Technometrics

ISSN: 0040-1706 (Print) 1537-2723 (Online) Journal homepage: https://www.tandfonline.com/loi/utch20

MacroPCA: An All-in-One PCA Method Allowing for Missing Values as Well as Cellwise and Rowwise Outliers

Mia Hubert, Peter J. Rousseeuw & Wannes Van den Bossche

To cite this article: Mia Hubert, Peter J. Rousseeuw & Wannes Van den Bossche (2019) MacroPCA: An All-in-One PCA Method Allowing for Missing Values as Well as Cellwise and Rowwise Outliers, Technometrics, 61:4, 459-473, DOI: 10.1080/00401706.2018.1562989

To link to this article: https://doi.org/10.1080/00401706.2018.1562989

TECHNOMETRICS
2019, VOL. 61, NO. 4, 459–473
https://doi.org/10.1080/00401706.2018.1562989

Taylor & Francis
Taylor & Francis Group

MacroPCA: An All-in-One PCA Method Allowing for Missing Values as Well as Cellwise and Rowwise Outliers

Mia Hubert, Peter J. Rousseeuw, and Wannes Van den Bossche

Department of Mathematics, KU Leuven, Leuven, Belgium

ABSTRACT

Multivariate data are typically represented by a rectangular matrix (table) in which the rows are the objects (cases) and the columns are the variables (measurements). When there are many variables one often reduces the dimension by principal component analysis (PCA), which in its basic form is not robust to outliers. Much research has focused on handling rowwise outliers, that is, rows that deviate from the majority of the rows in the data (e.g., they might belong to a different population). In recent years also cellwise outliers are receiving attention. These are suspicious cells (entries) that can occur anywhere in the table. Even a relatively small proportion of outlying cells can contaminate over half the rows, which causes rowwise robust methods to break down. In this article, a new PCA method is constructed which combines the strengths of two existing robust methods to be robust against both cellwise and rowwise outliers. At the same time, the algorithm can cope with missing values. As of yet it is the only PCA method that can deal with all three problems simultaneously. Its name MacroPCA stands for PCA allowing for Missingness And Cellwise & Rowwise Outliers. Several simulations and real datasets illustrate its robustness. New residual maps are introduced, which help to determine which variables are responsible for the outlying behavior. The method is well-suited for online process control.

ARTICLE HISTORY
Received June 2018
Accepted December 2018

KEYWORDS
Detecting deviating cells; Outlier map; Principal component analysis; Residual map; Robust estimation.

1. Introduction

Real data often contain outliers, which can create serious problems when analyzing it. Many methods have been developed to deal with outliers, often by constructing a fit that is robust to them and then detecting the outliers by their large deviation (distance, residual) from that fit. For a brief overview of this approach see Rousseeuw and Hubert (2018). Unfortunately, most robust methods cannot handle data with missing values, some rare exceptions being Cheng and Victoria-Feser (2002) and Danilov, Yohai, and Zamar (2012). Moreover, they are typically restricted to casewise outliers, which are cases that deviate from the majority. We call these *rowwise outliers* because multivariate data are typically represented by a rectangular matrix in which the rows are the cases and the columns are the variables (measurements). In general, robust methods require that fewer than half of the rows are outlying, see, for example, Lopuhaä and Rousseeuw (1991). However, recently a different type of outliers, called *cellwise outliers*, have received much attention (Alqallaf et al. 2009; Van Aelst, Vandervieren, and Willems 2012; Agostinelli et al. 2015). These are suspicious cells (entries) that can occur anywhere in the data matrix. Figure 1 illustrates the difference between these types of outliers. The regular cells are shown in gray, whereas black means outlying. Rows 3 and 7 are rowwise outliers, and the other rows contain a fairly small percentage of cellwise outliers. As in this example, a small proportion of outlying cells can contaminate over half the rows, which causes most methods to break down. This effect is at its worst when the dimension (the number of columns) is high.

In high-dimensional situations, which are becoming increasingly common, one often applies principal component analysis (PCA) to reduce the dimension. However, the classical PCA (CPCA) method is not robust to either rowwise or cellwise outliers. Robust PCA methods that can deal with rowwise outliers include Croux and Ruiz-Gazen (2005), Hubert, Rousseeuw, and Verboven (2002), Locantore et al. (1999), Maronna (2005), and the ROBPCA method (Hubert, Rousseeuw, and Vanden Branden 2005). The latter method combines projection pursuit ideas with robust covariance estimation.

To deal with missing values, Nelson, Taylor, and MacGregor (1996) and Kiers (1997) developed the *iterative classical PCA algorithm* (ICPCA), see Walczak and Massart (2001) for a tutorial. The ICPCA follows the spirit of the EM algorithm. It starts by replacing the missing values by initial estimates such as the columnwise means. Then it iteratively fits a CPCA, yielding scores that are transformed back to the original space resulting in new estimates for the missing values, until convergence.

Serneels and Verdonck (2008) proposed a rowwise robust PCA method that can also cope with missing values. We will call this method MROBPCA (ROBPCA for missing values) as its key idea is to combine the ICPCA and ROBPCA methods. MROBPCA starts by imputing the NAs by robust initial estimates. The main difference with the ICPCA algorithm is that in

CONTACT Peter J. Rousseeuw *peter@rousseeuw.net* Department of Mathematics, KU Leuven, Leuven, BE-3001, Belgium.
Color versions of one or more of the figures in the article can be found online at *www.tandfonline.com/r/TECH*.
Supplementary materials for this article are available online. Please go to *www.tandfonline.com/r/TECH*.

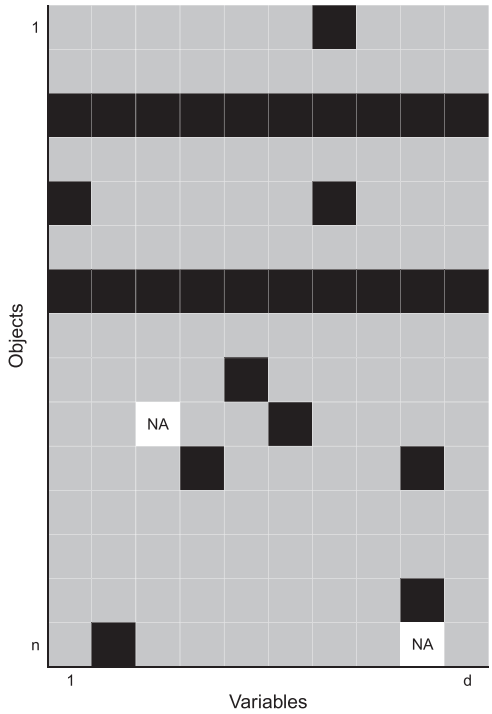

Figure 1. Data matrix with missing values and cellwise and rowwise contamination.

each iteration the PCA model is fit by ROBPCA, which yields different imputations and flags rowwise outliers.

As of yet there are no PCA methods that can deal with cellwise outliers in combination with rowwise outliers and NAs. This article aims to fill that gap by constructing a new method called MacroPCA, where "Macro" stands for Missingness And Cellwise and Rowwise Outliers. It starts by applying a multivariate method called DetectDeviatingCells (Rousseeuw and Van den Bossche 2018) for detecting cellwise outliers, which provides initial imputations for the outlying cells and the NAs as well as an initial measure of rowwise outlyingness. In the next steps, MacroPCA combines ICPCA and ROBPCA to protect against rowwise outliers and to create improved imputations of the outlying cells and missing values. MacroPCA also provides graphical displays to visualize the different types of outliers. R code for MacroPCA is publicly available (Section 8).

2. The MacroPCA Algorithm

2.1. Model

The data matrix is denoted as $\boldsymbol{X}_{n,d}$ in which the subscripts are the number of rows (cases) n and the number of columns (variables) d. In the absence of outliers and missing values the goal is to represent the data in a lower dimensional space, that is,

$$\boldsymbol{X}_{n,d} = \mathbf{1}_n \boldsymbol{\mu}_d' + \boldsymbol{\mathcal{T}}_{n,k}(\boldsymbol{\mathcal{P}}_{d,k})' + \boldsymbol{\mathcal{E}}_{n,d}, \tag{1}$$

with $\mathbf{1}_n$ the column vector with all n components equal to 1, $\boldsymbol{\mu}_d$ the d-variate column vector of location, $\boldsymbol{\mathcal{T}}_{n,k}$ the $n \times k$ score matrix, $\boldsymbol{\mathcal{P}}_{d,k}$ the $d \times k$ loadings matrix whose columns span the PCA subspace, and $\boldsymbol{\mathcal{E}}_{n,d}$ the $n \times d$ error matrix. The reduced dimension k can vary from 1 to d but we assume that k is low. The $\boldsymbol{\mu}_d$, $\boldsymbol{\mathcal{T}}_{n,k}$, and $\boldsymbol{\mathcal{P}}_{d,k}$ are unknown, and estimates of them will be denoted by $\boldsymbol{m}_d$, $\boldsymbol{T}_{n,k}$, and $\boldsymbol{P}_{d,k}$.

Several realities complicate this simple model. First, the data matrix may not be fully observed, that is, some cells x_{ij} may be missing. Here, we assume that they are *missing at random* (MAR), meaning that the missingness of a cell is unrelated to the value the cell would have had, but may be related to the values of other cells in the same row; see, for example, Schafer and Graham (2002). This is the typical assumption underlying EM-based methods, such as ICPCA and MROBPCA that are incorporated in our proposal.

Second, the data may contain rowwise outliers, for example, cases from a different population. The existing rowwise robust methods require that fewer than half of the rows are outlying, so we make the same assumption here.

Third, cellwise outliers may occur as described in Section 1. The outlying cells may be imprecise, incorrect or just unusual. Outlying cells do not necessarily stand out in their column because the correlations between the columns matter as well, so these cells may not be detectable by simple univariate outlier detection methods. There can be many cellwise outliers, and in fact each row may contain one or more outlying cells.

2.2. Dealing With Missing Values and Cellwise and Rowwise Outliers

We propose the MacroPCA algorithm for analyzing data that may contain one or more of the following issues: missing values, cellwise outliers, and rowwise outliers. Throughout the algorithm we will use the following two notations:

- the *NA-imputed matrix* $\mathring{\boldsymbol{X}}_{n,d}$ only imputes the missing values of $\boldsymbol{X}_{n,d}$;
- the *cell-imputed matrix* $\dot{\boldsymbol{X}}_{n,d}$ has imputed values for the outlying cells that do not belong to outlying rows, and for all missing values.

Both of these matrices still have n rows. Neither is intended to simply replace the true data matrix $\boldsymbol{X}_{n,d}$. Note that $\dot{\boldsymbol{X}}_{n,d}$ does not try to impute outlying cells inside outlying rows, which would mask these rows in subsequent computations.

Since we do not know in advance which cells and rows are outlying, the set of flagged cellwise and rowwise outliers (and hence $\mathring{\boldsymbol{X}}_{n,d}$ and $\dot{\boldsymbol{X}}_{n,d}$) will be updated in the course of the algorithm.

The first part of MacroPCA is the DetectDeviatingCells (DDC) algorithm. The description of this method can be found in Rousseeuw and Van den Bossche (2018) and in Section 1 of the supplementary material. The main purpose of the DDC method is to detect cellwise outliers. DDC outputs their positions $I_{c,\mathrm{DDC}}$ as well as imputations for these outlying cells and any missing values. It also yields an initial outlyingness measure on the rows, which is, however, not guaranteed to flag all outlying rows. The set of flagged rows $I_{r,\mathrm{DDC}}$ will be improved in later steps.

The second part of MacroPCA constructs principal components along the lines of the ICPCA algorithm but employing a version of ROBPCA (Hubert, Rousseeuw, and Vanden Branden

2005) to fit subspaces. It consists of the following steps, with all notations listed in Section 2 of the supplementary material.

1. Projection pursuit. The goal of this step is to provide an initial indication of which rows are the least outlying. For this ROBPCA starts by identifying the $h < n$ least outlying rows by a projection pursuit procedure. We write $0.5 \leqslant \alpha = h/n < 1$. This means that we can withstand up to a fraction $1 - \alpha$ of outlying rows. To be on the safe side the default is $\alpha = 0.5$.

