JDSSV

Journal of Data Science, Statistics, and Visualisation

July 2021, Volume I, Issue III. *doi: 10.52933/jdssv.v1i3.18*

Handling Cellwise Outliers by Sparse Regression and Robust Covariance

Jakob Raymaekers
KU Leuven

Peter J. Rousseeuw
KU Leuven

Abstract

We propose a data-analytic method for detecting cellwise outliers. Given a robust covariance matrix, outlying cells (entries) in a row are found by the cellHandler technique which combines lasso regression with a stepwise application of constructed cutoff values. The penalty term of the lasso has a physical interpretation as the total distance that suspicious cells need to move in order to bring their row into the fold. For estimating a cellwise robust covariance matrix, we construct a detection-imputation method which alternates between flagging outlying cells and updating the covariance matrix as in the EM algorithm. The proposed methods are illustrated by simulations and on real data about volatile organic compounds in children.

Keywords: anomalous cells, cellHandler, detection-imputation method, marginal outliers, volatile organic compounds.

1. Introduction

It is a fact of life that most real data sets contain outliers, that is, elements that do not fit in with the majority of the data. These outliers can be annoying errors, but may also contain valuable information. In either case, finding them is of practical importance. In statistics this is called outlier detection, and in the computer science literature it is also called anomaly detection or exception mining (Chandola et al. 2009).

The most common paradigm is that of casewise outliers, which assumes that most cases were drawn from a certain model distribution but some other cases were not. The latter are also called rowwise outliers, since data often comes in the form of a table (matrix) in which the rows are the cases and the columns represent the variables. In computer science, one often uses outlier detection methods based on Euclidean distances, which by construction are invariant for orthogonal transformations of the rows. In statistics, many outlier detection methods are also invariant for affine transformations, that is, nonsingular linear transforms combined with shifts.

The study of cellwise outliers is a more recent research topic. This is the situation where some individual cells (entries) of the data matrix deviate from what they should have been. Alqallaf et al. (2009) first formulated this paradigm. Note that cells are intimately tied to the coordinate system, whereas orthogonal or other linear transformations would change the cells. To illustrate the difference between the rowwise and cellwise approaches, consider the standard multivariate Gaussian model in dimension $d = 4$ with the suspicious point $(10, 0, 0, 0)$. By an orthogonal transformation of the data, this point can be moved to $(\sqrt{50}, \sqrt{50}, 0, 0)$ or to $(5, 5, 5, 5)$, and therefore any orthogonally invariant rowwise detection method will yield the same result in all three situations. But in the cellwise paradigm, $(10, 0, 0, 0)$ has one outlying cell, $(\sqrt{50}, \sqrt{50}, 0, 0)$ has two, and $(5, 5, 5, 5)$ has four.

For an illustration of cellwise outliers see Figure 1. It depicts part of a dataset that will be described later. The rows are cases and the columns are variables. The regular cells are shown in yellow. Red colored cells indicate that their value is higher than expected, while blue cells indicate unusually low values.

When the model has substantially correlated variables, the cellwise outliers need not be marginally outlying, and then it can be quite hard to detect them. Van Aelst et al. (2011) proposed one of the first methods, based on an outlyingness measure of the Stahel-Donoho type. Farcomeni (2014) looks for the cells that, when put to missing, yield the highest Gaussian partial likelihood. Agostinelli et al. (2015) and Leung et al. (2017) use a univariate or bivariate filter on the variables to flag cellwise outliers, followed by S-estimation. Rousseeuw and Van den Bossche (2018) predict the values of all cells and flag the observed cells that differ much from their prediction. Debruyne et al. (2019) consider rowwise outliers and ask which variables contribute the most to their outlyingness. The O3 plot of Unwin (2019) visualizes cases that are outlying in lower dimensions.

There has also been substantial work to estimate the covariance matrix underlying the model in the presence of cellwise outliers, which will be briefly reviewed in Section 3.1.

Most of the statistical research on cellwise outliers has focused on the FICM contamination model of Alqallaf et al. (2009) which assumes that the outlying cells come from a single distribution, and typically this distribution has all its mass in a single value γ. Here we will not restrict ourselves to that setting, and in the simulations we will allow for the cellwise outlying values to depend on which cells are contaminated, creating structured cellwise outliers. This is a more challenging problem, and it is clear that the underlying covariance structure will play a role.

