Latent Factor Prediction Pursuit for Rank Deficient Regressors

K. Luebke [*] I. Czogiel C. Weihs

September 2004

Universität Dortmund
Fachbereich Statistik

Abstract

In simulation studies Latent Factor Prediction Pursuit outperformed classical reduced rank regression methods. The algorithm described so far for Latent Factor Prediction Pursuit had two shortcomings: It was only implemented for situations where the explanatory variables were of full colum rank. Also instead of the projection matrix only the regression matrix was calculated. These problems are addressed by a new algorithm which finds the prediction optimal projection.

Keywords
simulated annealing, prediction oriented projections, reduced rank regression, rank deficient regressors, simulation study

1 Introduction

It is known that predictions in a multiple, multivariate linear regression are rather poor when the explanatory variables are collinear or the number of observations is not much larger than the number of parameters to estimate (Helland and Almøy, 1994). This may, for example, be caused by overfitting or unstable estimates. To tackle these problems, a reduced rank regression (RRR) method can be used (Reinsel and Velu, 1998). In a reduced rank regression the explanatory variables are projected on (few) so-called latent factors which are used as regressors for the response variables. There exist different RRR methods to find this projection, but it is not clear which method to use for data at hand.

In Luebke and Weihs (2004a, 2003) a new method which finds the projection on latent factors which is optimal for prediction was proposed. The Latent

[*]e-mail: luebke@statistik.uni-dortmund.de

Factor Prediction Pursuit (LaFaPP) method is based on computer intensive direct minimization of the mean squared error of prediction (MSEP) (Weihs and Hothorn, 2002). The optimization is carried out by simulated annealing (Bohachevsky et al., 1986; Salamon et al., 2002). With a search algorithm the optimum is found without using any distributional assumptions on the error structure, so it is ready for general application. The algorithm described in (Luebke and Weihs, 2004a) has two shortcomings: First, it only works with explanatory variables which are of full colum rank. Second, only the regression matrix instead of the projection matrix is calculated. Both problems are solved by the new algorithm described in this paper.

In order to compare the new method with 'classical' methods a simulation study is carried out. The simulation study is based on Breiman and Friedman (1997).

This paper is organized as follows: In Section 2 we introduce the latent factor model. The prediction criterion is investigated in Section 3. In Section 4 the new latent factor prediction pursuit algorithm is explained. The simulation study and the results thereof are shown in Sections 5 and 6.

2 Latent Factor model

The basic multiple, multivariate linear model looks as follows:

$$Y = 1_n\mu + XM + E \tag{1}$$

with

$Y \in \mathbb{R}^{n\times q}$	data of q response variables,
$\mu \in \mathbb{R}^{q}$	mean column vector of responses,
$X \in \mathbb{R}^{n\times p}$	data of p explanatory variables (X is mean centered),
$M \in \mathbb{R}^{p\times q}$	unknown regression coefficient matrix,
$E \in \mathbb{R}^{n\times q}$	matrix of errors.

Instead of the original explanatory variables X in this work a projection of these (possible) high dimensional variables on (few) variables Z is used. This may be important because of numerical reasons (collinearity / singularity or overfitting) or because of some model assumptions, e.g. that the response variables Y depend on some underlying latent factors. So in a latent factor model instead of the variables X in model (1) latent variables Z under the side-condition $Z'Z = I_r$ $(r \leq k)$ with $Z = XG$ are used. The model with latent factors is:

$$Y = 1_n\mu + XM + E = 1_n\mu + (XG)B + E \tag{2}$$

with the side condition

$$(XG)'(XG) = I_r. \tag{3}$$

Is is shown in Luebke and Weihs (2004b) that concerning a prediction optimal solution for G it is only necessary to look for $G \in \mathbb{R}^{p\times r}$ with $r \leq \min(rank(X), rank(Y))$.

