AUSTRIAN JOURNAL OF STATISTICS
Volume 34 (2005), Number 2, 117–126

A Comparison of Three Procedures for Robust PCA in High Dimensions

S. Engelen, M. Hubert, and K. Vanden Branden
Katholieke Universiteit Leuven, Belgium

Abstract: In this paper we compare three procedures for robust Principal Components Analysis (PCA). The first method is called ROBPCA (see Hubert et al., 2005). It combines projection pursuit ideas with robust covariance estimation. The original algorithm for its computation is designed to construct an optimal PCA subspace of a fixed dimension k. If instead the optimal PCA subspace is searched within a whole range of dimensions k, this algorithm is not computationally efficient. Hence we present an adjusted algorithm that yields several PCA models in one single run. A different approach is the LTS-subspace estimator (see Wolbers, 2002; Maronna, 2005). It seeks for the subspace that minimizes an objective function based on the squared orthogonal distances of the observations to this subspace. It can be computed in analogy with the computation of the LTS regression estimator (see Rousseeuw and Van Driessen, 2000). The three approaches are compared by means of a simulation study.

Keywords: Robustness, PCA, LTS, ROBPCA.

1 Introduction

We compare different procedures for robust Principal Components Analysis (PCA). The first method ROBPCA (ROBust PCA) (see Hubert et al., 2005) combines projection pursuit ideas with the MCD covariance estimator (see Rousseeuw, 1984) and is extremely suitable for high-dimensional data (when the dimension p is larger than half the number of observations n). In Section 2 we describe two different algorithms for its computation. The first one corresponds with the original algorithm described in Hubert et al. (2005). It is used to find the optimal PCA-subspace of a certain fixed dimension $k \ll p$. Consequently the output only includes the k eigenvectors and eigenvalues of this optimal k-dimensional subspace. When the PCA results are required for several dimensions, typically $k = 1, \ldots, k_{max}$, this algorithm is not very efficient as it needs to run the whole procedure for each k. However, some computations are common to all dimensions k and thus should not be repeated. This leads to the ROBPCA-k_{max} algorithm, described in Section 2.2.

As an alternative to ROBPCA, we also consider the LTS-subspace estimator of Wolbers (2002); Maronna (2005) which seeks for a subspace that minimizes an objective function based on the squared orthogonal distances of the observations. The corresponding algorithm is described in Section 3.

In Section 4 a comparison of the three procedures is made by means of a simulation study, whereas Section 5 concludes.

2 The ROBPCA Method

2.1 The Original ROBPCA Algorithm

For a complete description of the ROBPCA method we refer to Hubert et al. (2005). Here, we only indicate the major steps of the algorithm.

The first step of ROBPCA consists of performing a singular value decomposition of the data in order to project the observations on the space spanned by themselves. If p is much larger than n this step already yields a huge dimension reduction, whereas no information is lost.

Then a measure of outlyingness is defined for every sample by projecting all the points on many univariate directions through two data points. On this line, the standardized distance of each point to the center of the data is determined. For every observation the largest distance over all directions is stored which is called the outlyingness. The $h(> n/2)$ points with smallest outlyingness are then gathered in an h-subset H_0 and their empirical covariance matrix $\hat{\Sigma}_0$ is computed. The choice of h determines the robustness as well as the efficiency of the method. The larger h is taken, the more accurate ROBPCA will be, but the less robust. The default value of h is set equal to 75% of the total number of observations n.

As initial robust k-dimensional subspace estimate, ROBPCA considers the subspace V_0 spanned by the k dominant eigenvectors of $\hat{\Sigma}_0$. Note that the influence function of this estimator is bounded, as shown in Debruyne and Hubert (2005).