Note that the multivariate setting is very different from regression with a univariate response. In regression, having a robust fit is sufficient for flagging outlying responses,

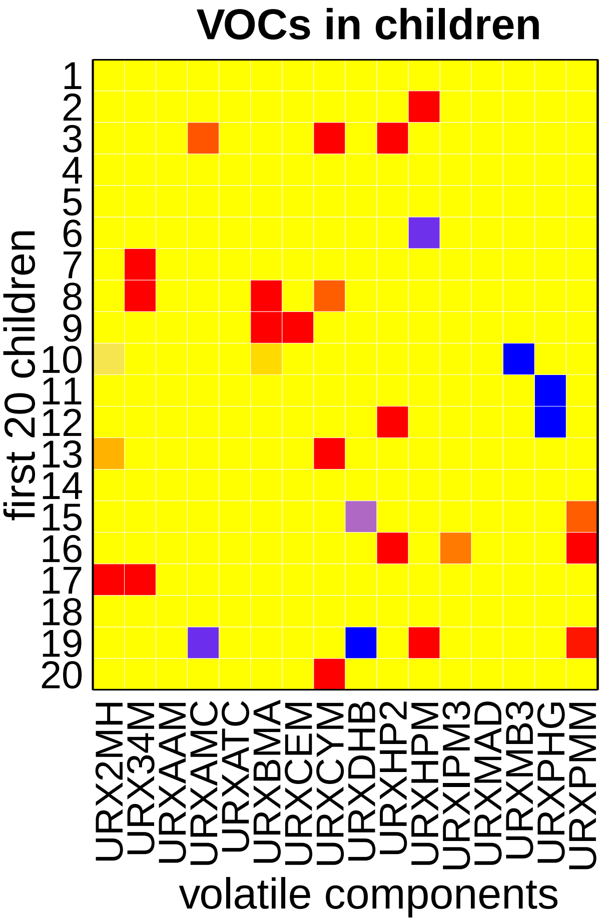

Figure 1: Illustration of cellwise outliers. Red squares indicate cells with unexpectedly high values, and blue squares indicate unusually low values. Regular cells are yellow.

because their residuals from the robust fit will be large in absolute value. In the multivariate situation it is much harder: even if we knew the true pre-contamination covariance matrix $\boldsymbol{\Sigma}$, how would we find the anomalous data cells? Currently, no method is available to do this. Our aim is to fill that gap by constructing such a method called cellHandler, described in Section 2. To estimate a cellwise robust covariance matrix $\hat{\boldsymbol{\Sigma}}$, Section 3 constructs the detection-imputation algorithm which alternates between cellHandler and re-estimating $\boldsymbol{\Sigma}$ as in the EM algorithm. In Section 5 the performance of this approach is studied by simulation, and Section 6 analyzes real data on volatile organic compounds in children.

2. The cellHandler Method

In this section, we construct a method to detect outlying cells when the true positive definite covariance matrix $\boldsymbol{\Sigma}$ is known. In reality $\boldsymbol{\Sigma}$ is usually unknown, but this method is a major component of the algorithm proposed in the next section for estimating $\boldsymbol{\Sigma}$.

2.1. Ranking Cells by their Outlyingness

We start by standardizing the columns (variables) of the dataset, using robust univariate estimates of location and scale such as the median and the median absolute deviation. This also ensures that the result will be equivariant to shifting and rescaling of the original variables. The resulting d-variate cases are denoted as $\boldsymbol{z}_i$ for $i = 1, \ldots, n$.

For a given case $\boldsymbol{z}$, the central question in this section is how we can identify the cells that are most likely to be contaminated. Any set of cells in $\boldsymbol{z}$ may be contaminated, and while it may be tempting to somehow investigate all 2^d subsets of $\boldsymbol{z}$, this quickly becomes infeasible due to the exponential complexity in d. Therefore, we need a different approach to provide candidate cells that may be contaminated while avoiding an insurmountable computational cost. Note that the squared Mahalanobis distance $\mathrm{MD}^2(\boldsymbol{z}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (\boldsymbol{z} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{z} - \boldsymbol{\mu})$ measures how far $\boldsymbol{z}$ lies from the uncontaminated distribution. The idea is to reduce the Mahalanobis distance of $\boldsymbol{z}$ by changing only a few cells. Mathematically, we look for a d-variate vector $\boldsymbol{\delta}$ such that $\mathrm{MD}^2(\boldsymbol{z} - \boldsymbol{\delta}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is small. Interestingly, this problem can be rewritten in an elegant form, as presented in the following proposition.