Given the estimates $\hat{G}$ of G, fulfilling the side-condition $(X\hat{G})'(X\hat{G}) = I_r$, and $\hat{\mu}$ of μ it is assumed that in the latent factor model the estimate of B is the usual least square estimate of Y on $Z = XG$.

$$\hat{B} = [(X\hat{G})'(X\hat{G})]^{-1}(X\hat{G})'(Y - 1_n\hat{\mu}) = (X\hat{G})'(Y - 1_n\hat{\mu}).$$

The ordinary least squares estimator for B is used by reduced rank regression techniques like Canonical Correlation Analysis (CCA), Principal Component Regression (PCR) or Partial Least Squares (PLS). Thus, the only difference between these methods is their way of estimation of G. So the estimate of M in a latent factor model is

$$\hat{M} = \hat{G}\hat{B} = \hat{G}(X\hat{G})'(Y - 1_n\hat{\mu}).$$

3 Prediction criteria

Suppose that the mean μ and the regression matrix M are estimated on a training set and that it is crucial how this estimator will perform on n_0 future values X_0, Y_0. The point prediction of the future response values is:

$$\hat{Y}_0 = 1_{n_0}\hat{\mu}_Y + X_0\hat{M}_{X,Y}.$$

M and μ are estimated on the training set X, Y. With a known test set X_0, Y_0 the loss in n_0 (new) observations can be measured by

$$L = \frac{1}{n_0}\|(Y_0 - \hat{Y}_0)\Gamma^{-\frac{1}{2}}\|^2, \tag{4}$$

where Γ is a fixed $q \times q$ weight matrix. One possible choice is the diagonal matrix of the variances of the response variables (Schmidli, 1995, p. 22). This was used here as the loss in Y and Y_0 is measured relative to the variance of the responses.

The loss (4) is closely linked to the mean R^2 of the response variables (Schmidli, 1995, p. 23) where the R^2 is measured on the future values Y_0 and the mean is taken over all q response variables:

$$R^2_{mean} = 1 - \frac{L}{q}. \tag{5}$$

Usually one is not only interested in the performance of the estimator for some observations but also in the 'general' or average performance. The corresponding mean loss (mean squared error of prediction) is defined as (Schmidli, 1995, p. 24):

$$\begin{aligned} MSEP &= \frac{1}{n_0}E_{Y|X}E_{Y_0|X_0}\|(Y_0 - \hat{Y}_0)\Gamma^{-\frac{1}{2}}\|^2 \\ &= \frac{1}{n_0}E_{Y|X}E_{Y_0|X_0}\|(Y_0 - (1_{n_0}\hat{\mu} + X_0\hat{M}_{Y,X}))\Gamma^{-\frac{1}{2}}\|^2 \qquad (6) \\ &= \frac{1}{n_0}E_{Y|X}E_{Y_0|X_0}\|(Y_0 - (1_{n_0}\hat{\mu} + X_0(\hat{G}_{X,Y}\hat{G}'_{X,Y}X'(Y - 1_n\hat{\mu}))))\Gamma^{-\frac{1}{2}}\|^2 \end{aligned}$$

Equation (6) shows that the *MSEP* can be seen as a function of the projection matrix G. So by using different G – and taking care of the side-condition – different *MSEP* can be achieved. Estimation of *MSEP* can be done by bootstrap methods.

4 Latent factor prediction pursuit

The latent factor prediction pursuit method (LaFaPP) tries to minimize the *MSEP* given in equation (6) within the possible solution space of the side-condition (see equation (3)). So its basic idea is to look at the *MSEP* as a function of G and minimize it directly by means of simulated annealing on G. The conditional expectations $E_{\cdot|\cdot}$ in (6) are estimated by bootstrap. This means that the given data are randomly split into two parts, namely X, Y and X_0, Y_0, respectively.

The computer intensive minimization is done by means of simulated annealing on the basis of a Nelder-Mead (NM) algorithm (Press et al., 1992, p. 451). The Nelder-Mead (or downhill simplex) algorithm is an optimization method that does not need any gradients. It transforms a simplex of $m + 1$ points for an m dimensional problem. The functional values are calculated and the worst point is reflected through the opposite face of the simplex. If this trial point is best, the new simplex is expanded further out. If the function value is worse than the second worst point, the simplex is contracted. If no improvement at all is found, the simplex is shrunk towards the best point. This procedure terminates when the differences in the function values between the best and worst points are neglectable.

In contrast to the pure downhill simplex method of the Nelder-Mead algorithm the simulated annealing method accepts a trial point with a probability proportional to a parameter t (called temperature) even if it is worse than the points in the simplex (a better point is always accepted). Because of this it is hoped that local minima can be overcome as the algorithm sometimes goes 'uphill'. By reducing the temperature slowly the algorithm will at last converge to the next minimum. The basic simulated annealing algorithm is labeled in Algorithm 1.