   However, due to cellwise outliers there may be far fewer than h uncontaminated rows, so we cannot apply this step to the original data $\boldsymbol{X}_{n,d}$. We also cannot use the entire imputed matrix $\tilde{\boldsymbol{X}}_{n,d}$ obtained from DDC in which all outlying cells are imputed, even those in potentially outlying rows, as this could mask outlying rows. Instead we use the cell-imputed matrix $\dot{\boldsymbol{X}}^{(0)}_{n,d}$ defined as follows:

   (a) In all rows we replace the missing values by the values $\mathring{\boldsymbol{x}}^{(0)}_i$ as imputed by DDC.
   (b) Next, in the h rows with the fewest cells flagged by DDC but not in $I_{r,\mathrm{DDC}}$ we impute those flagged cells, that is, $\dot{\boldsymbol{x}}^{(0)}_i = \tilde{\boldsymbol{x}}_i$.

   As in ROBPCA the outlyingness of a point $\dot{\boldsymbol{x}}^{(0)}_i$ is then computed as

$$\mathrm{outl}(\dot{\boldsymbol{x}}^{(0)}_i) = \max_{\boldsymbol{v} \in B} \frac{|\boldsymbol{v}'\dot{\boldsymbol{x}}^{(0)}_i - m_{\mathrm{MCD}}(\boldsymbol{v}'\dot{\boldsymbol{x}}^{(0)}_j)|}{s_{\mathrm{MCD}}(\boldsymbol{v}'\dot{\boldsymbol{x}}^{(0)}_j)} \,, \qquad (2)$$

   where $m_{\mathrm{MCD}}(\boldsymbol{v}'\dot{\boldsymbol{x}}^{(0)}_j)$ and $s_{\mathrm{MCD}}(\boldsymbol{v}'\dot{\boldsymbol{x}}^{(0)}_j)$ are the univariate MCD location and scale estimates (Rousseeuw and Leroy 1987) of $\{\boldsymbol{v}'\dot{\boldsymbol{x}}^{(0)}_1, \ldots, \boldsymbol{v}'\dot{\boldsymbol{x}}^{(0)}_n\}$. The set B contains 250 directions through two data points (or all of them if there are fewer than 250). Finally, the indices of the h rows $\dot{\boldsymbol{x}}^{(0)}_i$ with the lowest outlyingness and not belonging to $I_{r,\mathrm{DDC}}$ are stored in the set H_0.
2. Subspace dimension. Here, we choose the number of principal components. For this we build a new cell-imputed matrix $\dot{\boldsymbol{X}}^{(1)}_{n,d}$ which imputes the outlying cells in the rows of H_0 and imputes the NAs in all rows. This means that $\dot{\boldsymbol{x}}^{(1)}_i = \tilde{\boldsymbol{x}}_i$ for $i \in H_0$, and $\dot{\boldsymbol{x}}^{(1)}_i = \mathring{\boldsymbol{x}}^{(0)}_i$ if $i \notin H_0$. Then we apply classical PCA to the $\dot{\boldsymbol{x}}^{(1)}_i$ with $i \in H_0$. Their mean $\boldsymbol{m}^{(1)}_d$ is an estimate of the center, whereas the spectral decomposition of their covariance matrix yields a loading matrix $\boldsymbol{P}^{(1)}_{d,d}$ and a diagonal matrix $\boldsymbol{L}^{(1)}_{d,d}$ with the eigenvalues sorted from largest to smallest. These eigenvalues can be used to construct a screeplot from which an appropriate dimension k of the subspace can be derived. Alternatively, one can retain a certain cumulative proportion of explained variance, such as 80%. The maximal number of principal components that MacroPCA will consider is the tuning constant $k_{\max}$, which is set to 10 by default.
3. Iterative subspace estimation. This step aims to estimate the k-dimensional subspace fitting the data. As in ICPCA this requires iteration, for $s \geqslant 2$:

   (a) The scores matrix in Equation (1) based on the cell-imputed cases is computed as $\dot{\boldsymbol{T}}^{(s-1)}_{n,k} = (\dot{\boldsymbol{X}}^{(s-1)}_{n,d} - \boldsymbol{1}_n(\boldsymbol{m}^{(s-1)}_d)')\boldsymbol{P}^{(s-1)}_{d,k}$. The predicted data values are set to $\hat{\boldsymbol{X}}^{(s)}_{n,d} = \boldsymbol{1}_n(\boldsymbol{m}^{(s-1)}_d)' + \dot{\boldsymbol{T}}^{(s-1)}_{n,k}(\boldsymbol{P}^{(s-1)}_{d,k})'$. We then update the imputed matrices to $\mathring{\boldsymbol{X}}^{(s)}_{n,d}$ and $\dot{\boldsymbol{X}}^{(s)}_{n,d}$ by replacing the appropriate cells by the corresponding cells of $\hat{\boldsymbol{X}}^{(s)}_{n,d}$. That is, for $\mathring{\boldsymbol{X}}^{(s)}_{n,d}$ we update all the imputations of missing cells, whereas for $\dot{\boldsymbol{X}}^{(s)}_{n,d}$ we update the imputations of the outlying cells in rows of H_0 as well as the NAs in all rows.
   (b) The PCA model is reestimated by applying classical PCA to the $\dot{\boldsymbol{x}}^{(s)}_i$ with $i \in H_0$. This yields a new estimate $\boldsymbol{m}^{(s)}_d$ as well as an updated loading matrix $\boldsymbol{P}^{(s)}_{d,k}$.

   The iterations are repeated until $s = 20$ or until convergence is reached, that is, when the maximal angle between a vector in the new subspace and the vector most parallel to it in the previous subspace is below some tolerance (by default 0.005). Following Krzanowski (1979) this angle is computed as $\arccos(\sqrt{\delta_k})$ where δ_k is the smallest eigenvalue of $(\boldsymbol{P}^{(s)}_{d,k})'\boldsymbol{P}^{(s-1)}_{d,k}(\boldsymbol{P}^{(s-1)}_{d,k})'\boldsymbol{P}^{(s)}_{d,k}$.

   After all iterations we have the NA-imputed matrix $\mathring{\boldsymbol{X}}^{(s)}_{n,d}$, the cell-imputed matrix $\dot{\boldsymbol{X}}^{(s)}_{n,d}$ as well as the estimated center $\boldsymbol{m}^{(s)}_d$ and the loading matrix $\boldsymbol{P}^{(s)}_{d,k}$.
4. Reweighting. In robust statistics one often follows an initial estimate by a reweighting step to improve the statistical efficiency at a low computational cost, see, for example, Rousseeuw and Leroy (1987) and Engelen, Hubert, and Vanden Branden (2005). Here, we use the orthogonal distance of each $\dot{\boldsymbol{x}}^{(s)}_i$ to the current PCA subspace:

$$\dot{\mathrm{OD}}_i = \|\dot{\boldsymbol{x}}^{(s)}_i - \hat{\dot{\boldsymbol{x}}}^{(s)}_i\| = \|\dot{\boldsymbol{x}}^{(s)}_i - (\boldsymbol{m}^{(s)}_d + (\dot{\boldsymbol{x}}^{(s)}_i - \boldsymbol{m}^{(s)}_d)\boldsymbol{P}^{(s)}_{d,k}(\boldsymbol{P}^{(s)}_{d,k})')\| \,.$$

   The orthogonal distances to the power 2/3 are roughly Gaussian except for the outliers (Hubert, Rousseeuw, and Vanden Branden 2005), so we compute the cutoff value

$$c_{\mathrm{OD}} := \left(m_{\mathrm{MCD}}(\{\dot{\mathrm{OD}}_j^{2/3}\}) + s_{\mathrm{MCD}}(\{\dot{\mathrm{OD}}_j^{2/3}\})\Phi^{-1}(0.99)\right)^{3/2} . \qquad (3)$$

   All cases for which $\dot{\mathrm{OD}}_i \leqslant c_{\mathrm{OD}}$ are considered non-outlying with respect to the PCA subspace, and their indices are stored in a set H^*. As before, any $i \in I_{r,DDC}$ is removed from H^*. The final cell-imputed matrix $\dot{\boldsymbol{X}}_{n,d}$ is given by $\dot{x}_{i,j} = \hat{\dot{x}}^{(s)}_{i,j}$ if $i \in H^*$ and $j \in I_{c,DDC}$ and $\dot{x}_{i,j} = \mathring{x}^{(s)}_{i,j}$ otherwise. Applying classical PCA to the n^* rows $\dot{\boldsymbol{x}}_i$ in H^* yields a new center $\boldsymbol{m}^*_d$ and a new loading matrix $\boldsymbol{P}^*_{d,k}$.
5. DetMCD. Now we want to estimate a robust basis of the estimated subspace. The columns of $\boldsymbol{P}^*_{d,k}$ from Step 4 need not be robust, because some of the n^* rows in H^* might be outlying inside the subspace. These so-called good leverage points do not harm the estimation of the PCA subspace but they can still affect the estimated eigenvectors and eigenvalues, as illustrated by a toy example in Section A.1 of the Appendix.

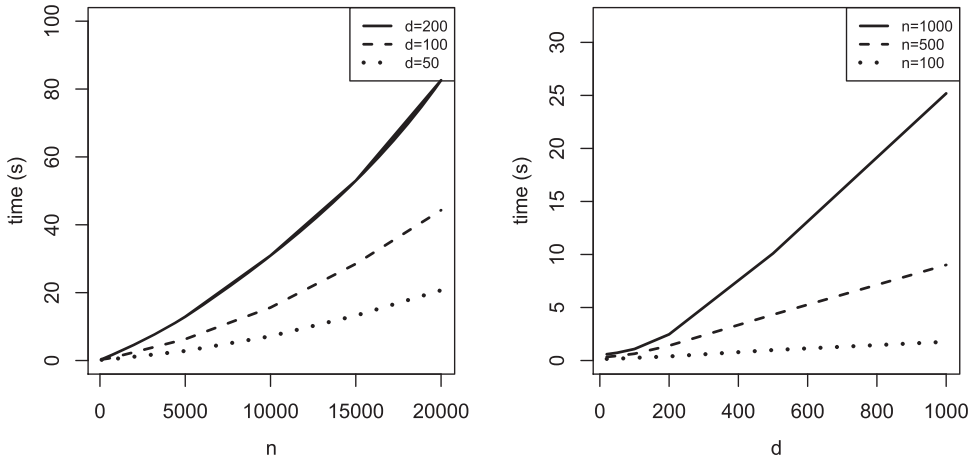

Figure 2. Computation times of MacroPCA in seconds on Intel i7-4800MQ at 2.70 GHz, as a function of the number of cases n (left) and of the dimension d (right).

In this step, we first project the n^* points of H^* onto the subspace, yielding

$$\dot{\boldsymbol{T}}_{n^*,k} = \left(\dot{\boldsymbol{X}}_{n^*,d} - \mathbf{1}_{n^*}\boldsymbol{m}_d^{*\prime}\right)\boldsymbol{P}_{d,k}^{*} \ .$$

Next, the center and scatter matrix of the scores $\dot{\boldsymbol{T}}_{n^*,k}$ are estimated by the DetMCD method of Hubert, Rousseeuw, and Verdonck (2012). This is a fast, robust and deterministic algorithm for multivariate location and scatter, yielding $\boldsymbol{m}_k^{\text{MCD}}$ and $\boldsymbol{S}_{k,k}^{\text{MCD}}$. Its computation is feasible because the dimension k of the subspace is quite low. The spectral decomposition of $\boldsymbol{S}_{k,k}^{\text{MCD}}$ yields a loading matrix $\boldsymbol{P}_{k,k}^{\text{MCD}}$ and eigenvalues $\hat{\lambda}_j$ for $j = 1, \ldots, k$. We set the final center to $\boldsymbol{m}_d = \boldsymbol{m}_d^* + \boldsymbol{P}_{d,k}^{*}\boldsymbol{m}_k^{\text{MCD}}$ and the final loadings to $\boldsymbol{P}_{d,k} = \boldsymbol{P}_{d,k}^{*}\boldsymbol{P}_{k,k}^{\text{MCD}}$.