Proposition 1. *Modifying cells to reduce the Mahalanobis distance of their row can be rewritten using the sum of squares in a linear model.*

Proof. Observe that

$$\begin{aligned}
\mathrm{MD}^2(\boldsymbol{z} - \boldsymbol{\delta}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= (\boldsymbol{z} - \boldsymbol{\delta} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{z} - \boldsymbol{\delta} - \boldsymbol{\mu}) \\
&= ||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\delta} - \boldsymbol{\mu})||_2^2 \\
&= ||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu}) - \boldsymbol{\Sigma}^{-1/2}\boldsymbol{\delta}||_2^2 \\
&= ||\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\delta}||_2^2 \qquad (1)
\end{aligned}$$

which is the objective of a regression without intercept with known response vector $\tilde{\boldsymbol{Y}} := \boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu})$ and predictor matrix $\tilde{\boldsymbol{X}} := \boldsymbol{\Sigma}^{-1/2}$ with coefficient vector $\boldsymbol{\delta}$. Here $\boldsymbol{\Sigma}^{-1/2}$ is the unique PD inverse root of $\boldsymbol{\Sigma}$. □

It is clear that the ordinary least squares (OLS) solution to (1) is $\hat{\boldsymbol{\delta}}_{LS} = \boldsymbol{z} - \boldsymbol{\mu}$ since it makes the sum of squares zero. However, using $\hat{\boldsymbol{\delta}}_{LS}$ would replace the entire row by the vector $\boldsymbol{\mu}$, which would lose the information in the non-outlying cells. We prefer to change as few cells as possible, so we want a sparse coefficient vector $\hat{\boldsymbol{\delta}}$. A natural choice for this problem is the lasso (Tibshirani 1996), given by the minimization of

$$||\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\delta}||_2^2 + \lambda||\boldsymbol{\delta}||_1 \qquad (2)$$

where $||\boldsymbol{\delta}||_1 = |\delta_1| + \ldots + |\delta_d|$. Lasso regression penalizes $||\boldsymbol{\delta}||_1$ which yields a path of sparse solutions to the regression problem for decreasing value of λ. Note that the penalty term $||\boldsymbol{\delta}||_1$ has a concrete physical meaning in this setting: it is the total distance which the corresponding cells of $\boldsymbol{z}$ need to travel in order to bring $\boldsymbol{z}$ into the fold. This is unusual, as the L^1 term is typically included as a device to induce sparsity without a specific subject-matter interpretation.

The description is not yet complete, because we have to take special care of cells z_j that lie far away. Moving such cells into place requires large components δ_j which inflate the penalty term, so these z_j would appear rather late in the lasso path. Fortunately such far marginal outliers z_j are easy to spot, as they have a large univariate outlyingness $O_j = |z_j - \mu_j|/\sqrt{\Sigma_{jj}}$. Therefore, we downweight the δ_j in the penalty term by a factor $w_j = \min(1, 1.5/O_j)$ which is the weight associated with the univariate Huber M-estimator. This replaces $||\boldsymbol{\delta}||_1$ in (2) by $||\boldsymbol{W}\boldsymbol{\delta}||_1$ where $\boldsymbol{W} := diag(w_1, \ldots, w_d)$. Note that this weighted lasso can be rewritten as a plain lasso as follows. Since all the weights are strictly positive, $\boldsymbol{W}$ is invertible, so we can write $\tilde{\boldsymbol{X}}\boldsymbol{\delta} = (\tilde{\boldsymbol{X}}\boldsymbol{W}^{-1})(\boldsymbol{W}\boldsymbol{\delta}) = \dot{\boldsymbol{X}}\boldsymbol{\beta}$ where $\dot{\boldsymbol{X}} := \tilde{\boldsymbol{X}}\boldsymbol{W}^{-1}$ and $\boldsymbol{\beta} := \boldsymbol{W}\boldsymbol{\delta}$. This merely changes the units of the variables in $\tilde{\boldsymbol{X}}$, so we minimize

$$||\tilde{\boldsymbol{Y}} - \dot{\boldsymbol{X}}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1 \tag{3}$$

followed by transforming $\hat{\boldsymbol{\beta}}$ back to $\hat{\boldsymbol{\delta}}$. The penalty term $||\boldsymbol{\beta}||_1$ keeps its interpretation in the new units determined by the w_j.

Note that lasso steps do not only add variables: sometimes they take a variable out of the model. But in our context, it is natural to impose that once a cell is flagged it stays flagged, meaning that a selected regressor stays in the model. By imposing this constraint, we arrive at the elegant and fast least angle regression (LAR) algorithm of Efron et al. (2004). This is the option `type="lar"` in the R-package `lars` (Hastie and Efron 2015), and its performance turned out to be very similar to that of `type="lasso"` in our setting. Using LAR also simplifies and speeds up the next step of cellHandler in Section 2.2.

