the bootstrap, comparable to the bootstrap proposed in [17] for PLS. If the L_1-median in the algorithm of Section 3 is replaced by the coordinatewise median [2], then the computation time is being further deflated

[2] Using the coordinatewise median would imply that we loose the orthogonal equivariance property, but it will

by an important factor, making application of cross-validation and bootstrapping techniques even more feasible.

Acknowledgments

Research financed by a PhD grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). The second author would like to thank the "Fonds voor Wetenschappelijk Onderzoek-Vlaanderen" contract number G.0385.03) and the Research Fund K.U.Leuven (for financial support).

not affect the variance of the estimator. Hence we propose to keep the L_1-median to compute the point estimate, but to use the coordinatewise median for the bootstrapping procedure.

References

[1] H. Wold, in: P.R. Krishnaiah (ed.), Multivariate Analysis III, Academic Press, New York (1973), 383-407.

[2] P.J. Huber, Robust Statistics, Wiley, New York, 1981.

[3] I.N. Wakeling, H.J.H. MacFie, J. Chemometr., 6 (1992), 189-198.

[4] D.J. Cummins, C. Andrews, J. Chemometr. 9 (1995), 489-507.

[5] M. Hubert, K. Vanden Branden, J. Chemometr., 17 (2003), 537-549.

[6] A.P. Dempster, N.M. Laird and D.B. Rubin, in: P.R. Krishnaiah (ed.), Multivariate Analysis V, North-Holland, Amsterdam, 1980, 35-57.

[7] P.J. Rousseeuw, A.N. Leroy, Robust regression and outlier detection, Wiley, New York, 1987.

[8] O. Hössjer, C. Croux, J. Nonparametr. Stat., 4 (1995), 293-308.

[9] S. de Jong, Chemometr. Intell. Lab. Syst., 18 (1993), 251-263.

[10] C. Croux, G. Haesbroeck, J. Multivariate Anal., 71 (1999), 161-190.

[11] C. Croux, A. Ruiz-Gazen, J. Multivariate Anal., to appear.

[12] C. Croux, G. Haesbrouck, Biometrika, 87 (2000), 603-618.

[13] M.I. Griep, I.N. Wakeling, P. Vankeerberghen, D.L. Massart, Chemom. Intell. Lab. Syst., 29 (1995), 37-50.

[14] K.H. Janssens, I. De Raedt, O. Schalm, J. Veeckman, Mikrochim. Acta, 15(Suppl.) (1998), 253-267.

[15] P. Lemberge, I. De Raedt, K.H. Janssens, F. Wei, P.J. Van Espen, J. Chemometr., 14 (2000), 751-763.

[16] Y. Dodge, A. Kondylis, J. Whittaker, in J. Antoch (ed.), Compstat 2004 Proceedings in Computational Statistics, Springer-Verlag, Heidelberg, 2004, 935-942.

[17] M.C. Denham, J. Chemometr., 11 (1997), 39-52.

Error distribution		N(0,1)	Laplace	t_5	t_2	Cauchy	Slash
$\frac{n}{p} = 4$, h=2	PLS	0.0199	0.0425	0.0337	0.3432	48.011	37195
	PM	0.0221	0.0300	0.0276	0.0415	0.0711	0.1584
	PRM	0.0240	0.0315	0.0295	0.0435	0.070	0.1666
	RSIMPLS	0.0462	0.0521	0.0520	0.0672	0.1026	0.2105
$\frac{n}{p} = \frac{1}{4}$, h=1	PLS	0.0011	0.0021	0.0020	0.0112	30.742	10.923
	PM	0.0012	0.0017	0.0017	0.0023	0.0042	0.0089
	PRM	0.0013	0.0019	0.0018	0.0026	0.0047	0.0099
	RSIMPLS	0.0024	0.0027	0.0029	0.0036	0.0067	0.0135
$\frac{n}{p} = \frac{1}{10}$, h=3	PLS	0.0040	0.0078	0.0064	0.0582	124.62	188.32
	PM	0.0046	0.0067	0.0060	0.0099	0.0293	0.0518
	PRM	0.0047	0.0069	0.0061	0.0099	0.0283	0.0469
	RSIMPLS	0.0155	0.0247	0.0211	0.0346	0.0969	0.1517

Table 1: Simulated Mean Squared Error for the PLS, PM, PRM, and RSIMPLS regression estimators at several error distributions and under 3 different sampling schemes of sample size n and predictor dimension p.