To increase the efficiency a first reweighting is then carried out. All data points which are close to V_0 receive weight 1, whereas the observations far away from it receive zero weight. More precisely, for each observation its orthogonal distance is computed as

$$\text{OD}(\boldsymbol{x}_i) = \|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_{i,k}\|$$

with $\hat{\boldsymbol{x}}_{i,k}$ the projection of $\boldsymbol{x}_i$ in the subspace V_0. In Hubert et al. (2005) it is argued that the orthogonal distances to the power $2/3$ are approximately normally distributed $N(\mu, \sigma^2)$. Consequently, observations with OD smaller than $(\hat{\mu} + \hat{\sigma}\Phi^{-1}(0.975))^{3/2}$, with $\hat{\mu}$ and $\hat{\sigma}$ robust estimates of μ and σ, are retained and their covariance matrix $\hat{\Sigma}_1$ is computed. An improved robust subspace estimate is now obtained as the subspace V_1 spanned by the k dominant eigenvectors of $\hat{\Sigma}_1$. Note that this reweighting step is not yet described in the original paper (Hubert et al., 2005) but now has been added as it turned out to improve the results.

In the next step all the n data points are projected on V_1. Within this subspace a slightly adapted version of the FAST-MCD algorithm (see Rousseeuw and Van Driessen, 1999) is performed in order to find a robust center and a robust covariance estimate of the projected samples. This means that the algorithm first looks for an optimal h-subset H_1 that contains the h observations of which the determinant of their empirical covariance matrix is minimal. The center $\hat{\boldsymbol{\mu}}_{raw}$ and scatter matrix $\hat{\Sigma}_{raw}$ of these h observations are calculated. Next, a second reweighting procedure is performed to increase further the efficiency of the algorithm. Weights are based on the robust distance of every point with respect to $\hat{\boldsymbol{\mu}}_{raw}$ and $\hat{\Sigma}_{raw}$. Samples obtain a zero weight if their robust distance is too large. Regular observation receive weight 1. Finally, the reweighted center $\hat{\boldsymbol{\mu}}$ and covari-

ance matrix $\hat{\Sigma}$ are determined as the classical mean and covariance matrix of the weighted observations.

The last stage of ROBPCA consists of representing $\hat{\boldsymbol{\mu}}$ and $\hat{\Sigma}$ in the original p-dimensional data space. The robust principal components then correspond with the eigenvectors of $\hat{\Sigma}$.

Note that V_1, the k-dimensional subspace spanned by the ROBPCA eigenvectors, depends on k through the first reweighting step. Hence, V_1 will in general not be nested into the $k+1$-dimensional subspace that is found by applying ROBPCA with $k+1$ components. Consequently, also the resulting principal components are not subsets of each other.

2.2 The Adjusted Algorithm for Several Components

When analyzing real data, we usually do not know in advance how many components k we need to select. In Hubert et al. (2005) it was suggested to make a scree plot of the eigenvalues of $\hat{\Sigma}_0$, but this is a too rough approximation. A more refined technique is based on the Predicted Residual Error Sum of Squares (PRESS) value (see Joliffe, 1986), of which the robust version is defined for a certain dimension k as in Hubert and Engelen (2004) and in Engelen and Hubert (2004) :

$$\text{R-PRESS}_k = \sum_{j=1}^{n} w_i \|\boldsymbol{x}_j - \hat{\boldsymbol{x}}_{j,k}\|^2$$

Here, the index j runs over all observations from a test set, and $\hat{\boldsymbol{x}}_{j,k}$ denotes the projection of test sample $\boldsymbol{x}_j$ in the k-dimensional PCA subspace. The weights w_i are added to obtain a robust PRESS-value. For more details see Hubert and Engelen (2004). If a test set is not available, a cross-validated version of the R-PRESS statistic can also be used. The optimal number of components is then selected as the k for which the R-PRESS value is small enough.

One sees from the definition of the R-PRESS value that the ROBPCA algorithm has to be run for every dimension $k = 1, \ldots, k_{max}$. It becomes even worse to compute the cross-validated R-PRESS as then also every observation $i\,(i = 1, \ldots, n)$ at a time has to be removed from the data set. This results in a very time consuming approach, especially for robust resampling algorithms such as ROBPCA.

Therefore we propose the ROBPCA-k_{max} algorithm as a much faster alternative to ROBPCA. It proceeds as follows:

1. As the first step of ROBPCA (the singular value decomposition of the data) and the computation of the outlyingness do not depend on k, we compute it just once. We obtain again $\hat{\Sigma}_0$ as the covariance matrix of the h data points with smallest outlyingness.