In LaFaPP the simulated annealing is applied to the vectorized projection matrices G. The stochastic search by means of simulated annealing is done within the solution space of the side condition.

- The first problem in the calculation of G in Luebke and Weihs (2004a) is, that it only works for X which are of full column rank. In the new algorithm this is solved by a QR decomposition of XG (XG of full column rank), see for example (Harville, 1997, 66). (In a QR decomposition a matrix A is decomposed into $A = QR$ with Q being orthonormal and R a triangle matrix.) As with the QR decomposition the image space of A is unchanged and just an orthonormal basis (Q) is constructed, it is a suitable tool for the given problem of orthonormalizing XG: A trial matrix G_{trial} is transformed to $G = G_{trial}R^{-1}$ with R calculated by the QR decomposition of $Z = XG$. By this the side-condition (3) is fulfilled:

$$\begin{aligned}(XG)'(XG) &= (XG_{trial}R^{-1})'(XG_{trial}R^{-1}) \\ &= (QRR^{-1})'(QRR^{-1}) \\ &= Q'Q \\ &= I_r\end{aligned}$$

With some feasible G_{trial} the condition that Z is of full column rank can be fulfilled if $r \leq rank(X)$.

- The second problem in Luebke and Weihs (2004a) is that it was only possible to calculate the overall M and not the overall G for the whole data. One of the reasons is, that there are equivalent projection matrices G which lead to the same estimator $\hat{M}$. Let D be an orthonormal matrix and G, B some estimators. Then

  $$\begin{aligned}\tilde{G} &= GD \\ \Rightarrow \tilde{B} &= (X\tilde{G})'Y = (XGD)'Y = D'B \\ \Rightarrow \tilde{M} &= \tilde{G}\tilde{B} = GDD'B = GB = M.\end{aligned}$$

  One idea to tackle this problem of combining the G's that lead to orthonormal latent factors in the individual bootstrap sample was to use some kind of procrustes rotation (Sibson, 1978). This turned out to be unsuccessful. A reason could be that the fit of the single transformed G of the bootstrap samples on the overall G could be very bad. So with a generalized procrustes analysis the goodness of LaFaPP decreases significantly compared to the method described in Luebke and Weihs (2004a). Therefore the side-condition (3) was only used the the whole (training and test) data and not for the individual bootstrap samples. So in the bootstrap samples the training data is projected by G and the regression estimate $\hat{M}$ for the sample is calculated by an ordinary regression of the response values of the sample of the projected data. By this the 'latent factors' of the individual bootstrap samples are not orthonormal in general and so for calculating the regression matrix some matrix inversion has do be done but for the whole data the latent factors are orthonormal.

Hence an estimator for the $MSEP$ and the objective function of G is:

$$\widehat{MSEP}(G) = \frac{1}{n_{boot}} \sum_{i=1}^{n_{boot}} \frac{1}{q} \|(Y_0^i - X_0^i G(G'(X^i)'(X^i)G)^{-1}(X^i)'Y^i)\Gamma^{-\frac{1}{2}}\|^2, \quad (7)$$

with X^i, Y^i, X_0^i, Y_0^i are the i-th bootstrap sample (mean centered by the mean of the training sample) and n_{boot} is the number of the bootstrap samples.

In our implementation G is constructed columnwise. After the first column is found the following columns are constructed so that they fulfill the side condition. Because of this stepwise construction it is also possible to choose the rank ($r \leq \min(rank(X), rank(Y))$) of G so that $\widehat{MSEP}$ is minimized.

One way of describing the LaFaPP method is given in Algorithm 1 (p is the number of columns of X).