6. Scores, predicted values, and residuals. We now provide the final output. We compute the scores of $\mathring{\boldsymbol{X}}_{n,d}$ as $\mathring{\boldsymbol{T}}_{n,k} = (\mathring{\boldsymbol{X}}_{n,d} - \mathbf{1}_n\boldsymbol{m}_d^{\prime})\boldsymbol{P}_{d,k}$ and the predictions of $\mathring{\boldsymbol{X}}_{n,d}$ as $\hat{\mathring{\boldsymbol{X}}}_{n,d} = \mathbf{1}_n\boldsymbol{m}_d^{\prime} + \mathring{\boldsymbol{T}}_{n,k}(\boldsymbol{P}_{d,k})^{\prime}$. (The formulas for $\dot{\boldsymbol{T}}_{n,k}$ and $\hat{\dot{\boldsymbol{X}}}_{n,d}$ are analogous.) This yields the difference matrix $\mathring{\boldsymbol{X}}_{n,d} - \hat{\mathring{\boldsymbol{X}}}_{n,d}$, which we then robustly scale by column, yielding the final standardized residual matrix $\boldsymbol{R}_{n,d}$. The orthogonal distance of $\mathring{\boldsymbol{x}}_i$ to the PCA subspace is given by

$$\mathring{\text{OD}}_i = \|\mathring{\boldsymbol{x}}_i - \hat{\mathring{\boldsymbol{x}}}_i\|. \tag{4}$$

See Section 8 for the R code carrying out MacroPCA.

MacroPCA can be carried out in $O(nd(\min(n,d) + \log(n) + \log(d)))$ time (see Section A.2 of the Appendix) which is not much more than the complexity $O(nd\min(n,d))$ of classical PCA. Figure 2 shows times as a function of n and d indicating that MacroPCA is quite fast. The fraction of NAs in the data had no substantial effect on the computation time, as seen in Figure A.3 in Section A.2, Appendix.

Note that PCA loadings are highly influenced by the variables with the largest variability. For this the MacroPCA code provides the option to divide each variable by a robust scale. This does not increase the computational complexity.

3. Outlier Detection

MacroPCA provides several tools for outlier detection. We illustrate them on a dataset collected by Alfons (2016) from the website of the Top Gear car magazine. It contains data on 297 cars, with 11 continuous variables. Five of these variables (price, displacement, BHP, torque, top speed) are highly skewed, and were logarithmically transformed. The dataset contains 95 missing cells, which is only 2.9% of the $297 \times 11 = 3267$ cells. We retained two principal components ($k = 2$).

The right-hand panel of Figure 3 shows the results of MacroPCA by a modification of the cell map introduced by Rousseeuw and Van den Bossche (2018). The computations were performed on all 297 cars, but to make the map fit on a page it only shows 24 cars, including some of the more eventful cases. The color of the cells stems from the standardized residual matrix $\boldsymbol{R}_{n,d}$ obtained by MacroPCA. Cells with $|r_{ij}| \leqslant \sqrt{\chi^2_{1,0.99}} = 2.57$ are considered regular and colored yellow in the residual map, whereas the missing values are white. Outlying residuals receive a color which ranges from light orange to red when $r_{ij} > 2.57$ and from light purple to dark blue when $r_{ij} < -2.57$. So a dark red cell indicates that its observed value is much higher than its fitted value, while a dark blue cell means the opposite.

To the right of each row in the map is a circle whose color varies from white to black according to the orthogonal distance $\mathring{\text{OD}}_i$ given by Equation (4) compared to the cutoff Equation (3). Cases with $\mathring{\text{OD}}_i \leqslant c_{\text{OD}}$ lie close to the PCA subspace and receive a white circle. The others are given darker shades of gray up to black according to their $\mathring{\text{OD}}_i$.

On these data, we also ran the ICPCA method, which handles missing values in classical PCA. It differs from MacroPCA in some important ways: the initial imputations are by nonrobust column means, the iterations carry out CPCA and do not exclude outlying rows, and the residuals are standardized by the nonrobust standard deviation. By itself ICPCA does not provide a residual map, but we can construct one anyway by plotting the nonrobust standardized residuals with the same color scheme, yielding the left panel of Figure 3.

The ICPCA algorithm finds high orthogonal distances (dark circles) for the BMW i3, the Chevrolet Volt, the Renault Twizzy, and the Vauxhall Ampera. These are hybrid or purely electrical cars with a high or missing MPG (miles per gallon). Note that the Ssangyong Rodius and Renault Twizzy get blue cells for their acceleration time of zero seconds, which is physically impossible.

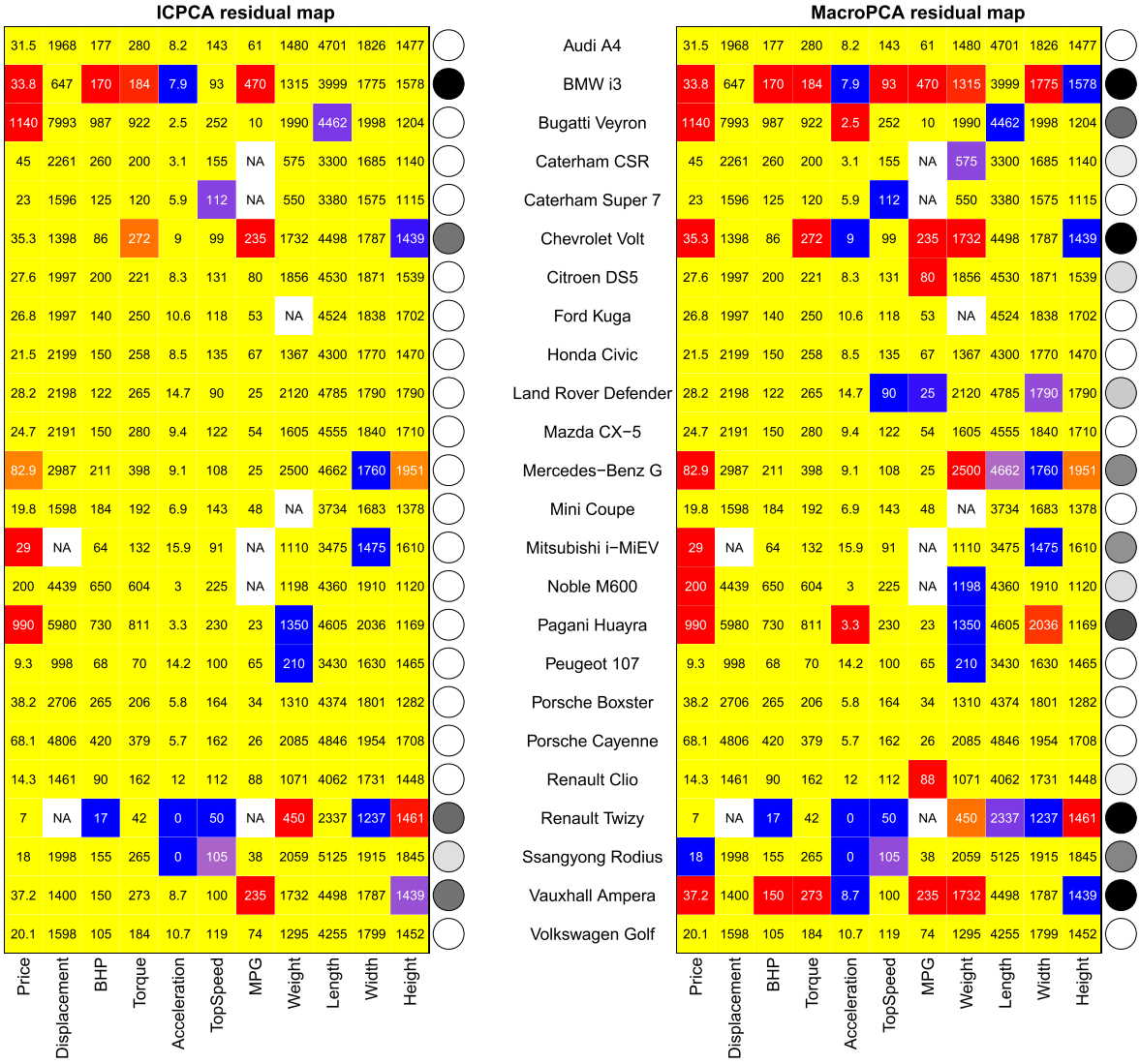

Figure 3. Residual map of selected rows from Top Gear data: (left) when using ICPCA; (right) when using MacroPCA. The numbers shown in the cells are the original data values (with price in units of 1000 UK Pounds).

On this dataset the ICPCA algorithm provides decent results because the total number of outliers is small compared to the size of the data, and indeed the residual map of all 297 cars was mostly yellow. But MacroPCA (right panel) detects more deviating behavior. The orthogonal distance of the hybrid Citroen DS5 and the electrical Mitsubishi i-MiEV are now on the high side, and the method flags the Bugatti Veyron and Pagani Huayra supercars as well as the Land Rover Defender and Mercedes-Benz G all-terrain vehicles. It also flags more cells, giving a more complete picture of the special characteristics of some cars.

We can also compute the *score distance* of each case, which is the robustified Mahalanobis distance of its projection on the PCA subspace among all such projected points. It is easily computed as

$$\mathring{\mathrm{SD}}_i = \sqrt{\sum_{j=1}^{k} (\mathring{t}_{ij})^2 / \hat{\lambda}_j}, \tag{5}$$

where $\mathring{t}_{ij}$ are the scores and $\hat{\lambda}_j$ the eigenvalues obtained by MacroPCA. This allows us to construct a PCA outlier map of cases as introduced in Hubert, Rousseeuw, and Vanden Branden (2005), which plots the orthogonal distances $\mathring{\mathrm{OD}}_i$ on the vertical axis versus the score distances $\mathring{\mathrm{SD}}_i$. The MacroPCA outlier map of these data is the right panel of Figure 4. The vertical line indicates the cutoff $c_{\mathrm{SD}} = \sqrt{\chi^2_{k,0.99}}$ and the horizontal line is the cutoff c_{OD}.

Regular cases are those with a small $\mathring{\mathrm{SD}}_i \leqslant c_{\mathrm{SD}}$ and a small $\mathring{\mathrm{OD}}_i \leqslant c_{\mathrm{OD}}$. Cases with large $\mathring{\mathrm{SD}}_i$ and small $\mathring{\mathrm{OD}}_i$ are called good leverage points. The cases with large $\mathring{\mathrm{OD}}_i$ can be divided into orthogonal outliers (when their $\mathring{\mathrm{SD}}_i$ is small) and bad leverage points (when their $\mathring{\mathrm{SD}}_i$ is large too). We see several orthogonal outliers such as the Vauxhall Ampera as well as some bad leverage points, especially the BMW i3. There are also some good leverage points.

The left panel displays the outlier map for ICPCA. It flags the BMW i3 as an orthogonal outlier. This behavior is typical

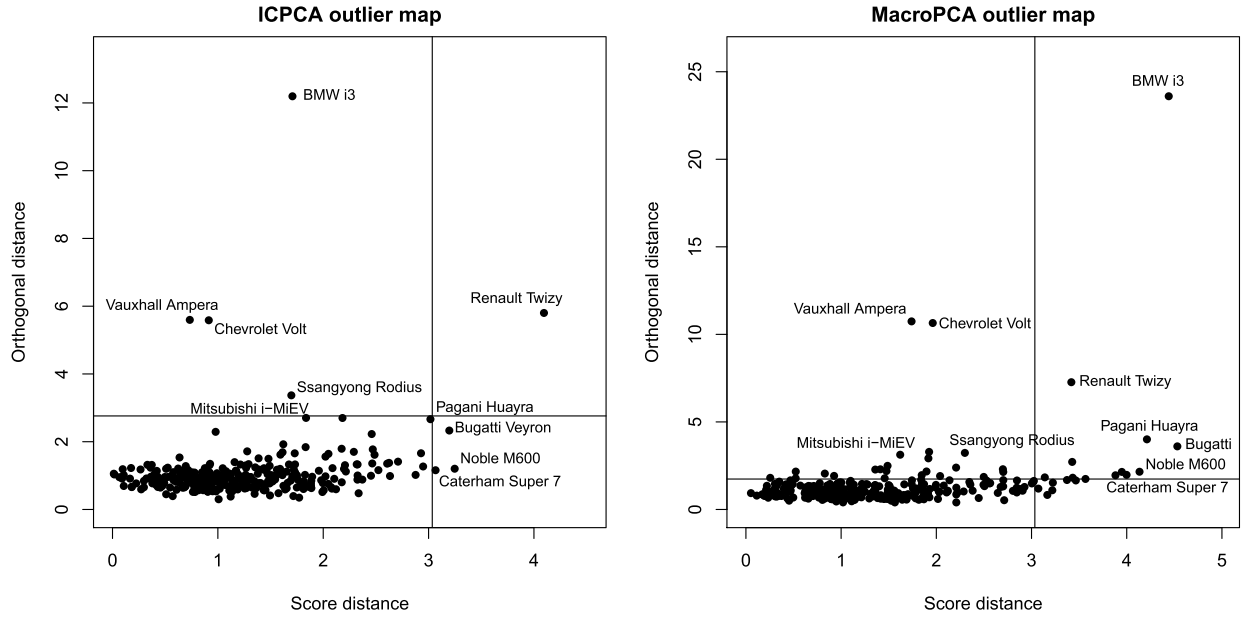

Figure 4. Outlier map of Top Gear data: (left) when using ICPCA; (right) when using MacroPCA.

because a bad leverage point will attract the fit of classical methods, making it appear less special. For the same reason ICPCA considers some of the good leverage points as regular cases. That the ICPCA outlier map is still able to flag some outliers is due to the fact that this dataset only has a small percentage of outlying rows.