The way LAR works in our problem is intuitive. The gradient of $\mathrm{MD}^2(\boldsymbol{z} - \boldsymbol{\delta}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = ||\tilde{\boldsymbol{Y}} - \dot{\boldsymbol{X}}\boldsymbol{\beta}||_2^2$ with respect to $\boldsymbol{\beta}$ is $\boldsymbol{\nabla} = -2\dot{\boldsymbol{X}}'(\tilde{\boldsymbol{Y}} - \dot{\boldsymbol{X}}\boldsymbol{\beta})$. This gradient $\boldsymbol{\nabla} = (\nabla_1, \ldots, \nabla_d)$ is zero at the minimum of MD^2, when $\boldsymbol{\beta}$ is the OLS fit $(\dot{\boldsymbol{X}}'\dot{\boldsymbol{X}})^{-1}\dot{\boldsymbol{X}}\tilde{\boldsymbol{Y}} = \boldsymbol{W}(\boldsymbol{z} - \boldsymbol{\mu})$. LAR first takes the coordinate with highest $|\nabla_j|$ and moves β_j, that is, cell j, to reduce $|\nabla_j|$ until it equals the second largest $|\nabla_h|$. Then it moves cells j and h such that $|\nabla_j| = |\nabla_h|$ decrease together, until it reaches the third largest $|\nabla_m|$, and so on.

For each row $\boldsymbol{z}$, we have now obtained a ranking of its cells, corresponding to the order in which they occurred in the path for reducing $\mathrm{MD}^2(\boldsymbol{z} - \boldsymbol{\delta}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$. Each row $\boldsymbol{z}$ can have its own $\boldsymbol{\delta}$.

2.2. Handling Outlying Cells

After k steps of LAR, we have a set of k candidate cells. The question is whether these candidate cells are sufficient. In other words, is it possible to edit these k cells while keeping the remaining $d - k$ cells intact, in such a way that the edited row behaves like a clean row? To this end, we will edit the k candidate cells to maximize the Gaussian likelihood given the remaining cells. Suppose without loss of generality that the candidate cells are the first k entries of $\boldsymbol{z}$. Then we can denote $\boldsymbol{z}' = [\boldsymbol{z}_1' \ \boldsymbol{z}_2']$ and $\boldsymbol{\mu}' = [\boldsymbol{\mu}_1' \ \boldsymbol{\mu}_2']$ where $\boldsymbol{z}_1'$ and $\boldsymbol{\mu}_1'$ have length k. Also write $\boldsymbol{\Sigma}_{11}$ for the upper left submatrix of $\boldsymbol{\Sigma}$ of size $k \times k$ and so on. As in the E step of the EM algorithm (see for instance Little and Rubin (1987)), maximizing the Gaussian likelihood implies that $\boldsymbol{z}_1$ should be shifted to $E_{\boldsymbol{\mu},\boldsymbol{\Sigma}}[\boldsymbol{Z}_1|\boldsymbol{Z}_2 = \boldsymbol{z}_2] = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{z}_2 - \boldsymbol{\mu}_2)$.

This imputation appears to require inverting the submatrix $\boldsymbol{\Sigma}_{22}$ of $\boldsymbol{\Sigma}$. However, it can also be obtained by OLS regression in the model (1) of Proposition 1 but restricted to the set of k candidate variables. This is shown in the following proposition, the proof of which is given in Section A of the Appendix.

Proposition 2. *Let the k-variate $\hat{\boldsymbol{\theta}}_1$ be the OLS fit to the regression problem given by*

$$\text{argmin}_{\boldsymbol{\theta}} \,||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z}-\boldsymbol{\mu}) - (\boldsymbol{\Sigma}^{-1/2})_{.1}\boldsymbol{\theta}_1||_2^2$$

where $(\boldsymbol{\Sigma}^{-1/2})_{.1}$ denotes the first k columns of the matrix $\boldsymbol{\Sigma}^{-1/2}$. Then

$$\boldsymbol{z}_1 - \hat{\boldsymbol{\theta}}_1 = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{z}_2 - \boldsymbol{\mu}_2) \ \ .$$

In the implementation of cellHandler, these vectors $\hat{\boldsymbol{\theta}}_1$ are obtained as a byproduct of the LAR algorithm without extra computational cost; see Section C of the Appendix. This means that it carries out the above computation for $k = 1, \ldots, d$ without having to invert any matrix.

We now have a sequence of length d of cells in $\boldsymbol{z}$, with their possible imputations at every stage k. The question remains where to stop in this path, that is, how many cells should we actually flag? For that we use the following proposition:

Proposition 3. *For every $1 \leqslant k \leqslant d$ we have:*

1. *The residual sum of squares $RSS_k = ||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z}-\boldsymbol{\mu}) - (\boldsymbol{\Sigma}^{-1/2})_{.1}\hat{\boldsymbol{\theta}}_1||_2^2$ of the OLS fit $\hat{\boldsymbol{\theta}}_1$ to the first k cells in the path equals the squared partial Mahalanobis distance $MD^2(\boldsymbol{z}_2, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22}) = (\boldsymbol{z}_2 - \boldsymbol{\mu}_2)'\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{z}_2 - \boldsymbol{\mu}_2)$.*

2. *For Gaussian data, the difference between two subsequent RSS follows the χ^2 distribution with 1 degree of freedom, that is, $\Delta_k := RSS_{k-1} - RSS_k \sim \chi^2(1)$.*

The proof is in Section B of the Appendix. The distributional assumption in Part 2 is unrealistic in our setting, but at least the result provides a rough yardstick that we can use in our data-analytic procedure. Following the path for $1 \leqslant k \leqslant d$, we will compare the Δ_k to a cutoff q, say the 0.99 quantile of $\chi^2(1)$, and flag the cells with $\Delta_k > q$.