Algorithm 1 Minimization of *MSEP* by LaFaPP

```
1: Initialize $t_0$, $c_s$, $n_{iter}$ and $n_{temp}$
2: Initialize number of bootstrap replications $n_{boot}$
3: for $i = 1$ to $n_{boot}$ do
4:   Generate bootstrap samples
5:   $G_{old} := \varnothing$
6:   for Rank $r = 1$ to $\min(rank(X), rank(Y))$ do
7:     —Begin Simulated Annealing—
8:     Set $t := t_0$
9:     Build random start simplex $S$ ($p+1$ vertices) as candidates for the $r^{th}$ column of
       $G_{new}$
10:    $G^w_{new} = [G_{old}, S_{w,\cdot}]$,   $w = 1, \ldots, p+1$. $S_{w,\cdot}$ is the $w^{th}$ point in the simplex
11:    Adapt each $G^w_{new}$ to side-condition (3)
12:    Calculate objective function $f(G^w_{new})$, $w = 1, \ldots, k+1$, (cf. eq. (7))
13:    for $i = 1$ to $n_{temp}$ do
14:      for $j = 1$ to $n_{iter}$ do
15:        Set $f_{temp}(F^w_{new}) = f(F^w_{new}) + t|log(u)|$, $w = 1, \ldots, k+1$, $u \in U(0,1)$
16:        According to NM transition function generate trial point
17:        Adapt each $G_{trial}$ to side-condition (3))
18:        Calculate $f_{temp}(G^{trial}) = f(G^{trial}) - t|log(u)|$
19:        if $f_{temp}(G^{trial}) < \max(G(F^{\cdot}_{new}))$ then
20:          Replace according to NM worst point in simplex with trial point
21:        else
22:          Contract (old) simplex according to NM
23:        end if
24:      end for
25:      Reduce temperature $t = c_s\, t$
26:    end for
27:    —End Simulated Annealing—
28:    Attach columnwise to $G_{old}$ the best point in the final simplex
29:  end for
30: end for
```

The parameters for the Simulated Annealing are shown in Table 1.

We implemented the algorithm in C++ with help of the matrix library Newmat (Eddelbüttel, 1996). For the data generation and the other reduced rank regression methods compared in the simulation study we used the statistical program R (R Development Core Team, 2003).

5 Design Simulation Study

In order to compare LaFaPP with several previously proposed methods for predicting multivariate responses, a simulation study was carried out. As mentioned earlier the design of the study is taken from Breiman and Friedman (1997). In this study data was generated according to a multivariate linear model (1). To obtain general results, some features of the data, the goodness of a prediction method largely depends on, were varied, namely:

- The number of observations n

Table 1: Parameter of the implemented Simulated Annealing Algorithm.

Name	*Abbreviation*	*Value*
Start temperature	t_0	1
Linear cooling parameter	c_s	0.8
Number of iterations at each temperature	n_{iter}	100
Number of iterations of temperature	n_{temp}	50
Number of bootstrap replications	n_{boot}	100

- The number of response variables q
- The number of the predictor variables p
- The correlation structure of the predictor variables
- The signal to noise ratio
- The correlation structure of the response variables
- The covariance structure among the errors

For each observation of each situation the predictor variables were generated as a random sample from a p-dimensional normal distribution with zero mean and covariance matrix V,

$$x \sim N(0, V).$$

As indicated above the covariance structure of the predictor variables was varied. This was done by creating V in the following manner

$$V_{ij} = s^{|i-j|}, \quad i, j = 1, \ldots, p.$$

In the simulation study two levels of s were considered: 0 (independence of the predictor variables) and 0.99 (ensures a high degree of collinearity among the predictor variables).
The matrix of (true) coefficients M was generated as a product of two matrices $T \in I\!R^{p \times 10}$ and $U \in I\!R^{10 \times q}$ which contain random components,

$$M = TU. \tag{8}$$

The coefficients of T are calculated as follows

$$T_{jk} = h_k (T^*_{jk})^2,$$

where

$$T^*_{jk} = \max\{l_k - |j - j_k|, 0\}.$$

The quantities l_k and j_k are random integers sampled from a uniform distribution in the ranges $[1, p]$ and $[1, 6]$ respectively. h_k is a scaling factor which ensures $\sum_{j=1}^{p} T_{jk} = 1$:

$$h_k = \frac{1}{\sum_{j=1}^{p} T^*_{jk}}, \quad k = 1, \ldots, 10.$$

By this every row of T contains only a few non-zero elements which form a triangle centered at j_k, so in the regression $(M = TU)$ only some variables are important. This is illustrated in Figure 1. The matrix U row-wise contains 10