4. Online Data Analysis

Applying MacroPCA to a dataset $\boldsymbol{X}_{n,d}$ yields a PCA fit. Now suppose that a new case (row) $\boldsymbol{x}$ comes in, and we would like to impute its missing values, detect its outlying cells and impute them, estimate its scores, and find out whether or not it has an outlying score and/or a high orthogonal distance. We could of course append $\boldsymbol{x}$ to $\boldsymbol{X}_{n,d}$ and rerun MacroPCA, but that would be very inefficient.

Instead we propose a method to analyze $\boldsymbol{x}$ using only the output of MacroPCA on the initial set $\boldsymbol{X}_{n,d}$. This can be done quite fast, which makes the procedure suitable for online process control. For outlier-free data with NAs this was studied by Nelson, Taylor, and MacGregor (1996) and Walczak and Massart (2001). Folch-Fortuny, Arteaga, and Ferrer (2015) call this model exploitation, as opposed to model building (fitting a PCA model). Our procedure consists of two stages, along the lines of MacroPCA.

1. DDCpredict is a new function which only uses $\boldsymbol{x}$ and the output of DDC on the initial data $\boldsymbol{X}_{n,d}$. First the entries of $\boldsymbol{x}$ are standardized using the robust location and scale estimates from DDC. Then all x_j with $|x_j| > \sqrt{\chi^2_{1,0.99}} = 2.57$ are replaced by NAs. Next, all NAs are estimated as in DDC making use of the pre-built coefficients b_{jh} and weights w_{jh}. Also the deshrinkage step uses the original robust slopes. The *DDCpredict* stage yields the imputed vector $\tilde{\boldsymbol{x}}^{(0)}$ and the standardized residual of each cell x_j.
2. MacroPCApredict improves on the initial imputation $\tilde{\boldsymbol{x}}^{(0)}$. The improvements are based solely on the $\boldsymbol{m}_d$ and $\boldsymbol{P}_{d,k}$ that were obtained by MacroPCA on the original data $\boldsymbol{X}_{n,d}$. Step $s \geqslant 1$ is of the following form:
   (a) Project the imputed case $\tilde{\boldsymbol{x}}^{(s-1)}$ on the MacroPCA subspace to obtain its scores vector $\boldsymbol{t}^{(s)} = (\boldsymbol{P}_{d,k})'(\tilde{\boldsymbol{x}}^{(s-1)} - \boldsymbol{m}_d)$;
   (b) transform the scores to the original space, yielding $\hat{\boldsymbol{x}}^{(s)} = \boldsymbol{m}_d + \boldsymbol{P}_{d,k}\boldsymbol{t}^{(s)}$;
   (c) reimpute the outlying cells and missing values of $\boldsymbol{x}$ by the corresponding values of $\hat{\boldsymbol{x}}^{(s)}$, yielding $\tilde{\boldsymbol{x}}^{(s)}$.

   These steps are iterated until convergence (when the new imputed values are within a tolerance of the old ones) or the maximal number of steps (by default 20) is reached. We denote the final $\tilde{\boldsymbol{x}}^{(s)}$ as $\tilde{\boldsymbol{x}}$.

   Next, we create $\mathring{\boldsymbol{x}}$ by replacing the missing values in $\boldsymbol{x}$ by the corresponding cells in $\tilde{\boldsymbol{x}}$. We then compute the orthogonal distance $\mathrm{OD}(\mathring{\boldsymbol{x}})$ and the score distance $\mathrm{SD}(\mathring{\boldsymbol{x}})$, and classify it as a regular case, a leverage point or an orthogonal outlier. Finally the cell residuals $\mathring{x}_j - \hat{\mathring{x}}_j$ are standardized as in the last step of MacroPCA, and used to flag outlying cells in $\boldsymbol{x}$.

To illustrate this prediction procedure we reanalyze the Top Gear dataset. We exclude the 24 cars shown in the residual map of Figure 3 and build the MacroPCA model on the remaining data. This model was then provided to analyze the 24 selected cars as "new" data. Figure 5 shows the result. As before the cells are colored according to their standardized residual, and the circles on the right are filled according to their $\mathring{\mathrm{OD}}$. The left panel is the MacroPCA residual map shown in Figure 3, which was obtained by applying MacroPCA to the entire dataset. The right panel shows the result of analyzing these 24 cases using the fit obtained without them. The residual maps are quite similar. Note that each cell now shows its standardized residual (instead

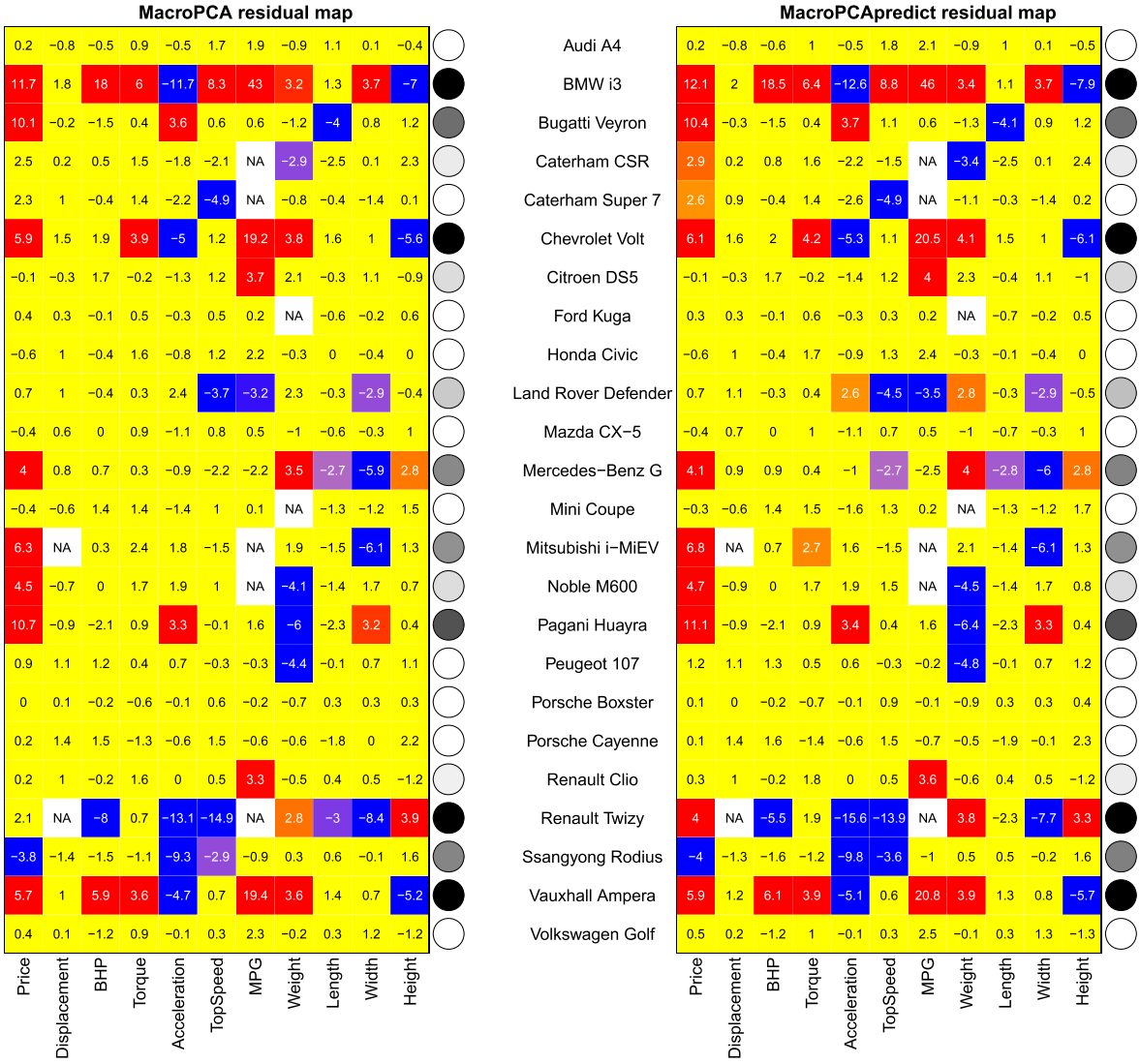

Figure 5. Top Gear dataset: residual maps obtained by (left) including and (right) excluding these 24 cars when fitting the PCA model.

of its data value as in Figure 3), making it easier to see the differences.

5. Simulations

We have compared the performance of ICPCA, MROBPCA, and MacroPCA in an extensive simulation study. Several contamination models were considered with missing values, cellwise outliers, rowwise outliers, and combinations of them. Only a few of the results are reported here since the others yielded similar conclusions.

The clean data $\boldsymbol{X}^0_{n,d}$ are generated from a multivariate Gaussian with $\boldsymbol{\mu} = \boldsymbol{0}$ and two types of covariance matrices $\boldsymbol{\Sigma}_{d,d}$. The first one is based on the structured correlation matrix called A09 where each off-diagonal entry is $\rho_{i,j} = (-0.9)^{|i-j|}$. The second type of covariance matrix is based on the random correlation matrices of Agostinelli et al. (2015) and will be called ALYZ. These correlation matrices are turned into covariance matrices with other eigenvalues. More specifically, the diagonal elements of the matrix $\boldsymbol{L}_{d,d}$ from the spectral decomposition $\boldsymbol{\Sigma}_{d,d} = \boldsymbol{P}_{d,d}\boldsymbol{L}_{d,d}\boldsymbol{P}'_{d,d}$ are replaced by the desired values listed below. The specifications of the clean data are $n = 100$, $d = 200$, $\boldsymbol{L}_{d,d} = \text{diag}(30, 25, 20, 15, 10, 5, 0.098, 0.0975, \ldots, 0.0020, 0.0015)$, and $k = 6$ (since $\sum_{j=1}^{6} \lambda_j / \sum_{j=1}^{200} \lambda_j = 91.5\%$). MacroPCA takes less than a second for $n = 100$, $d = 200$ as seen in Figure 2.

In a first simulation setting, the clean data $\boldsymbol{X}^0_{n,d}$ are modified by replacing a random subset of 5%, 10%, ... up to 30% of the $n \times d$ cells with NAs. The second simulation setting generates NAs and outlying cells by randomly replacing 20% of the cells x_{ij} by missing values and 20% by the value $\gamma\sigma_j$ where σ_j^2 is the jth diagonal element of $\boldsymbol{\Sigma}_{d,d}$ and γ ranges from 0 to 20. The third simulation setting generates NAs and outlying rows. Here 20% of random cells are replaced by NAs and a random subset of 20% of the rows is replaced by rows generated from $N(\gamma\boldsymbol{v}_{k+1}, \boldsymbol{\Sigma}_{d,d})$, where γ varies from 0 to 50 and $\boldsymbol{v}_{k+1}$ is the $(k+1)$th eigenvector of $\boldsymbol{\Sigma}_{d,d}$. The last simulation setting generates 20% of NAs, together with 10% of cellwise outliers and 10% of rowwise outliers in the same way.