2. Then we project the data on the k_{max} dominant eigenvectors of $\hat{\Sigma}_0$, perform the first reweighting based on the orthogonal distances, compute $\hat{\Sigma}_1$ and define V_1 as the k_{max}-dimensional subspace spanned by the dominant eigenvectors of $\hat{\Sigma}_1$. Within this subspace, we compute the MCD estimates of location $\hat{\mu}$ and covariance $\hat{\Sigma}$.

3. The optimal PCA subspace of dimension k is now defined as the space spanned by the k dominant eigenvectors of $\hat{\Sigma}$.

We see that this approach avoids the computation of the outlyingness and the MCD covariance matrix k_{max} times. Moreover, the principal components obtained by applying ROBPCA-k_{max} with k components are now a subset of the components that are found by applying ROBPCA-k_{max} with more than k components.

In Hubert and Engelen (2004) the ROBPCA-k_{max} approach is used to obtain a fast method for the computation of the cross-validated R-PRESS value. The approximate method then needs e.g. only 19.03 seconds for a data set with $n = 100$ observations and $p = 500$ variables versus 5191.1 seconds for the naive approach, whereas the R-PRESS curves are very similar. Also robust calibration methods such as robust principal component regression and robust partial least squares benefit from ROBPCA-k_{max} as shown in Engelen and Hubert (2005).

On the other hand, this algorithm has the disadvantage that it computes the MCD estimator in k_{max} dimensions. This becomes less precise, less robust and more time consuming if k_{max} is chosen too large. Of course it depends on the number of observations, but our experience is that with moderate data sets of sizes up to 100, the differences between the original ROBPCA algorithm and its adjusted version remain small if k_{max} is at most 10 and if the curse of dimensionality is taken into account, i.e. $n/k_{max} > 5$. This will also be illustrated in Section 4. Moreover in many data sets in chemometrics and bio-informatics with thousands of variables a small number of components is usually sufficient to summarize the data well. Hence we always set $k_{max} \leq 10$.

Another disadvantage of ROBPCA-k_{max} is the estimation of the eigenvalues. The MCD covariance estimator is always multiplied with a consistency factor, which decreases with the dimension. If the optimal subspace has a dimension k_{opt}, much smaller than k_{max}, the estimate is not inflated enough. This affects the robust distances within the subspace, and consequently the reweighted MCD estimates. Hence, we propose to use the consistency factor based on $k_{max}/2$ components instead of the factor associated with k_{max}. But when ROBPCA-k_{max} is used to compute R-PRESS values, which then allows to select a certain k, we apply ROBPCA for k components with the consistency factor for k-dimensional variables.

3 The LTS-subspace Estimator

The LTS-subspace estimator is introduced in Rousseeuw and Leroy (1987) and studied in Wolbers (2002) and in Maronna (2005). It is based on the connection between classical PCA and subspace estimation. The subspace spanned by the k first principal components (the dominant eigenvectors of the empirical covariance matrix of the data) has the property that it also minimizes the sum of the squared orthogonal distances of the observations to that subspace. From this point of view, a robust alternative can be found in searching the h-subset that minimizes the objective function:

$$\sum_{i=1}^{h} \mathrm{OD}^2_{(i)}$$

with $\text{OD}^2_{(1)} \leq \text{OD}^2_{(2)} \leq \cdots \leq \text{OD}^2_{(n)}$ the ordered squared orthogonal distances and h defined as in ROBPCA. The resulting subspace is called the LTS-subspace (of dimension k). When $k = p - 1$, it coincides with robust orthogonal regression.

For its computation, an algorithm can be developed which is very similar to the FAST-LTS algorithm to compute the LTS regression estimator (see Rousseeuw and Van Driessen, 2000). It starts by drawing many random $(k+1)$-subsets. A classical PCA is performed on these $(k+1)$ points and the orthogonal distances of all the observations are calculated with respect to this k-dimensional subspace. Then for each subspace the h points with smallest orthogonal distance are stored, and used to compute the classical PCA subspace of dimension k. Next, the orthogonal distances with respect to this subspace are calculated for all data, and the h points with the smallest orthogonal distance are again retained. This procedure is guaranteed to converge. The optimal h-subset is then defined as the h-subset which has the lowest objective function over all the obtained h-subsets. Finally classical PCA is performed on this optimal h-subset in order to obtain eigenvectors and eigenvalues. Remark that several time saving techniques as in the FAST-LTS algorithm can be included as well, but they will not be discussed here.