We illustrate cellHandler by two simple bivariate examples. The left part of Figure 2 assumes that the true $\boldsymbol{\mu} = \mathbf{0}$ and that $\boldsymbol{\Sigma}$ is the identity matrix, so the correlation ρ is zero. For any point $\boldsymbol{z} = [z_1 \ \ z_2]'$, we can then run cellHandler to see which of these cells, if any, are flagged. In the central square no cells are flagged, to its left and right z_1 is flagged, above and below it z_2 is flagged, and in the outer regions both z_1 and z_2 are flagged. Things get more eventful when $\boldsymbol{\Sigma}$ has 1 on the diagonal and $\rho = 0.9$ elsewhere. In the right panel of Figure 2, we see that no cells are flagged when $\boldsymbol{z}$ lies in part of an elliptical region. The domain where only z_1 is flagged now has a more complicated form, and the same holds for z_2 , whereas the region in which both are flagged is similar to before. Of course, the main purpose of cellHandler is to deal with higher dimensions, which are harder to visualize.

2.3. Simulation Study

To evaluate the performance of cellHandler we run a small simulation study in which the uncontaminated data are d-variate Gaussian with $\boldsymbol{\mu} = \mathbf{0}$. Since cellwise methods are neither affine or orthogonal invariant, we consider underlying covariance matrices $\boldsymbol{\Sigma}$ of

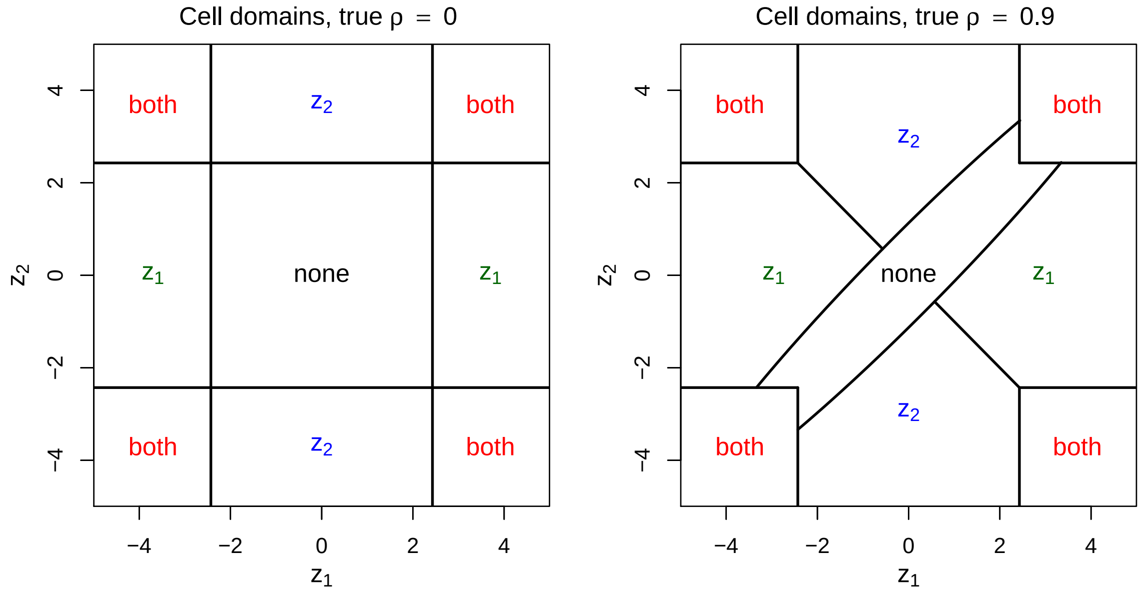

Figure 2: Bivariate domains where no cells are flagged, where only z_1 is flagged, where only z_2 is flagged, and where both are flagged, when the true correlation is $\rho = 0$ (left panel) and when $\rho = 0.9$ (right panel).

two types. Type ALYZ are the randomly generated correlation matrices of Agostinelli et al. (2015) which typically have relatively small correlations. Type A09 is given by $\Sigma_{jh} := (-0.9)^{|j-h|}$ and contains both large and small correlations.