In each setting, we consider the set C consisting of the rows i that were not replaced by rowwise outliers, with $c := \#C$, and

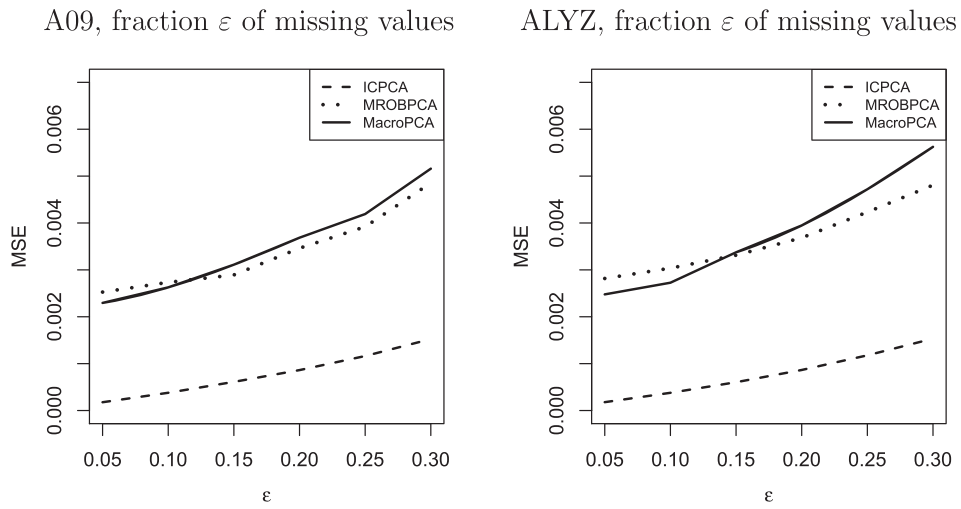

Figure 6. Average MSE as a function of the fraction ε of missing values. The data were generated using A09 (left) and ALYZ (right).

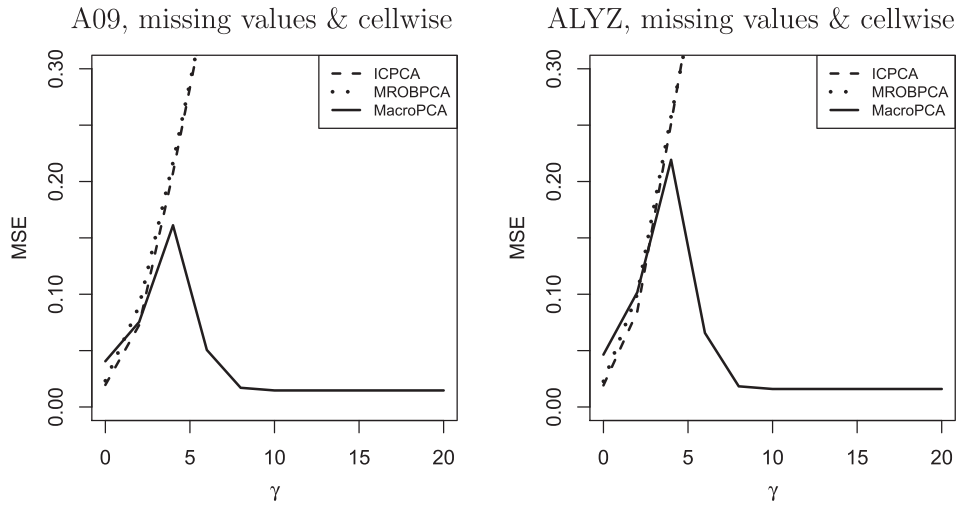

Figure 7. Average MSE for data with 20% of missing values and 20% of cellwise outliers, as a function of γ which determines the distance of the cellwise outliers.

the data matrix $\boldsymbol{X}^0_{c,d}$ consisting of those rows of the clean data $\boldsymbol{X}^0_{n,d}$. As a baseline for the simulation we apply classical PCA to $\boldsymbol{X}^0_{c,d}$ and denote the resulting predictions by $\hat{x}^C_{ij}$ for i in C. We then measure the mean squared error (MSE) from the baseline:

$$\text{MSE} = \frac{1}{cd} \sum_{i \in C} \sum_{j=1}^{d} \left(\hat{x}_{ij} - \hat{x}^C_{ij} \right)^2 ,$$

where $\hat{x}_{ij}$ is the predicted value for x_{ij} obtained by applying the different methods to the contaminated data. The MSE is then averaged over 100 replications.

Figure 6 shows the performance of ICPCA, MROBPCA, and MacroPCA when some data become missing. As CPCA and ROBPCA cannot deal with NAs, they are not included in this comparison. Since there are no outliers the classical ICPCA performs best, followed by MROBPCA and MacroPCA which perform similarly to each other, and only slightly worse than ICPCA considering the scale of the vertical axis which is much smaller than in the other three simulation settings.

Now we set 20% of the data cells to missing and add 20% of cellwise contamination given by γ. Figure 7 shows the performance of ICPCA, MROBPCA, and MacroPCA in this situation. The MSE of both ICPCA and MROBPCA grows very fast with γ which indicates that these methods are not at all robust to cellwise outliers. Note that $d = 200$ so on average $1 - (1 - 0.2)^{200} \approx 100\%$ of the rows are contaminated, whereas no purely rowwise method can handle more than 50%. MacroPCA is the only method that can withstand cellwise outliers here. When γ is smaller than five the MSE goes up, but this is not surprising as in that case the values in the contaminated cells are still close to the clean ones. As soon as the contamination is sufficiently far away, the MSE drops to a very low value.

Figure 8 presents the results of ICPCA, MROBPCA, and MacroPCA when there are 20% of missing values combined with 20% of rowwise contamination. As expected, the ICPCA algorithm breaks down while MROBPCA and MacroPCA provide very good results. MROBPCA and MacroPCA are affected the most (but not much) by nearby outliers, and very little by far contamination.

Finally, Figure 9 presents the results in the situation of 20% of missing values combined with 10% of cellwise and 10% of rowwise contamination. In this scenario the ICPCA and

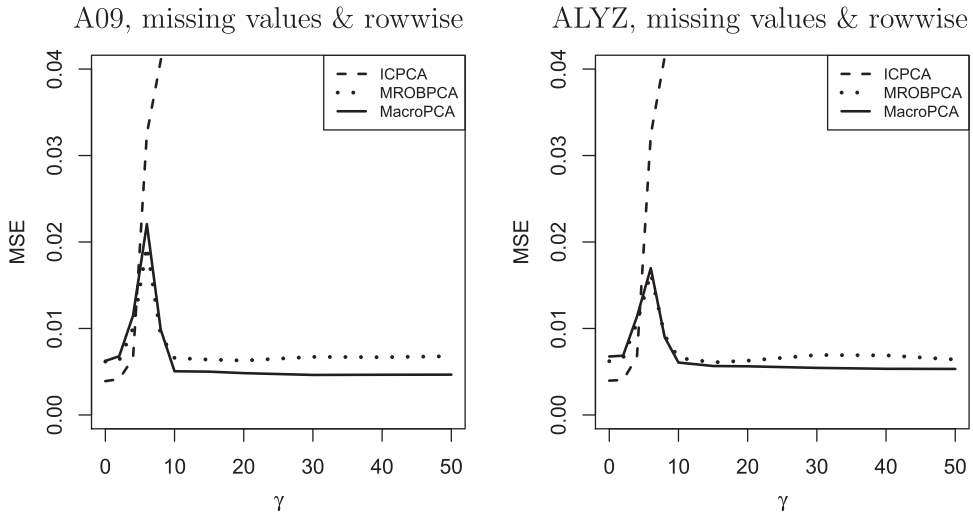

Figure 8. Average MSE for data with 20% of missing values and 20% of rowwise outliers, as a function of γ, which determines the distance of the rowwise outliers.

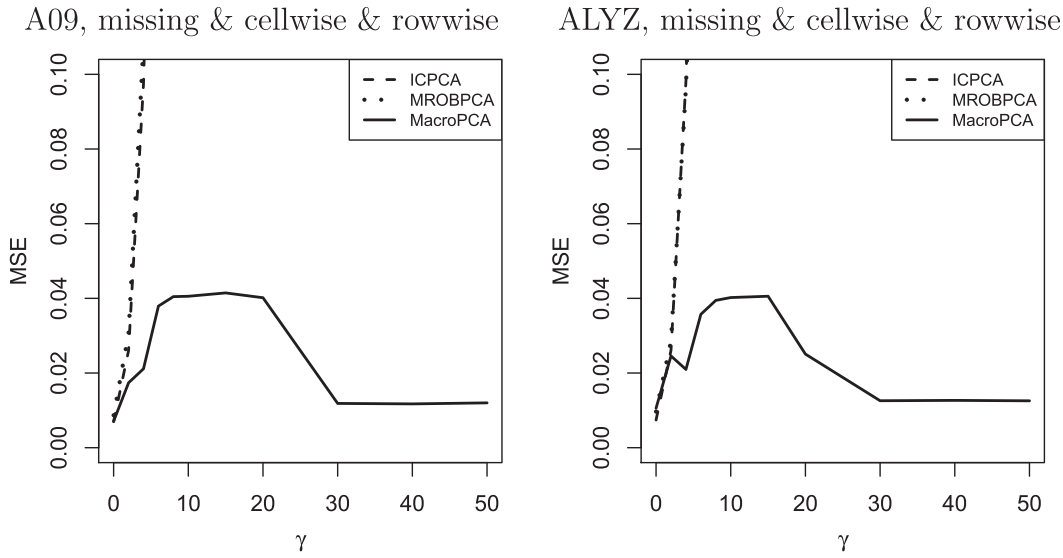

Figure 9. Average MSE for data with 20% of missing values, 10% of cellwise outliers and 10% of rowwise outliers, as a function of γ which determines the distance of both the cellwise and the rowwise outliers.

MROBPCA algorithms break down whereas MacroPCA still provides reasonable results.

In this section, the missing values were generated in a rather simple way. In Section A.3 they are generated in a more challenging way but still MAR, with qualitatively similar results.

6. Real Data Examples

6.1. Glass Data

The *glass* dataset (Lemberge et al. 2000) contains spectra with $d = 750$ wavelengths of $n = 180$ archeological glass samples. It is available in the R package *cellWise* (Raymaekers et al. 2019). The MacroPCA method selects four principal components and yields a 180×750 matrix of standardized residuals. There is not enough resolution on a page to show so many individual cells in a residual map. Therefore, we created a map (the top panel of Figure 10) which combines the residuals into blocks of 5×5 cells. The color of each block now depends on the most frequent type of outlying cell in it, the resulting color being an average. For example, an orange block indicates that quite a few cells in the block were red and most of the others were yellow. The more red cells in the block, the darker red the block will be. We see that MacroPCA has flagged a lot of cells, that happen to be concentrated in a minority of the rows where they show patterns. In fact, the colors indicate that some of the glass samples (between 22 and 30) have a higher concentration of phosphor, whereas rows 57–63 and 74–76 had an unusually high concentration of calcium. The bottom part of the residual map looks very different, due to the fact that the measuring instrument was cleaned before recording the last 38 spectra. One could say that those outlying rows belong to a different population.

Since the dataset has no NAs and we found that fewer than half of the rows are outlying, it can also be analyzed by the original ROBPCA method as was done by Hubert, Rousseeuw, and Vanden Branden (2005), also for $k = 4$. This detects the same rowwise outliers. In principle ROBPCA is a purely rowwise

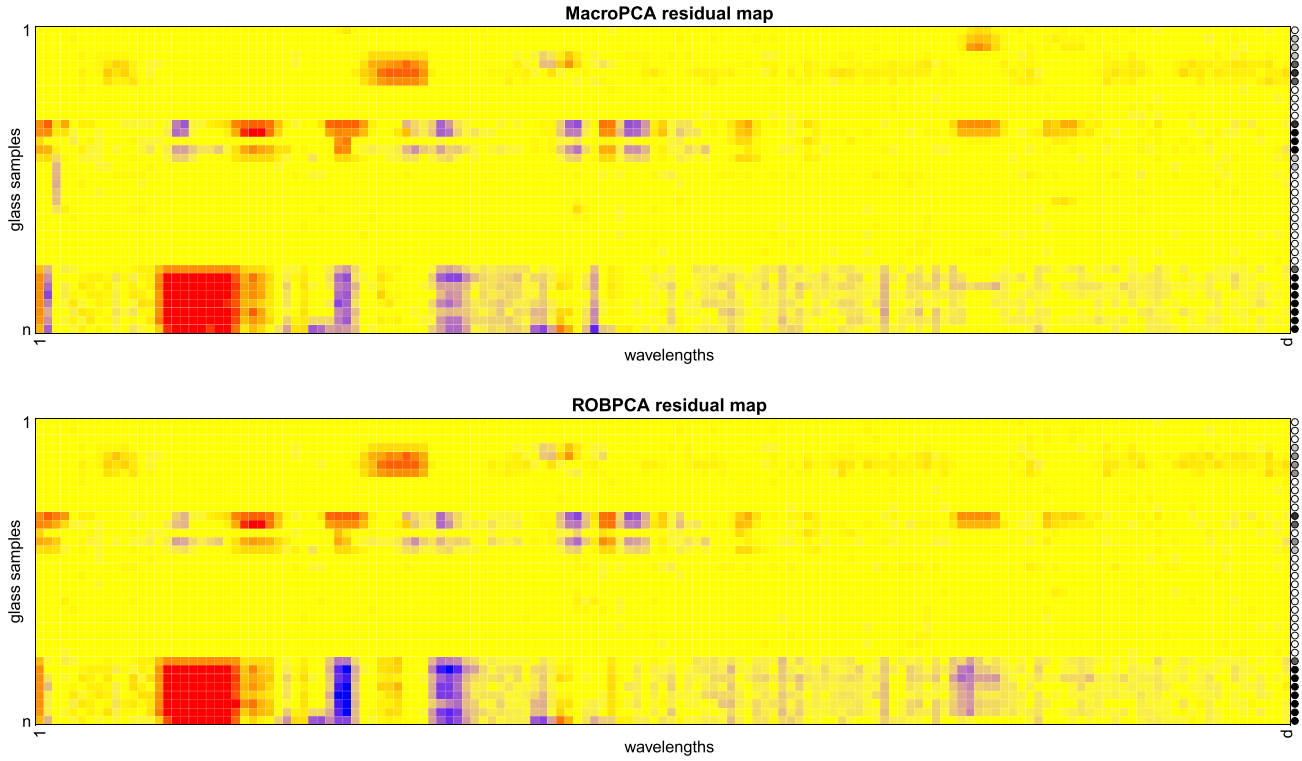

Figure 10. Residual maps of the glass dataset when fitting the PCA model by MacroPCA (top) and ROBPCA (bottom).