4 Simulations

In this section a simulation study is performed to gain insight in the performance of the three robust methods compared with classical PCA. Also the effect of k_{max} on the ROBPCA-k_{max} results is studied in more detail.

All the data are generated from the following contamination model :

$$(1-\epsilon)N_p(\mathbf{0}, \Sigma) + \epsilon N_p(\tilde{\boldsymbol{\mu}}, \Sigma) \tag{1}$$

with ϵ the percentage outliers included in the data. This means that the bulk of the data is normally distributed with center $\boldsymbol{\mu} = \mathbf{0}$ and covariance matrix $\Sigma = diag(10, 8, 2, 1, \ldots)$, where the dots indicate eigenvalues that are negligibly small and $diag$ represents a diagonal matrix.

By varying the values for ϵ, n, p, k, k_{max} and $\tilde{\boldsymbol{\mu}}$ different simulation settings are created. Here is an overview of the values assigned to these parameters :

- The data matrix is $n = 40$ by $p = 200$, and $n = 100$ by $p = 1000$.
- The percentage of outliers ϵ is set to 0%, 10% and 20%.
- The quantile h used in all the algorithms is set to roughly $0.75n$ (the default) in order to obtain an outlier resistance of approximately 25%. As we prefer to use the same h value for all algorithms, we take $h = \min\{[2[(n+k_{max}+1)/2] - n + 2(n - [(n+k_{max}+1)/2])\alpha], n\}$ with $\alpha = 0.75$, $k_{max} = 8$ if $n = 40$ and $k_{max} = 10$ if $n = 100$. This formula for h yields an interpolated value between the minimal one $[(n+k_{max}+1)/2]$ if $\alpha = 0.5$ and the maximal value n if $\alpha = 1$. Here, it implies that $h = 32$ if $n = 40$ and $h = 77$ if $n = 100$.
- In ROBPCA-k_{max}, the maximal number of components k_{max} is taken as 5 or 8 if $n = 40$ and as 7, 10 or 15 if $n = 100$. No higher values are considered to avoid the curse of dimensionality.

- The number of principal components is chosen as $k = 2$ and $k = 4$. For $n = 40$, two components account for 79% of the total variance, whereas four components explain 92% of the variance. For $n = 100$ these percentages are 84%, and 98% respectively.
- The center of the outliers $\tilde{\boldsymbol{\mu}}$ is set to $\tilde{\boldsymbol{\mu}} = (0, 0, 0, 0, 15, 0, \ldots, 0)'$ to generate orthogonal outliers (O.O), and to $\tilde{\boldsymbol{\mu}} = (15, 15, 15, 15, 15, 0, \ldots, 0)'$ to obtain bad leverage (B.L.) points (with respect to the 2-dimensional and 4-dimensional subspaces).

For every setting we have constructed 100 data sets. We did not take a higher value because the LTS-estimator is rather computationally intensive, as we will see further on. To evaluate the simulation results, we have computed the maximal angle between the space spanned by the estimated principal components and E_k, where E_k is the subspace spanned by the k dominant eigenvectors of Σ. Thus, $E_k = \text{span}\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_k\}$ with $\boldsymbol{e}_j$ the jth column of $I_{p,k}$ (the subscripts indicate the dimensions of the matrix). A measure to calculate this angle is proposed by Krzanowski (see Krzanowski, 1979) and is called *maxsub* (see Hubert et al., 2005). Let the loading matrix $P_{p,k}$ contain the estimated eigenvectors columnwise, then $maxsub$ is computed as

$$\text{maxsub} = arccos(\sqrt{\lambda_k})$$

where λ_k is the smallest eigenvalue of $I'_{k,p}P_{p,k}P'_{k,p}I_{p,k}$. It represents the largest angle between a vector in E_k and the vector most parallel to it in the estimated PCA subspace. To standardize this value, we have divided it by $\frac{\pi}{2}$. The closer this Krzanowski measure is to 0, the more accurate the method.