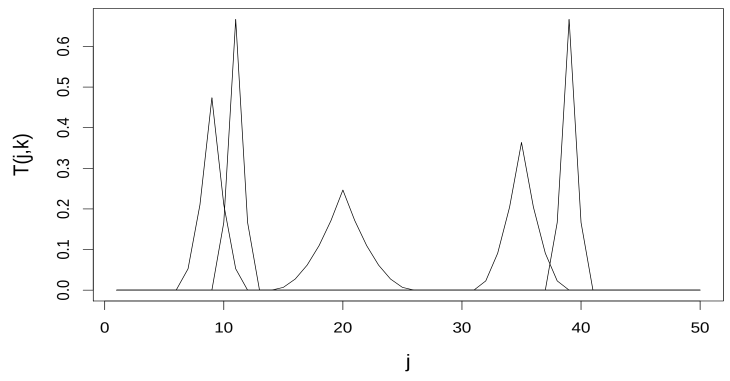

Figure 1: T_{jk} for $p = 50$ and 5 different k

independent random samples from a $q-$dimensional normal distribution with zero mean and covariance matrix Γ,

$$\Gamma_{ij} = \rho^{|i-j|}, \quad i, j = 1, \ldots, q.$$

The value of ρ represents the average correlation among the response functions $f_i(x) = \sum_{j=1}^{p} m_{ij} x_j$ and therefore controls the correlation structure among the response variables. In our simulation study we considered two values of ρ: 0 (low degree of correlation among the response variables) and 0.7 (high degree of correlation among the response variables). By the matrix product (8) the coefficients in M for each response are different random superpositions of the same 10 'bumps' or triangles. Finally, all coefficients of M were normalized by the same scaling factor so that the average (signal) variance is equal to one. By the whole (random) construction of M the coefficients in each column have sometimes roughly the same absolute values, whereas sometimes they are very different (Breiman and Friedman, 1997).
The matrix of errors E consists of n random samples from a q-dimensional normal distribution with zero mean and covariance matrix Σ. Two covariance structures among the errors were considered:

$$\Sigma = \sigma I_q \quad \text{(Homoscedasticity)} \tag{9}$$

$$\Sigma = \sigma \operatorname{diag}\big(\{i^2\}_1^q\big) \quad \text{(Heteroscedasticity).} \tag{10}$$

For each of the two covariance structures two values of σ were studied. They were chosen to produce the average signal-to-noise ratios $e = 1.0$ and $e = 3.0$

$$\sigma = \frac{q}{\text{trace}(\Sigma)e}.$$

For each factor listed above two levels were considered. The chosen values are shown in Table 2. The seven-factor-interaction was confounded so that a

Table 2: Design of experiment of simulation study

Factor	*Realization -1*	*Realization +1*
n	250	25
q	5	10
p	50	100
s	0	0.99
e	3	1
ρ	0	0.7
covariance structure of error	homoscedasticity	heteroscedasticity

2^{7-1} fractional factorial design with 64 runs was performed to generate the data. The whole design was repeated 10 times so 640 different runs are used in the simulation study.

6 Result of Simulation Study

Latent Factor Prediction Pursuit was compared to Partial Least Squares (PLS, see for example Weihs and Jessenberger (1999, p. 170)). PLS is a heuristically motivated reduced rank regression technique that turned out to be work very well concerning prediction in many situations (see e.g. Almøy (1996), Luebke and Weihs (2004a)). Note that in Breiman and Friedman (1997) the performance measures are based on some divergence of M and $\hat{M}$. As the prime interest here is prediction the maximum of the average prediction prediction R^2 (5) and 0 on 1000 validation examples is used as the performance measure:

$$\tilde{R}^2 = \max(R^2_{mean}, 0)$$

As the R^2 is evaluated on validation data it could be less than zero (this happened especially in situations when $n = 25$) but to make it comparable to the standard R^2 which takes values from 0 to 1 we adjusted it to the same interval.

The new LaFaPP outperformed PLS 283 times whereas PLS was better in 134 runs. The $\tilde{R}^2$ of LaFaPP is 0.287 – remember that $\tilde{R}^2$ is based on future (validation) data not used to estimate the M. The $\tilde{R}^2$ of PLS is 0.256. So concerning prediction power LaFaPP is roughly 10% better than PLS. This is also shown in Figure 2 where the boxplot of the performance of both methods