method that does not flag cells. Even though ROBPCA does not produce a residual map, we can construct one analogously to that of MacroPCA. First we construct the residual matrix of ROBPCA, the rows of which are given by $\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i$, where $\hat{\boldsymbol{x}}_i$ is the projection of $\boldsymbol{x}$ on the ROBPCA subspace. We then standardize the residuals in each column by dividing them by a robust one-step scale M-estimate. This yields the bottom panel of Figure 10. We see that the two residual maps look quite similar.

This example illustrates that purely rowwise robust methods can be useful to detect cellwise outliers when these cells occur in fewer than 50% of the rows. But if the cellwise outliers contaminate more rows, this approach is insufficient.

6.2. DPOSS Data

In our last example, we analyze data from the Digitized Palomar Sky Survey (DPOSS) described by Odewahn et al. (1998). This is a huge database of celestial objects, from which we have drawn 20,000 stars at random. Each star has been observed in the color bands J, F, and N. Each band has seven variables. Three of them measure light intensity: for the J band they are MAperJ, MTotJ, and MCoreJ, where the last letter indicates the band. The variable AreaJ is the size of the star based on its number of pixels. The remaining variables IR2J, csfJ, and EllipJ combine size and shape. (There were two more variables in the original data, but these measured the background rather than the star itself.) There are substantial correlations between these 21 variables.

In this dataset 84.6% of the rows contain NAs (in all there are 50.2% missing entries). Often an entire color band is missing, and sometimes two. We applied MacroPCA to these data, choosing $k = 4$ components according to the screeplot. The left panel of Figure 11 shows the loadings of the first and second component. It appears that the first component captures the overall negative correlation between two groups of variables: those measuring light intensity (the first three variables in each band) and the others (variables 4–7 in each band). The right panel is the corresponding scores plot, in which the 150 stars with the highest orthogonal distance OD are shown in red. Most of these stand out in the space of PC1 and PC2 (bad leverage points), whereas some only have a high OD (orthogonal outliers).

Figure 12 shows the residual map of MacroPCA, in which each row block combines 25 stars. The six rows at the top correspond to the 150 stars with highest OD. We note that the outliers tend to be more luminous (MTot) than expected and have a larger Area, which suggests giant stars. The analogous residual map of ICPCA (not shown) did not reveal much. Note that the non-outlying rows in the bottom part of the residual map are yellow, and the missing color bands show up as blocks in lighter yellow (a combination of yellow and white cells).

7. Conclusions

The MacroPCA method is able to handle missing values, cellwise outliers, and rowwise outliers. This makes it well-suited for the analysis of possibly messy real data. Simulation showed that its performance is similar to a classical method in the case of outlier-free data with missing values, and to an existing robust method when the data only has rowwise outliers. The algorithm is fast enough to deal with many variables, and has recently been sped up by recoding parts in C++, making it much faster than the pure R code in Figure 2 and Figure A.3.

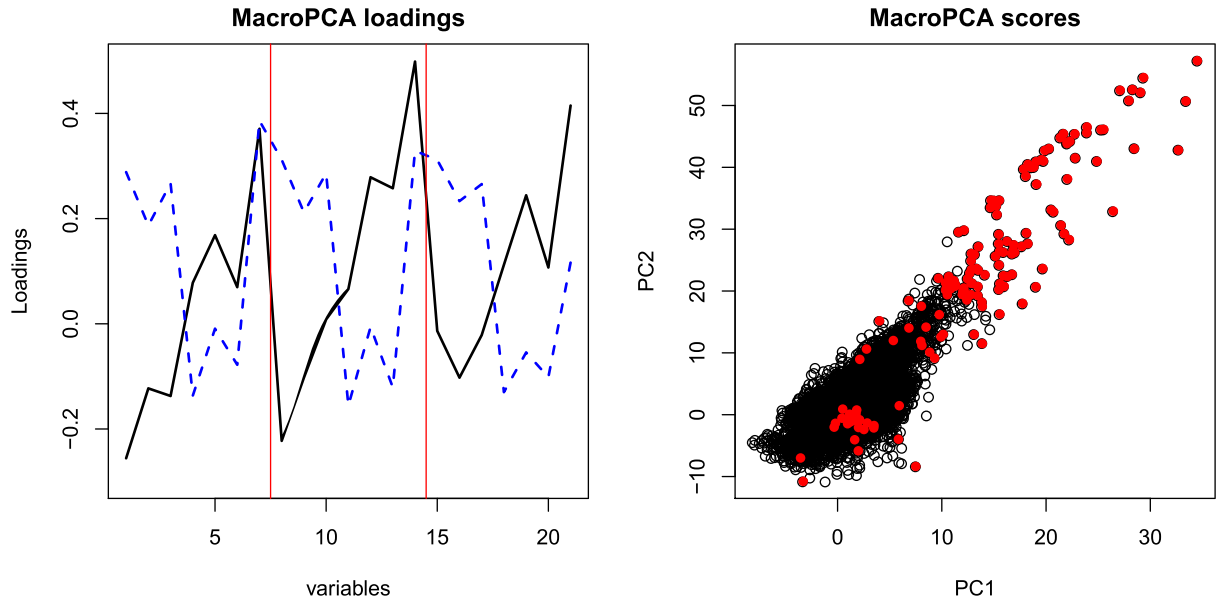

Figure 11. DPOSS stars data: (left) loadings of the first (black full line) and the second (blue dashed line) component of MacroPCA, with vertical lines separating the three color bands; (right) plot of the first two scores, with filled red circles for stars with high orthogonal distance $\overset{\circ}{\text{OD}}$ and open black circles for the others.

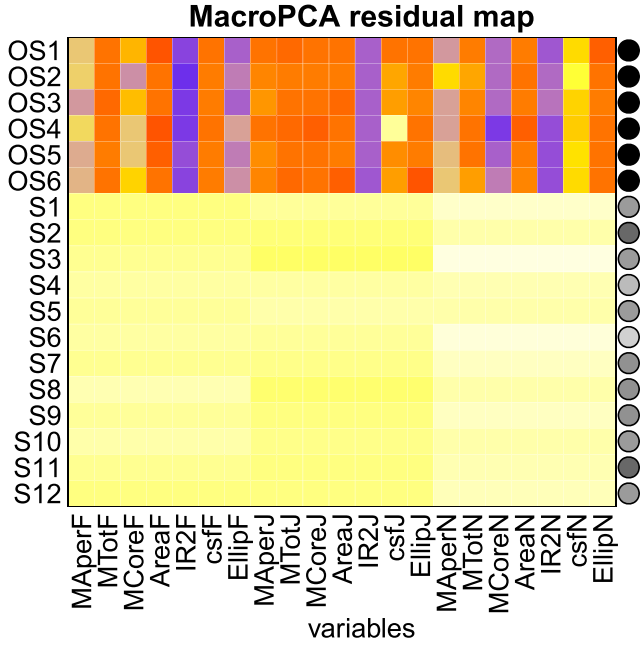

Figure 12. MacroPCA residual map of stars in the DPOSS data, with 25 stars per row block. The six row blocks at the top correspond to the stars with highest $\overset{\circ}{\text{OD}}$.

MacroPCA can analyze new data as they come in, only making use of its existing output obtained from the initial dataset. It imputes missing values in the new data, flags and imputes outlying cells, and flags outlying rows. This computation is fast, so it can be used to screen new data in quality control or even online process control. (One can update the initial fit offline from time to time.) The advantage of MacroPCA is that it not only tells us when the process goes out of control, but also which variables are responsible.

Potential extensions of MacroPCA include methods of PCA regression and partial least squares able to deal with rowwise and cellwise outliers and missing values.

8. Software Availability

The methods described in this article as well as the DPOSS data have been made available in the R package *cellWise* (Raymaekers et al. 2019) on CRAN, with a vignette reproducing the examples shown.

Supplementary Materials

This is a text with more details about the first part of the MacroPCA algorithm and a list of notations.

Acknowledgments

The reviewers provided helpful suggestions to improve the presentation. Jakob Raymaekers optimized and translated parts of our R-code to C++ for CRAN.

Funding

The research of P. Rousseeuw and M. Hubert has been supported by projects of Internal Funds KU Leuven. W. Van den Bossche obtained financial support from the EU Horizon 2020 project SCISSOR: Security in trusted SCADA and smart-grids 2015–2018.

Appendix

A.1. Example Illustrating Step 5 of MacroPCA

Steps 1–4 of the MacroPCA algorithm construct a robust k-dimensional subspace. The purpose of Step 5 is to robustly estimate basis vectors (loadings) of the subspace. The columns of the loadings matrix $\boldsymbol{P}^*_{d,k}$ from Step 4 need not be robust, because some of the n^* rows in H^* might be outlying inside the subspace. Such points are called good leverage points, where "good" refers to the fact that they do not harm the estimation of the subspace, but on the other hand they can affect the estimated eigenvectors and eigenvalues. We illustrate this by a small toy example.

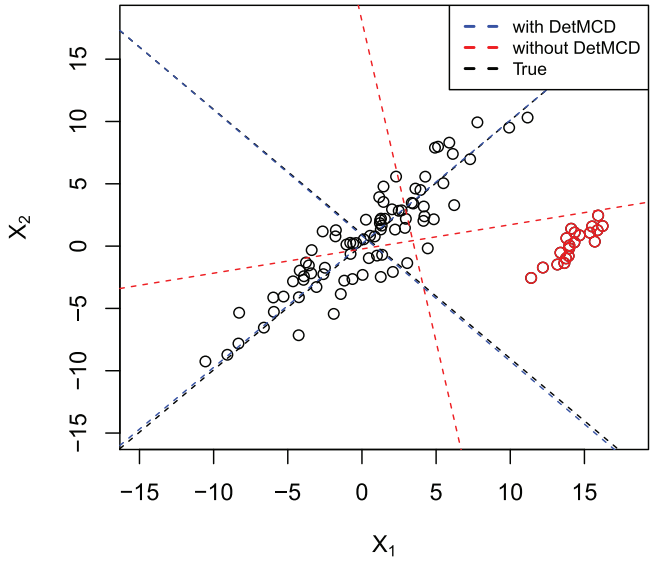

Figure A.1. Scatterplot of the contaminated dataset projected on the subspace with $k = 2$. The black points are from the clean data, and the red ones are good leverage points. The dashed lines represent the eigenvectors.

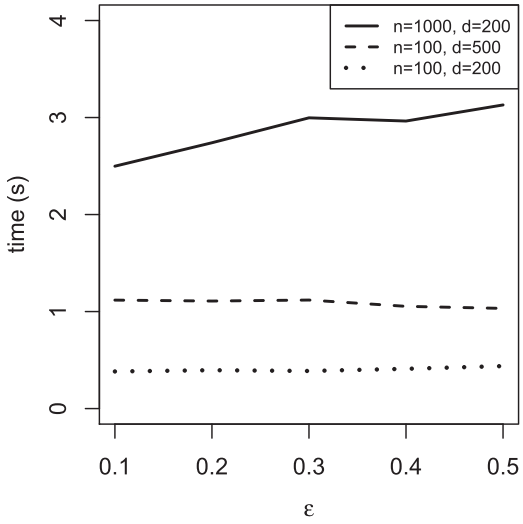

Figure A.3. Computation times of MacroPCA in seconds, as a function of the fraction ε of NAs in the data.