From the simulation results in Tables 1-4, we conclude that at uncontaminated data both PCA and the three robust methods work well, with (not surprisingly) the best performance achieved by classical PCA. In all situations where outliers were included, we see that PCA immediately breaks down, whereas ROBPCA always yields very accurate results and clearly outperforms the two other robust approaches. These robust results indicate that the k-dimensional subspaces V_0 and V_1 are not influenced by outliers, nor is the MCD-estimator applied to the projected data in V_1.

With 10% contamination, both LTS-subspace and ROBPCA-k$_{max}$ still attain robust results, but they break down in some situations with $\varepsilon = 20\%$. With this large amount of contamination, we notice for the LTS-subspace estimator a high bias when $n = 40$ and $k = 4$, but the outliers are no longer a cause for concern when $n = 100$.

With ROBPCA-k_{max} breakdown occurs in several situations where a large value of k_{max} is taken (equal to 8 if $n = 40$ and to 15 when $n = 100$). When k_{max} is small to moderate, we obtain low values for *maxsub*. From these results and the very accurate estimates of ROBPCA-k_{max} for smaller k_{max}-values, we can conclude that this method can be used as a robust alternative for PCA, at least when selecting k_{max} small enough.

Finally, we compare the mean computation times (in seconds) of the three robust algorithms (on a Pentium IV with 2.40Ghz). Table 5 shows that their computation time is comparable in small dimensions, but the LTS-subspace method becomes significantly slower when the data matrix becomes larger.

Further we notice that ROBPCA is slightly faster than ROBPCA-k_{max}, which is to be expected as ROBPCA computes the MCD only in a k-dimensional subspace. However,

Table 1: The simulation results for $n = 40, p = 200$ and $k = 2$.

		PCA	ROBPCA	LTS-subspace	ROBPCA-k_{max}	
					$k_{max} = 5$	$k_{max} = 8$
$\epsilon = 0\%$		0.109	0.151	0.151	0.155	0.140
$\epsilon = 10\%$	B.L.	0.586	0.143	0.145	0.139	0.136
	O.O.	0.913	0.143	0.144	0.151	0.140
$\epsilon = 20\%$	B.L.	0.600	0.128	0.227	0.122	0.130
	O.O.	0.946	0.128	0.130	0.153	0.158

Table 2: The simulation results for $n = 40, p = 200$ and $k = 4$.

		PCA	ROBPCA	LTS-subspace	ROBPCA-k_{max}	
					$k_{max} = 5$	$k_{max} = 8$
$\epsilon = 0\%$		0.148	0.182	0.195	0.186	0.1918
$\epsilon = 10\%$	B.L.	0.340	0.179	0.229	0.182	0.188
	O.O.	0.980	0.181	0.188	0.196	0.206
$\epsilon = 20\%$	B.L.	0.399	0.170	0.368	0.170	0.171
	O.O.	0.984	0.170	0.480	0.204	0.472

Table 3: The simulation results for $n = 100, p = 1000$ and $k = 2$.

		PCA	ROBPCA	LTS-subspace	ROBPCA-k_{max}		
					$k_{max} = 7$	$k_{max} = 10$	$k_{max} = 15$
$\epsilon = 0\%$		0.061	0.079	0.097	0.069	0.067	0.067
$\epsilon = 10\%$	B.L.	0.567	0.078	0.090	0.070	0.069	0.069
	O.O.	0.947	0.080	0.089	0.069	0.069	0.069
$\epsilon = 20\%$	B.L.	0.575	0.076	0.085	0.070	0.071	0.487
	O.O.	0.971	0.076	0.076	0.071	0.074	0.082

Table 4: The simulation results for $n = 100, p = 1000$ and $k = 4$.

		PCA	ROBPCA	LTS-subspace	ROBPCA-k_{max}		
					$k_{max} = 7$	$k_{max} = 10$	$k_{max} = 15$
$\epsilon = 0\%$		0.047	0.056	0.063	0.057	0.058	0.060
$\epsilon = 10\%$	B.L.	0.391	0.054	0.073	0.055	0.056	0.057
	O.O.	0.988	0.054	0.058	0.055	0.056	0.057
$\epsilon = 20\%$	B.L.	0.392	0.055	0.070	0.055	0.055	0.082
	O.O.	0.991	0.055	0.056	0.055	0.059	0.538

when several solutions are required, for example the components for $k = 2$ and $k = 4$, the computation times for ROBPCA need to be cumulated, whereas ROBPCA-k_{max} yields those components in one single run of the algorithm.