The outlying cells are generated as follows. The positions of the cells to be contaminated are obtained by randomly drawing $n\varepsilon$ indices in each column of the data matrix. Then we look at each row $(z_1, \ldots, z_d)$ with such cells, and denote the indices of those cells as the set $K = \{j(1), \ldots, j(k)\}$ of size k. Next, we replace $(z_{j(1)}, \ldots, z_{j(k)})$ by the k-dimensional row $\boldsymbol{v} = \gamma\sqrt{k}\,\boldsymbol{u}'/\,\mathrm{MD}(\boldsymbol{u}, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_K)$ where $\boldsymbol{\mu}_K$ and $\boldsymbol{\Sigma}_K$ are restricted to the indices in K and where $\boldsymbol{u}$ is the eigenvector of $\boldsymbol{\Sigma}_K$ with smallest eigenvalue. This procedures generates $\boldsymbol{v}$ which are structurally outlying in the subspace of the coordinates in K, while many of these cells will not be marginally outlying. This produces cellwise outliers that are more challenging than in the earlier literature, which used $\boldsymbol{v} = (\gamma, \ldots, \gamma)$.

Figure 3 shows the performance of cellHandler on samples of size $n = 400$ in $d = 20$ dimensions with $\varepsilon = 20\%$ of cellwise outliers, using the covariance matrix estimated by the algorithm DDCW.DI described in Section 3.2. The other curves are from three existing techniques for flagging cells. The first one is the univariate Gervini-Yohai filter (GY) specified in (Agostinelli et al. 2015). The second is the multivariate DetectDeviatingCells (DDC) algorithm of Rousseeuw and Van den Bossche (2018), available in the `cellWise` package (Raymaekers et al. 2020) as the function `DDC`. The third is the default filter of the 2SGS method in (Leung et al. 2017), which is a combination of a bivariate GY filter with DDC. The top panels in Figure 3 show the *recall*, which is the fraction of generated cellwise outliers that are flagged as such. The data in the left plot were generated by the contaminated ALYZ model, and on the right by the contaminated A09 model. We see that cellHandler has the highest recall at each γ. When γ

increases, the cellwise outliers become marginally outlying, making them easier to flag. The middle row of the figure shows the *precision*, which is the fraction of cells flagged as outlying that were generated as such. We see that cellHandler does not have the best precision among competing methods at high γ, which is due to the tradeoff between