A clean dataset with $n = 100$ and $d = 3$ was generated according to a trivariate Gaussian distribution, with covariance matrix chosen in such a way that $k = 2$ components explain 90% of the total variance. Next, the data was contaminated by replacing 20% of the points by good leverage points. This means that these points are only outlying inside the two-dimensional subspace, and not in the direction of its orthogonal complement.

Figure A.1 shows the scatterplot of the data points projected on this two-dimensional subspace. The black dashed lines represent the eigenvectors of the clean data, and the blue and red dashed lines are the estimated eigenvectors of MacroPCA on the contaminated data, with and without Step 5. The center and directions of the blue lines were obtained by DetMCD in Step 5, whereas the red lines correspond to the center $\boldsymbol{m}_d^*$ and the loading matrix $\boldsymbol{P}_{d,k}^*$ obtained in Step 4. The blue lines (with Step 5) are almost on top of the black ones, whereas the red lines (without Step 5) are very different. Although the blue and the red fits span the same subspace, we see that without Step 5 the estimated loading vectors are far off.

This can also be seen in Figure A.2 which shows the scores on the first two components of MacroPCA with and without Step 5. In the left panel, we see that the scores reflect the shape of the clean data, whereas those in the right panel were affected by the leverage points. This illustrates that the interpretation of the results can be distorted by good leverage points if Step 5 is not taken.

A.2. Computational Complexity of MacroPCA

The computational complexity of MacroPCA depends on how its steps are implemented. The DDC in the first part has an $O(nd\log(d))$ implementation (Raymaekers et al. 2019). The outlyingness Equation (2) in

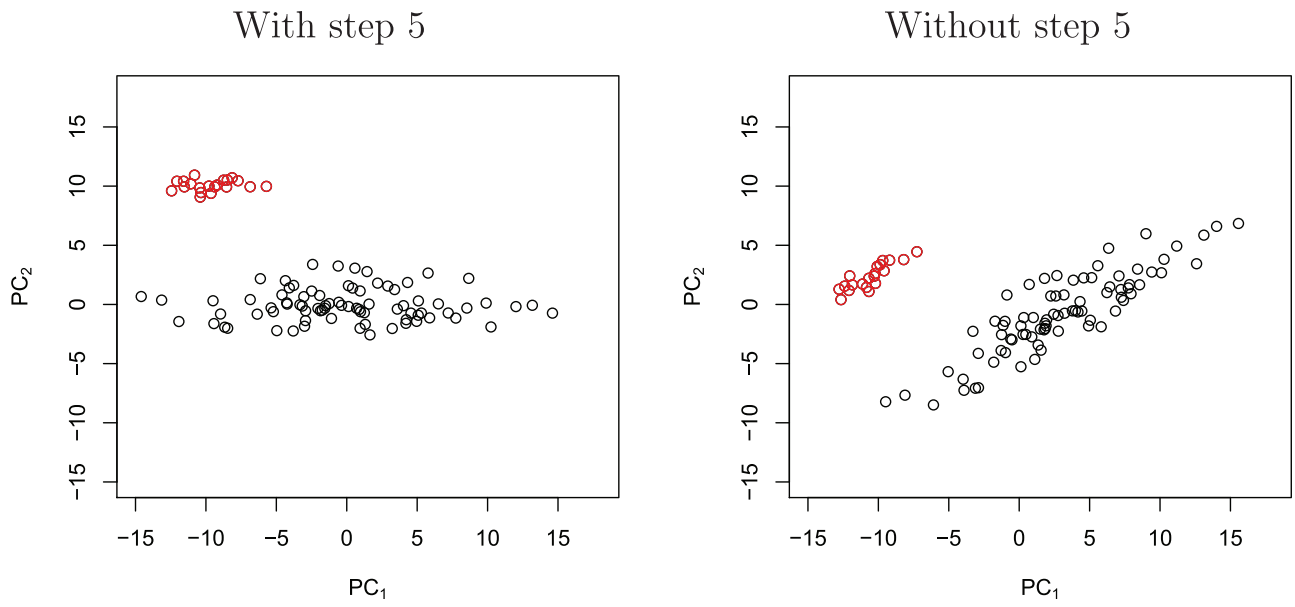

Figure A.2. Scores plots with Step 5 (left) and without it (right).

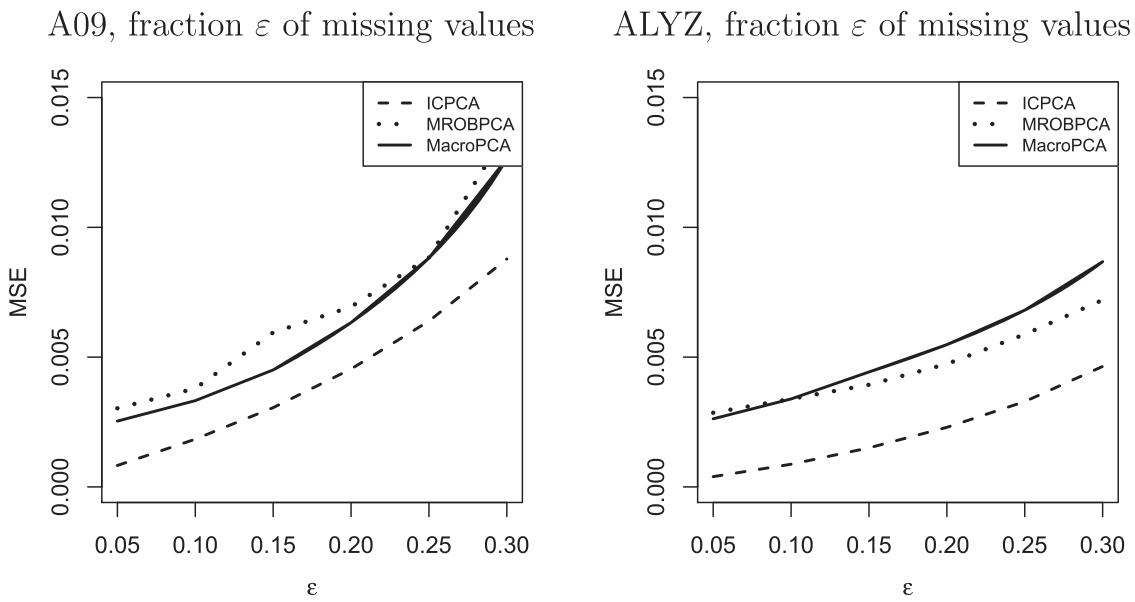

Figure A.4. Average MSE for data with MAR missingness as a function of the fraction ε of missing cells.

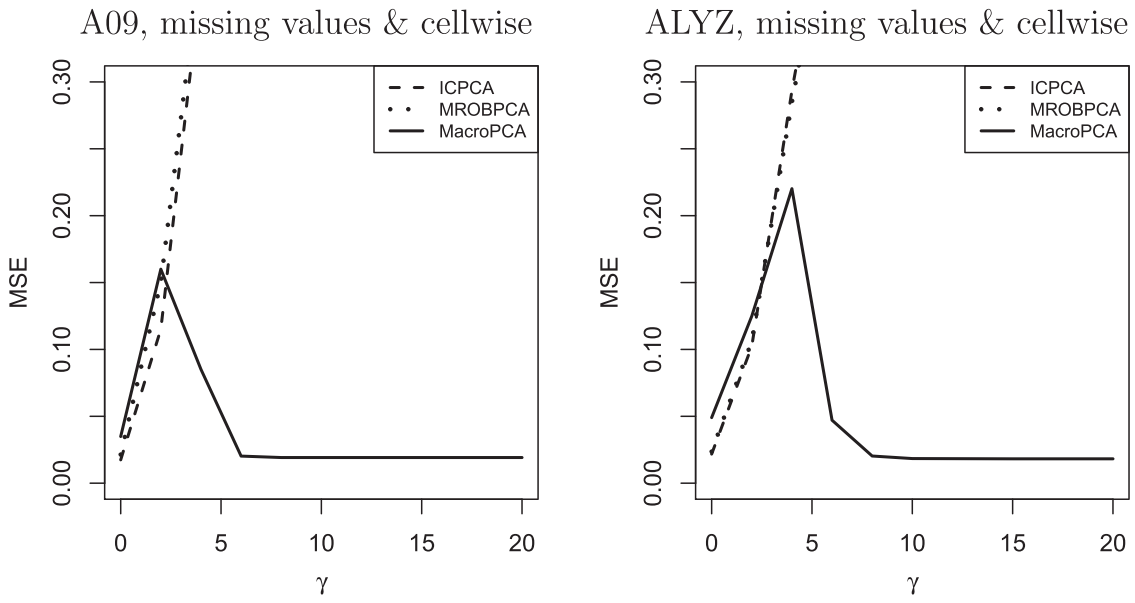

Figure A.5. Average MSE for data with 20% of MAR NAs and 20% of cellwise outliers, as a function of γ, which determines the distance of the cellwise outliers.

Step 1 requires $O(nd + n\log(n))$ time. The classical PCA in Steps 2–4 needs $O(nd\min(n,d))$ if it is carried out by singular value decomposition (SVD) instead of the eigendecomposition of a covariance matrix. The scores and predictions in Steps 3–6 require $O(ndk)$ but since we assume that the number of components k is at most 10 this becomes $O(nd)$. The DetMCD in Step 5 combines initial estimators of total complexity $O(n\log(n)k^2)$ with C-steps that require $O(nk^2)$ if performed by SVD. The robust scales of the residuals in Step 6 take $O(n\log(n)d)$ time. The overall complexity of MacroPCA thus, becomes $O(nd(\min(n,d)+\log(n)+\log(d)))$ which is not much higher than the $O(nd\min(n,d))$ of classical PCA.

The computation times for a range of values of n and d are shown in Figure 2 in the article. The fraction of NAs in the data had no substantial effect on the computation time, as seen from Figure A.3.

A.3. Generating NAs by a Different MAR Mechanism

In this article, we have assumed that the missingness mechanism is MAR (missing at random), a typical assumption underlying EM-based methods such as ICPCA and MROBPCA that are incorporated in MacroPCA. MAR says that the missingness of a cell is unrelated to the value the cell would have had, but may be related to the values of other cells in the same row; see, for example, Schafer and Graham (2002).

In the simulations of Section 5 a random subset of ε 100% of the $n \times d$ cells was replaced by NAs. This is a simple special case of MAR, in fact it is MCAR (missing completely at random) which assumes that the missingness of a cell does not depend on its own value or that of any other cell in its row.

Here, we will look at a more challenging mechanism which is still MAR but no longer MCAR. Many MAR mechanisms are possible. We

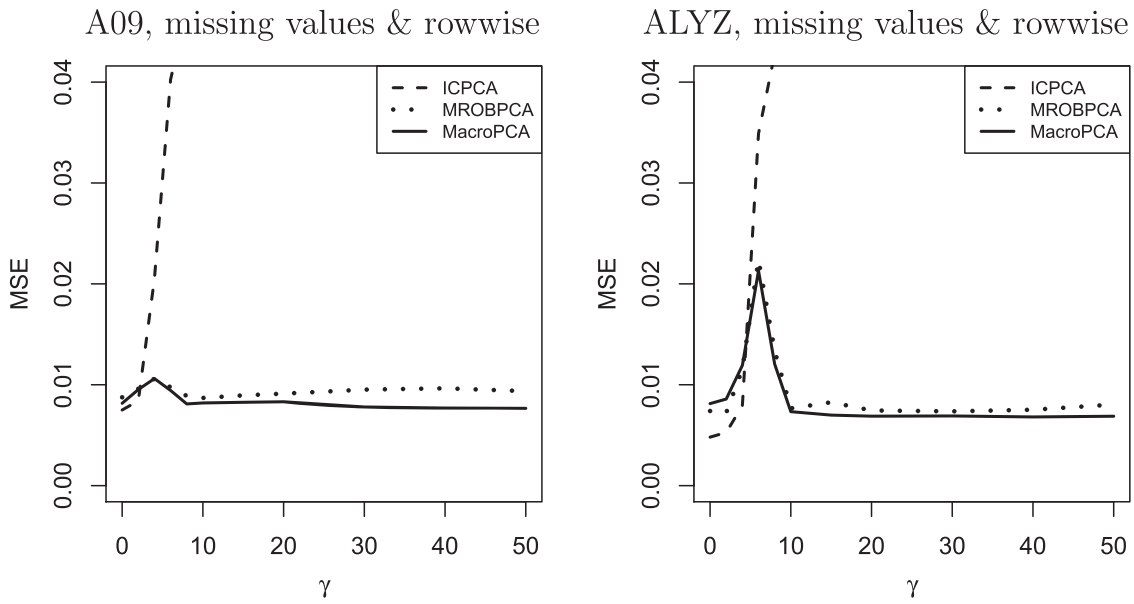

Figure A.6. Average MSE for data with 20% of MAR NAs and 20% of rowwise outliers, as a function of γ, which determines the distance of the rowwise outliers.