Table 5: The mean computation time in seconds of ROBPCA, ROBPCA-k_{max} and the LTS-subspace.

	ROBPCA		ROBPCA-kmax		LTS-subspace
$n = 40, p = 200$	$k = 2$	3.84	$k_{max} = 5$	3.99	5.58
	$k = 4$	3.92	$k_{max} = 8$	4.32	
$n = 100, p = 1000$	$k = 2$	4.26	$k_{max} = 7$	4.62	19.49
	$k = 4$	4.37	$k_{max} = 10$	4.88	
			$k_{max} = 15$	5.35	

5 Conclusions

We have shown that the original ROBPCA method is very robust as it can withstand many types of contamination. To save computation time (for example in order to select the number of principal components), we have constructed the fast ROBPCA-k_{max} method. In our simulations we found that ROBPCA-k_{max} performs almost as well as the original ROBPCA algorithm, unless k_{max} is set too large. This makes ROBPCA-k_{max} a valuable alternative to ROBPCA, especially to compute cross-validated statistics. We also compared these ROBPCA methods with the LTS-subspace estimator. This method appears to be less robust in some particular contamination settings, and it requires more computation time. We thus conclude that ROBPCA and ROBPCA-k_{max} are more in favor to use.

The ROBPCA method is available at the website

`http://www.wis.kuleuven.ac.be/stat/robust.html`

as part of LIBRA, the Matlab Library for Robust Analysis, (see Verboven and Hubert, 2005). In the future also ROBPCA-k_{max} will be integrated into this library.

Acknowledgements

We would like to thank Randy Pell for his interesting discussions which has led to the construction of ROBPCA-k_{max}.

References

M. Debruyne and M. Hubert. The influence function of Stahel-Donoho type methods for robust PCA. 2005. In preparation.

S. Engelen and M. Hubert. Fast cross-validation for robust PCA. In J. Antoch, editor, *Proceedings in Computational Statistics*, pages 989–996, Heidelberg, 2004. Springer, Physica-Verlag.

S. Engelen and M. Hubert. Fast model selection for robust calibration methods. *Analytica Chemica Acta*, 2005. To appear.

M. Hubert and S. Engelen. Fast cross-validation for high-breakdown resampling algorithms for PCA, 2004. Submitted.

M. Hubert, P. J. Rousseeuw, and K. Vanden Branden. ROBPCA: a new approach to robust principal components analysis. *Technometrics*, 47:64–79, 2005.

I.T. Joliffe. *Principal Component Analysis*. Springer, New York, 1986.

W.J. Krzanowski. Between-groups comparison of principal components. *Journal of the American Statistical Association*, 74:703–707, 1979.

R. A. Maronna. Principal components and orthogonal regression based on robust scales. *Technometrics*, 2005. To appear.

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

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

P. J. Rousseeuw and K. Van Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41:212–223, 1999.

P. J. Rousseeuw and K. Van Driessen. An algorithm for positive-breakdown methods based on concentration steps. In W. Gaul, O. Opitz, and M. Schader, editors, *Data Analysis: Scientific Modeling and Practical Application*, pages 335–346, New York, 2000. Springer-Verlag.

S. Verboven and M. Hubert. LIBRA: a Matlab library for robust analysis. *Chemometrics and Intelligent Laboratory Systems*, 75:127–136, 2005.

M. Wolbers. *Linear unmixing of multivariate observations*. PhD thesis, ETH Zürich, 2002.

Authors' address:

Sanne Engelen
Mia Hubert
Karlien Vanden Branden
Department of Mathematics
Katholieke Universiteit Leuven
W. De Croylaan 54
B-3001 Leuven

Tel. +32 16 322023
Fax +32 16 322831
E-mail: mia.hubert@wis.kuleuven.ac.be
http://www.wis.kuleuven.ac.be/stat/robust.html