PLS	PM	PRM	RSIMPLS
40.10	42.94	5.02	5.75

Table 2: Simulated Mean Squared Error for the PLS, PM, PRM, and RSIMPLS regression estimators at a sampling scheme generating bad leverage points.

	Na_2O		Fe_2O_3	
	Original	Cleaned	Original	Cleaned
PLS	2.66	1.26	0.14	0.12
PRM	1.50	–	0.10	–

Table 3: Root mean squared errors of prediction for the EPXMA data set using the PLS and PRM-estimator, once using the original training sample and once using a clean version of the training sample, as in [15].

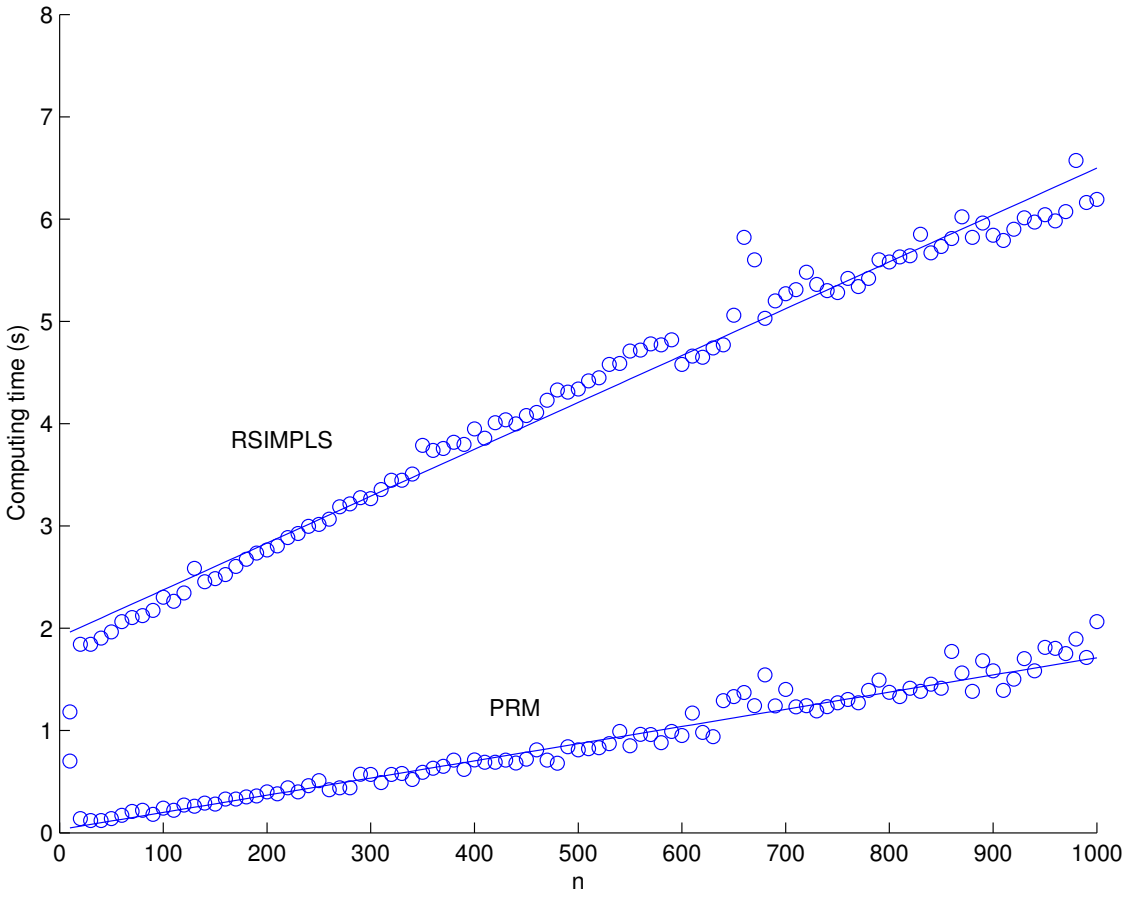

Figure 1: Computation times in seconds for PRM and RSIMPLS for simulated data sets with an increasing number of observations n and with $p = 20$.