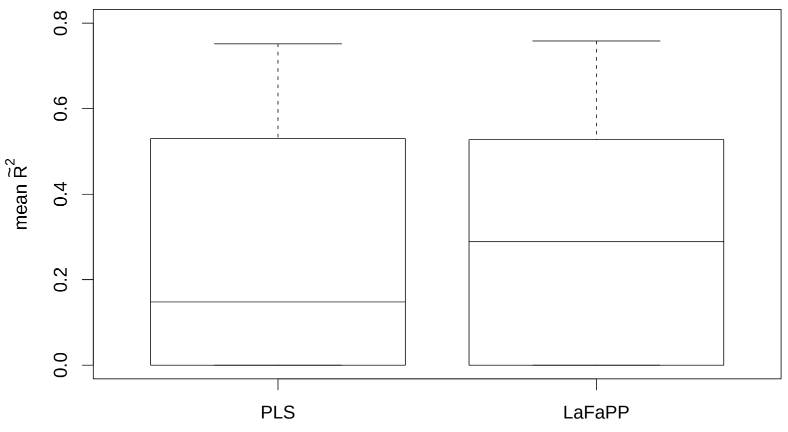

Figure 2: Comparison of LaFaPP and PLS concerning $\tilde{R}^2$

is shown. It can be seen that both methods differ in the median but not in the other quartiles.

Moreover let us comment on the results for the different factors of the simulation study. The Figures for the results are given in the Appendix (page 14) and it can be seen that LaFaPP outperforms PLS in all settings of the simulation study.

- Figure 4 shows that both methods are improved by correlation in the predictor variables but LaFaPP more than PLS.
- With a lower signal-to-noise ratio both methods are getting worse – which could be expected (see Figure 5).
- With a higher correlation in the response variables (ERR, covariance structure) the methods improve slightly (Figure 6).
- In Figure 7 and 8 it can be seen that ρ and q seem to have no influence on the performance of LaFaPP and PLS.
- PLS is getting worse with more predictor variables. LaFaPP is rather stable. Remember that no method of finding the correct rank (number of latent factors) is used (compare Figure 9).
- The Deterioration with less training samples which is about the same for both methods is shown in Figure 10.

So it can be claimed that LaFaPP is more stable than PLS and is performing better in general in our simulation study.

In the simulation study above the training- and test data were split in two equal halves for estimating (7). In order to check whether this is optimal we repeated the experiment where all factors set on -1 with a training fraction ranging form 0.1 to 0.9 with 3 random samples obtained in this setting of simulation study. As it can be seen in Figure 3 the number of observations used for training in the bootstrap samples in (7) is not important.

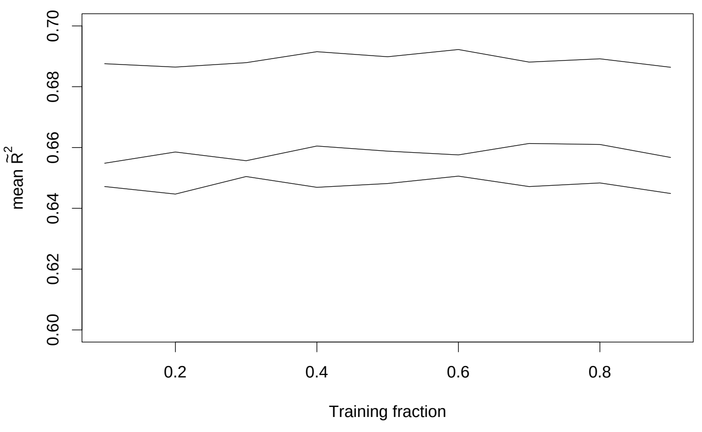

Figure 3: Comparison of different training fraction in LaFaPP concerning $\tilde{R}^2$

7 Conclusion

Concerning predictive power, the new LaFaPP method turns out to outperform the classical reduced rank methods Partial Least Squares in all situations tested in the simulation study. It has several advantages like small *MSEP* and the fact that it does not make any assumptions on the data. As it is a computer intensive method it demands more computational time. A further advantage of LaFaPP is that it projects on no more than necessary latent factors. (Luebke and Weihs, 2004b).

Acknowledgment

This work has been supported by the Collaborative Research Center ‘Reduction of Complexity in Multivariate Data Structures’, SFB 475, of the German Research Foundation (DFG).

References

Trygve Almøy. A simulation study on comparison of prediction methods when only a few components are relevant. *Computational Statistics & Data Analysis*, 21:87–107, 1996.

Ihor O. Bohachevsky, Mark E. Johnson, and Myron L. Stein. Generalized simulated annealing for function optimization. *Technometrics*, 28(3):209–217, 1986.