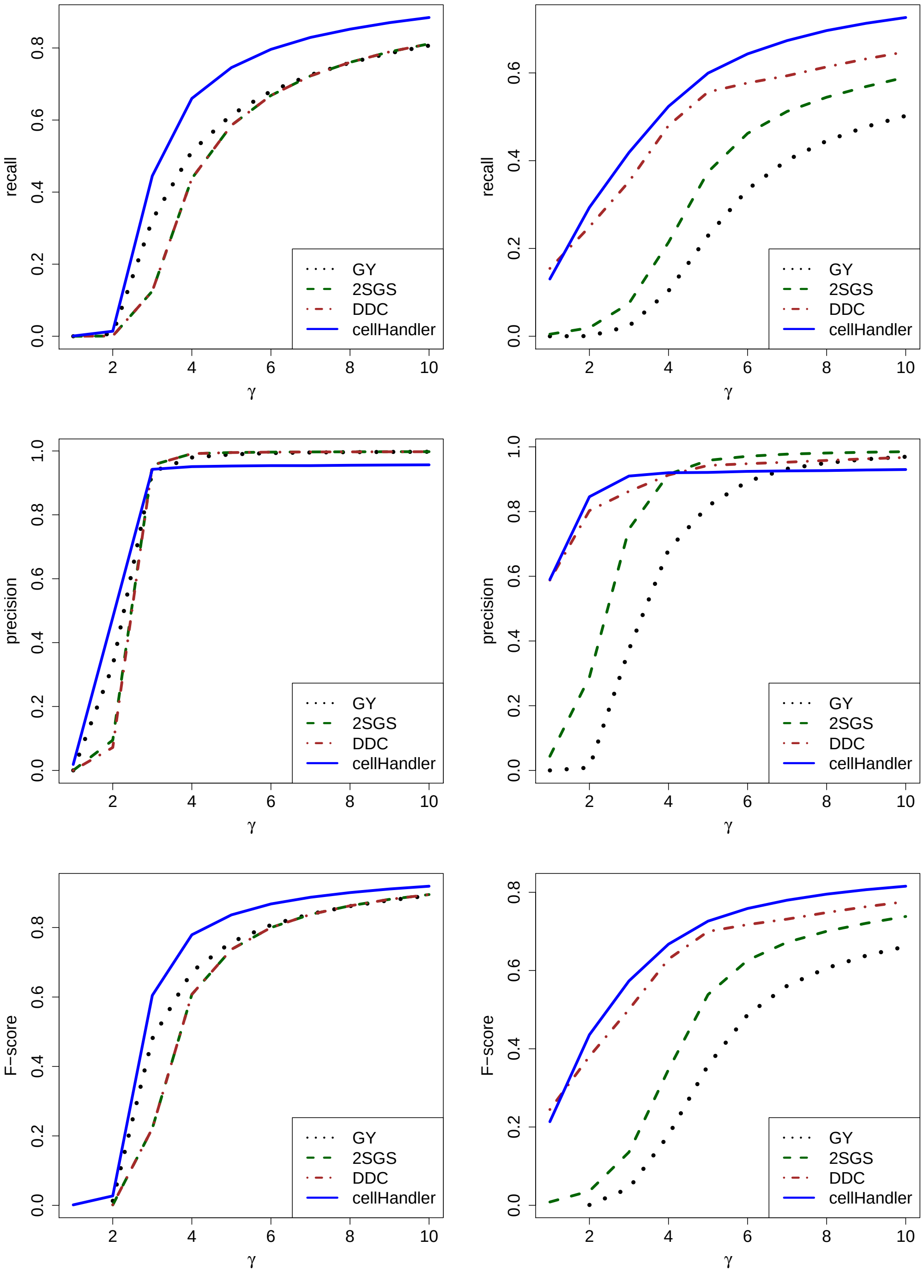

Figure 3: Comparison of methods for detecting cellwise outliers on data generated by the contaminated ALYZ model (left) and the contaminated A09 model (right), for $n = 400$ points in $d = 20$ dimensions. The plots show recall (top), precision (middle), and F-score (bottom).

precision and recall. Finally, the bottom row shows the F-score, also called the Dice coefficient (Dice 1945), which summarizes the performance of a binary classification through the harmonic mean of precision and recall. Based on this summary measure, cellHandler performs best.

3. Cellwise Robust Estimation of a Covariance Matrix

3.1. Existing Approaches

The previous section described a method for flagging cellwise outliers when the true center $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ are known. Of course these are rarely given in practice, so they have to be estimated. The center $\boldsymbol{\mu}$ can be estimated quite easily by applying a robust estimator (like the median) to each coordinate. Estimating the covariance matrix $\boldsymbol{\Sigma}$ is the hard part. There exist several approaches to this problem.

A popular technique is to compute robust covariances between each pair of variables, and to assemble them in a matrix. To estimate these pairwise covariances, Öllerer and Croux (2015) and Croux and Öllerer (2016) use rank-based methods such as the Spearman and normal scores correlations. Tarr et al. (2016) instead propose the use of the robust pairwise correlation estimator of Gnanadesikan and Kettenring (1972) in combination with the robust scale estimator Q_n of Rousseeuw and Croux (1993). As the resulting matrix is not necessarily positive semidefinite (PSD), they then compute the nearest PSD matrix by the algorithm of Higham (2002). All of these pairwise covariance estimators are fast to compute. We will compare the performance of these methods in Section 5.

A second approach is the snipEM procedure proposed by Farcomeni (2014) and implemented in the R package `snipEM` of Farcomeni and Leung (2019). Its first step flags cellwise outliers in each variable separately using a boxplot rule, and then "snips" them, which means making them missing. The second step tries many interchanges that unsnip a randomly chosen snipped cell and at the same time snip a randomly chosen unsnipped cell, and only keeps an interchange when it increases the partial Gaussian likelihood. This procedure is slower than the pairwise covariance approach.

The current state of the art to deal with complex cellwise outliers is the two-step generalized S-estimator (2SGS) of Agostinelli et al. (2015) and Leung et al. (2017) implemented in the R package `GSE` (Leung et al. 2019). In a first step, the method uses a filter (called 2SGS in Figure 3 above) to detect cellwise outliers. These cells are then set to missing, and the generalized S-estimator of Danilov et al. (2012) is run. A short survey of cellwise robust covariance estimators can be found in Sections 6.13 and 6.14 of Maronna et al. (2019).

3.2. The Detection-Imputation Algorithm

Our algorithm for constructing a cellwise robust covariance matrix starts by standardizing the columns of the dataset as in the beginning of Section 2.1. Next, we compute initial estimators $\hat{\boldsymbol{\mu}}^0$ and $\widehat{\boldsymbol{\Sigma}}^0$. For this we can use the 2SGS estimator of Leung et al. (2017) described above. We will also try a different initial estimator called DDCW, which is a combination of the DDC method (Rousseeuw and Van den Bossche 2018) and the wrapped covariance matrix of Raymaekers and Rousseeuw (2019). This initial estimator is described in Section D of the Appendix.

The detection-imputation (DI) algorithm then alternates the D-step and the I-step, both described below.