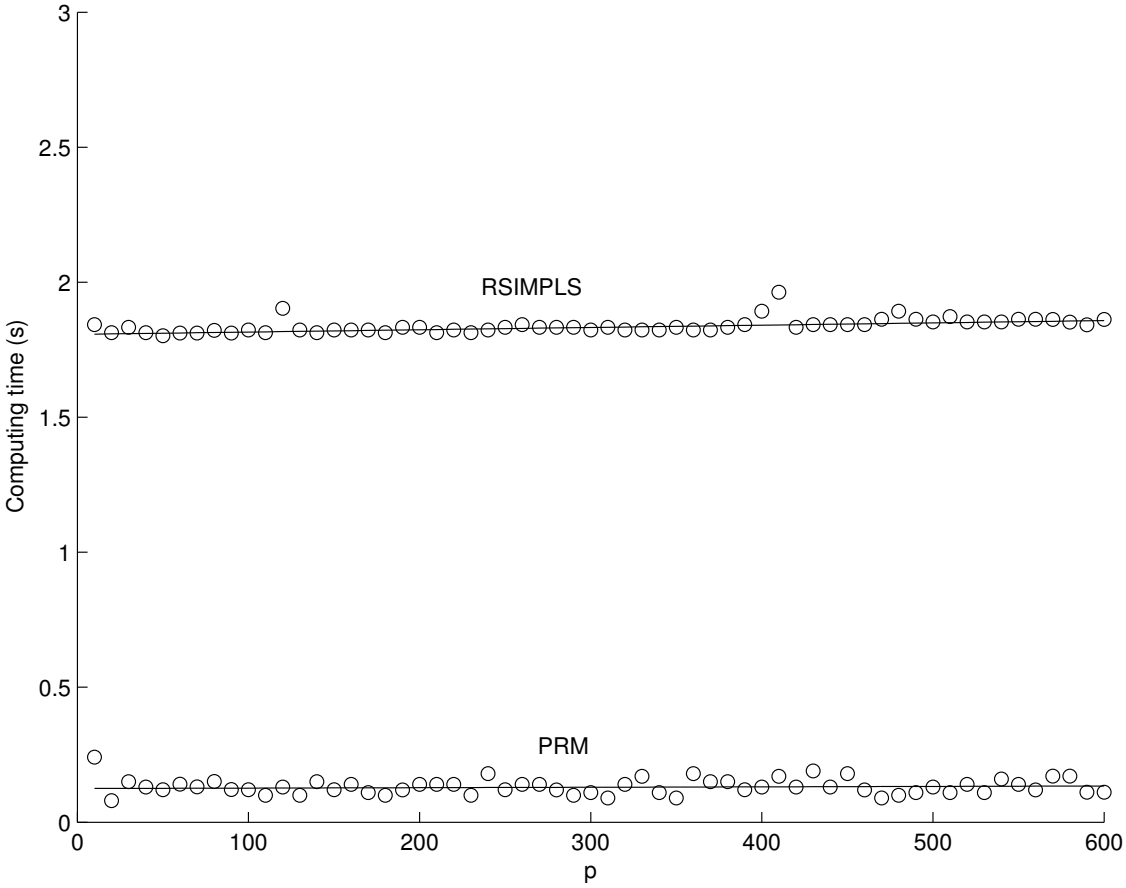

Figure 2: Computation times in seconds for PRM and RSIMPLS for simulated data sets with an increasing number p of predictor variables and sample size $n = 20$.

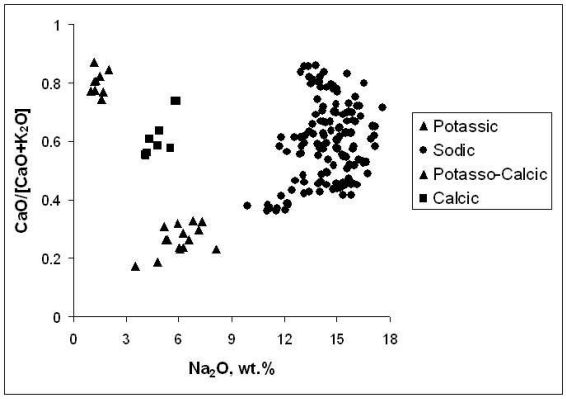

Figure 3: Ratio $CaO/[CaO+K_2O]$ plotted against Na_2O concentration for all glass vessels analyzed.