Leo Breiman and Jerome H. Friedman. Predicting multivariate responses in multiple linear regression. *Journal of the Royal Statistical Society, Series B, Methodological*, 59:3–37, 1997.

Dirk Eddelbüttel. Object-oriented econometrics: Matrix programming in c++ using gcc and newmat. *Journal of Applied Econometrics*, 11(2):299–314, 1996.

David A. Harville. *Matrix Algebra From a Statisticians's Perspective.* Springer, 1997.

Inge S. Helland and Trygve Almøy. Comparison of prediction methods when only a few components are relevant. *Journal of the American Statistical Association*, 89(426):583–591, 1994.

Karsten Luebke and Claus Weihs. Prediction optimal data analysis by means of stochastic search. In Martin Schader, Wolfgang Gaul, and Maurizio Vichi, editors, *Between Data Science and Applied Data Analysis*, pages 305–312. Springer, 2003.

Karsten Luebke and Claus Weihs. Generation of prediction optimal projection on latent factors by a stochastic search algorithm. *Computational Statistics & Data Analysis*, 47(2):297–310, 2004a.

Karsten Luebke and Claus Weihs. A note on the dimension of the projection space in a latent factor regression model with application to business cycle classification. Technical Report 29, Sonderforschungsbereich 475, Universität Dortmund, 2004b.

W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, second edition, 1992.

R Development Core Team. *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria, 2003.

Gregory C. Reinsel and Raja P. Velu. *Multivariate Reduced-Rank Regression, Theory and Applications.* Springer, 1998.

Peter Salamon, Paolo Sibani, and Richard Frost. *Facts, Conjectures and Improvement for Simulated Annealing.* Monographs on Mathematical Modeling and Computation. SIAM, 2002.

Heinz Schmidli. *Reduced Rank Regression.* Physica Verlag, 1995.

Robin Sibson. Studies in the robustness of multidimensional scaling: Procrustes statistics. *Journal of the Royal Statistical Society B*, 40(2):234–238, 1978.

Claus Weihs and Torsten Hothorn. Determination of optimal prediction oriented multivariate latent factor models using loss functions. Technical Report 15, Sonderforschungsbereich 475, Universität Dortmund, 2002.

Claus Weihs and Jutta Jessenberger. *Statistische Methoden zur Qualit"atssicherung und -optimierung in der Industrie.* Wiley VCH, 1999.