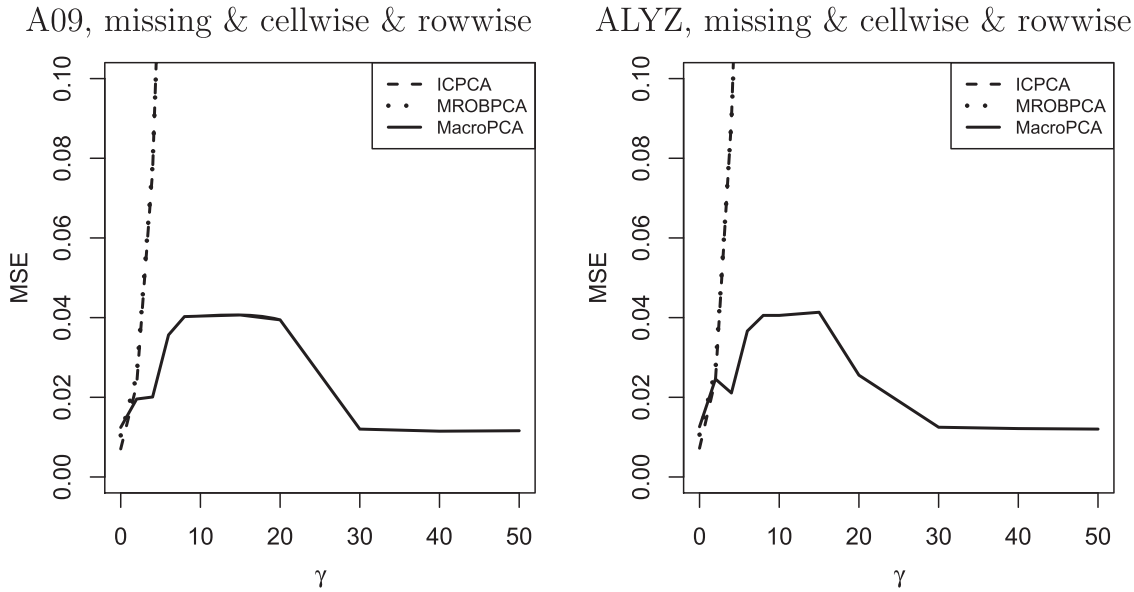

Figure A.7. Average MSE for data with 20% of MAR missing values, 10% of cellwise outliers, and 10% of rowwise outliers, as a function of γ, which determines the distance of both the cellwise and the rowwise outliers.

chose the following one where the positions of the NAs clearly depend on the values of the other cells. Given the uncontaminated matrix $\boldsymbol{X}$ we first construct the matrix $\boldsymbol{U}$ with cell values

$$\begin{aligned} u_{ij} &= |x_{id}| + |x_{i(j+1)}| \qquad \text{if } j = 1, \\ &= |x_{i(j-1)}| + |x_{i(j+1)}| \qquad \text{if } 1 < j < d, \\ &= |x_{i(j-1)}| + |x_{i1}| \qquad \text{if } j = d. \end{aligned}$$

Next, the cells of $\boldsymbol{X}$ corresponding to the ε 100% highest values of $\boldsymbol{U}$ are set to missing. Other than this, the simulation setup is as described in Section 5.

The simulation results of ICPCA, MROBPCA, and MacroPCA are shown in Figure A.4. Compared to the results with MCAR missing values in Figure 6 we see that the shape of the plots is very similar, only the scale of the MSE values is increased but it remains very low.

Next, the other three settings in Section 5 were repeated with MAR missing values. This yielded Figures A.5–A.7, which are extremely similar to the corresponding Figures 7–9. This confirms the suitability of MacroPCA in the MAR setting.

A.4. Simulation With Data Resembling DPOSS

The DPOSS data analyzed in Section 6.2 has many NAs, in fact 50% of the cells are missing. The simulation study of Section 5 did not include such an extreme situation. To check whether MacroPCA can handle data with these characteristics, we redid the simulation leading to Figure 9 for $d = 21$ variables and 50% of missing values as well as

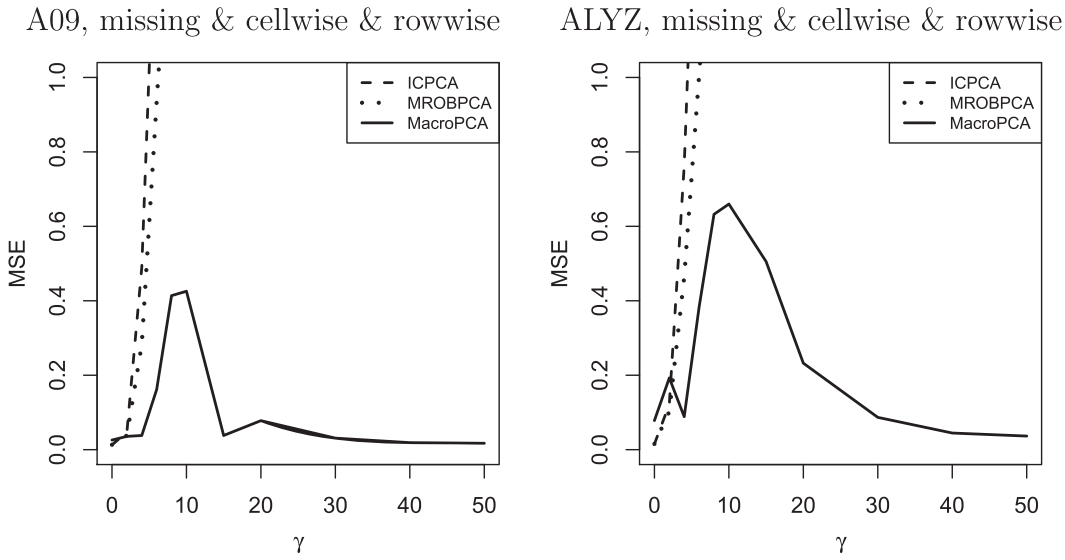

Figure A.8. Average MSE for data with 21 variables, 50% of missing values, 5% of cellwise outliers, and 5% of rowwise outliers, as a function of γ, which determines the distance of both the cellwise and the rowwise outliers.

5% of cellwise outliers and 5% of rowwise outliers, all as in the DPOSS data.

Figure A.8 shows the results, which indicate that ICPCA and MROBPCA broke down whereas MacroPCA still worked well.

References

Agostinelli, C., Leung, A., Yohai, V. J., and Zamar, R. H. (2015), "Robust Estimation of Multivariate Location and Scatter in the Presence of Cellwise and Casewise Contamination," *Test*, 24, 441–461. [459,465]

Alfons, A. (2016), *robustHD: Robust Methods for High-Dimensional Data*, CRAN. R package version 0.5.1. [462]

Alqallaf, F., Van Aelst, S., Yohai, V. J., and Zamar, R. H. (2009), "Propagation of Outliers in Multivariate Data," *The Annals of Statistics*, 37, 311–331. [459]

Cheng, T.-C., and Victoria-Feser, M.-P. (2002), "High Breakdown Estimation of Multivariate Location and Scale With Missing Observations," *British Journal of Mathematical and Statistical Psychology*, 55, 317–335. [459]

Croux, C., and Ruiz-Gazen, A. (2005), "High Breakdown Estimators for Principal Components: The Projection-Pursuit Approach Revisited," *Journal of Multivariate Analysis*, 95, 206–226. [459]

Danilov, M., Yohai, V. J., and Zamar, R. H. (2012), "Robust Estimation of Multivariate Location and Scatter in the Presence of Missing Data," *Journal of the American Statistical Association*, 107, 1178–1186. [459]

Engelen, S., Hubert, M., and Vanden Branden, K. (2005), "A Comparison of Three Procedures for Robust PCA in High Dimensions," *Austrian Journal of Statistics*, 34, 117–126. [461]

Folch-Fortuny, A., Arteaga, F., and Ferrer, A. (2015), "PCA Model Building With Missing Data: New Proposals and a Comparative Study," *Chemometrics and Intelligent Laboratory Systems*, 146, 77–88. [464]

Hubert, M., Rousseeuw, P. J., and Vanden Branden, K. (2005), "ROBPCA: A New Approach to Robust Principal Component Analysis," *Technometrics*, 47, 64–79. [459,461,463,467]

Hubert, M., Rousseeuw, P. J., and Verboven, S. (2002), "A Fast Robust Method for Principal Components With Applications to Chemometrics," *Chemometrics and Intelligent Laboratory Systems*, 60, 101–111. [459]

Hubert, M., Rousseeuw, P. J., and Verdonck, T. (2012), "A Deterministic Algorithm for Robust Location and Scatter," *Journal of Computational and Graphical Statistics*, 21, 618–637. [462]

Kiers, H. (1997), "Weighted Least Squares Fitting Using Ordinary Least Squares Algorithms," *Psychometrika*, 62, 251–261. [459]

Krzanowski, W. (1979), "Between-Groups Comparison of Principal Components," *Journal of the American Statistical Association*, 74, 703–707. [461]

Lemberge, P., De Raedt, I., Janssens, K., Wei, F., and Van Espen, P. J. (2000), "Quantitative Z-analysis of 16th–17th Century Archaeological Glass Vessels Using PLS Regression of EPXMA and μ-XRF Data," *Journal of Chemometrics*, 14, 751–763. [467]

Locantore, N., Marron, J., Simpson, D., Tripoli, N., Zhang, J., and Cohen, K. (1999), "Robust Principal Component Analysis for Functional Data," *Test*, 8, 1–73. [459]

Lopuhaä, H., and Rousseeuw, P. J. (1991), "Breakdown Points of Affine Equivariant Estimators of Multivariate Location and Covariance Matrices," *The Annals of Statistics*, 19, 229–248. [459]

Maronna, R. (2005), "Principal Components and Orthogonal Regression Based on Robust Scales," *Technometrics*, 47, 264–273. [459]

Nelson, P., Taylor, P., and MacGregor, J. (1996), "Missing Data Methods in PCA and PLS: Score Calculations With Incomplete Observations," *Chemometrics and Intelligent Laboratory Systems*, 35, 45–65. [459,464]

Odewahn, S., Djorgovski, S., Brunner, R., and Gal, R. (1998), "Data From the Digitized Palomar Sky Survey," Technical Report, California Institute of Technology. [468]

Raymaekers, J., Rousseeuw, P. J., Van den Bossche, W., and Hubert, M. (2019), *cellWise: Analyzing Data With Cellwise Outliers*, CRAN, R package version 2.1.0. [467,469,470]

Rousseeuw, P. J., and Hubert, M. (2018), "Anomaly Detection by Robust Statistics," *WIREs Data Mining and Knowledge Discovery*, e1236, 1–14. [459]

Rousseeuw, P. J., and Leroy, A. (1987), *Robust Regression and Outlier Detection*, New York: Wiley-Interscience. [461]

Rousseeuw, P. J., and Van den Bossche, W. (2018), "Detecting Deviating Data Cells," *Technometrics*, 60, 135–145. [460,462]

Schafer, J., and Graham, J. (2002), "Missing Data: Our View of the State of the Art," *Psychological Methods*, 7, 147–177. [460,471]

Serneels, S., and Verdonck, T. (2008), "Principal Component Analysis for Data Containing Outliers and Missing Elements," *Computational Statistics & Data Analysis*, 52, 1712–1727. [459]

Van Aelst, S., Vandervieren, E., and Willems, G. (2012), "A Stahel-Donoho Estimator Based on Huberized Outlyingness," *Computational Statistics & Data Analysis*, 56, 531–542. [459]

Walczak, B., and Massart, D. (2001), "Tutorial: Dealing With Missing Data, Part I," *Chemometrics and Intelligent Laboratory Systems*, 58, 15–27. [459,464]