Appendix

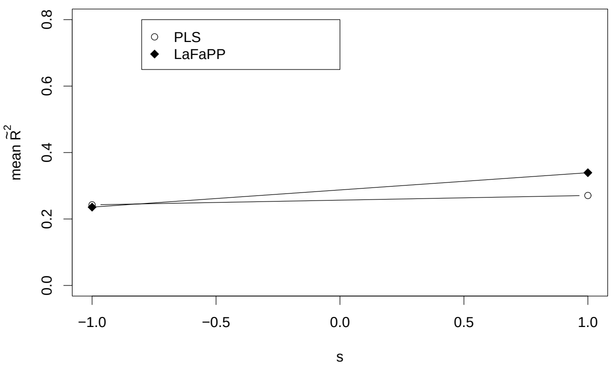

Figure 4: Comparison of LaFaPP and PLS concerning $\tilde{R}^2$ for the factor s

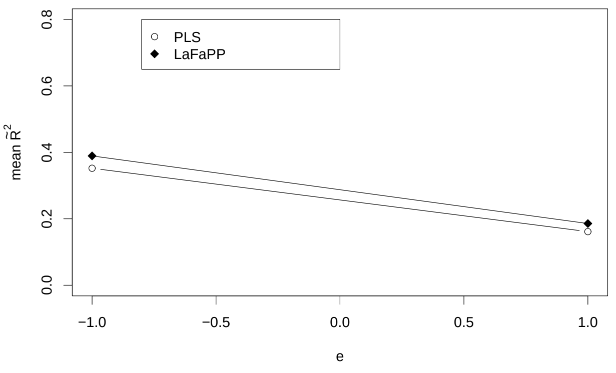

Figure 5: Comparison of LaFaPP and PLS concerning $\tilde{R}^2$for the factor e

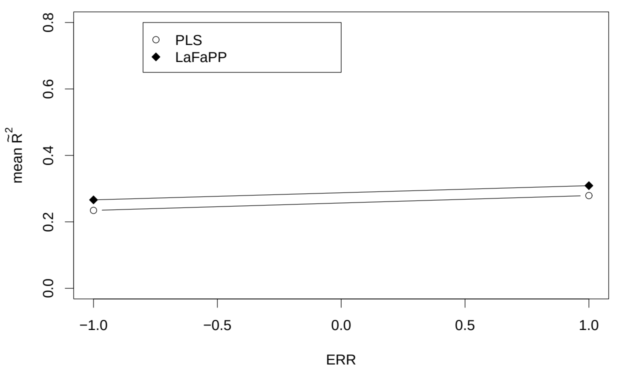

Figure 6: Comparison of LaFaPP and PLS concerning $\tilde{R}^2$ for the factor 'covariance structure'

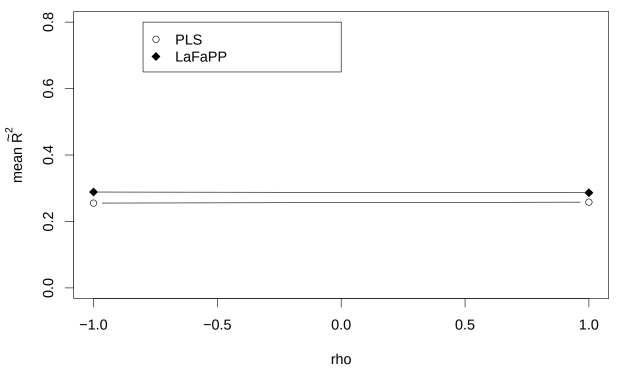

Figure 7: Comparison of LaFaPP and PLS concerning $\tilde{R}^2$ for the factor ρ

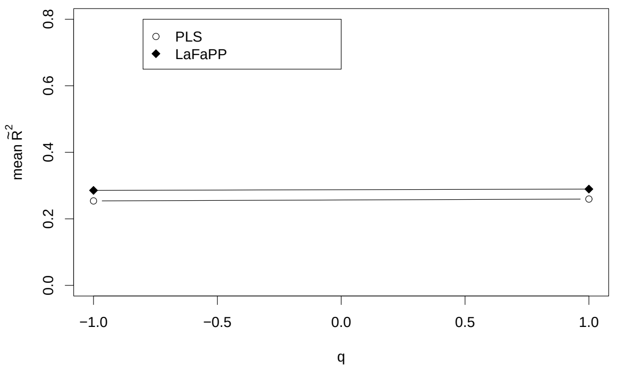

Figure 8: Comparison of LaFaPP and PLS concerning $\tilde{R}^2$ for the factor q

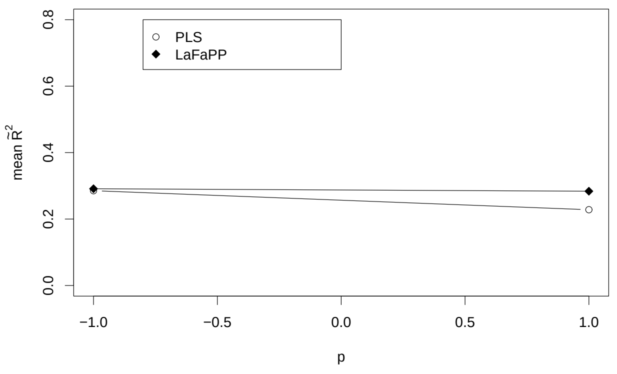

Figure 9: Comparison of LaFaPP and PLS concerning $\tilde{R}^2$ for the factor p

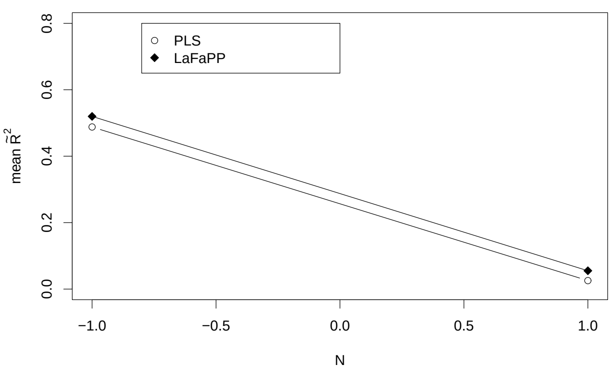

Figure 10: Comparison of LaFaPP and PLS concerning $\tilde{R}^2$ for the factor n