D-step: detecting outlying cells across all rows.
The D-step first applies the cellHandler method of Section 2 to each row $(z_{i1},\ldots,z_{id})$ based on the estimates $\hat{\boldsymbol{\mu}}^{t-1}$ and $\widehat{\boldsymbol{\Sigma}}^{t-1}$ from the previous iteration step. This creates a path with steps $k=1,2,\ldots,d$ and their Δ_k. Note that $\Delta_1,\ldots,\Delta_d$ is not necessarily monotone, but we create the nonincreasing sequence $D_k := \max\{\Delta_k,\ldots,\Delta_d\}$ for $k=1,2,\ldots,d$. Missing cells are put in front of the path, with $D_k := +\infty$. Finally, we create an $n\times d$ matrix $\boldsymbol{C}$ whose i-th row contains the D_k values of $(z_{i1},\ldots,z_{id})$, in fact $C_{ij}=D_k$ when cell z_{ij} is obtained in step k.

Should some columns have too many flagged cells (including NA's), it could become difficult to estimate a correlation between them, especially if the flagged sets overlap little. Even worse, flagging all cells in a column would remove all information about that variable. Therefore, we impose a maximal number of flagged cells in each column, including the NA's. This number is n `maxCol` where the input parameter `maxCol` is set to 25% by default. Note that this is a constraint on the columns, whereas we are flagging cells by row. We resolve this with the following algorithm:

- sort the criterion values C_{ij} of all cells in the matrix in decreasing order;

- walk down this list. If a C_{ij} lies below the cutoff value q we "lock" row i, meaning that no cells of row i can be flagged any more. If $C_{ij} > q$ the cell is flagged, unless it belongs to a column which already has n `maxCol` flagged cells. In the latter case, row i is locked also.

This procedure yields a (possibly empty) list of flagged cells in each row, which overall contains the most outlying cells subject to the `maxCol` constraint.

I-step: Re-estimate μ and $\boldsymbol{\Sigma}$.
The I-step is basically one step of the EM algorithm which considers the flagged cells as missing. However, it is computationally more efficient since it reuses results that are already available. In each row, the set of flagged cells is one of the active sets considered by LAR in cellHandler, so its coefficient $\hat{\boldsymbol{\theta}}_1$ from Proposition 2 is known. This makes it trivial to impute the flagged cells, so the E-step of EM requires no additional computation. Next, $\hat{\boldsymbol{\mu}}^t$ and $\widehat{\boldsymbol{\Sigma}}^t$ are computed as in the M-step, as described in more detail in Section E of the Appendix. This iterative procedure stops when both $\hat{\boldsymbol{\mu}}^t - \hat{\boldsymbol{\mu}}^{t-1}$ and $\widehat{\boldsymbol{\Sigma}}^t - \widehat{\boldsymbol{\Sigma}}^{t-1}$ are small. At the end of the DI algorithm, we unstandardize $\hat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\Sigma}}$ using the univariate location and scale estimates of the original data columns.

The time complexity of the DI algorithm is $O(Tnd^3)$, where T is the number of iteration steps. This is the same complexity as that of the classical EM algorithm for covariance estimation with missing data.

Note that for the DI method to work, the initial covariance matrix (whether 2SGS or DDCW) and those in all iteration steps need to be invertible. This requires that $n > d$, so for now the approach does not allow for $d \geqslant n$. Possible extensions are a topic for further research, and would likely require penalization or other forms of regularization.

4. Measuring Scatter Matrix Discrepancy

In the simulation study in the next section, we want to measure how much an estimated scatter matrix deviates from the true underlying positive definite (PD) scatter matrix. For this we need a discrepancy measure for scatter matrices. Here we will construct a pre-existing discrepancy measure from first principles, in order to dispel a common misconception that this measure would only make sense when the underlying data follow a multivariate normal (Gaussian) distribution.

Suppose we want to measure how much a scatter matrix $\boldsymbol{A}$ deviates from a reference scatter matrix $\boldsymbol{B}$, where the $d \times d$ matrix $\boldsymbol{B}$ is PD but $\boldsymbol{A}$ only needs to be positive semidefinite (PSD). A simple measure of this type is

$$||\boldsymbol{A}-\boldsymbol{B}||_2 = \left(\sum_{i=1}^{d}\sum_{j=1}^{d}(a_{ij}-b_{ij})^2\right)^{1/2} \tag{4}$$

but it is insufficiently suited to our scatter matrix context, as it does not tell us whether $\boldsymbol{A}$ is singular. And this is important, since a singular scatter matrix $\boldsymbol{A}$ cannot be used as an approximation of $\boldsymbol{B}$, for instance when computing a Mahalanobis-style statistical distance as in (1) which requires the inverse matrix.

In order to stay in the realm of scatter matrices, we instead compute

$$\boldsymbol{S} = \boldsymbol{B}^{-1/2}\boldsymbol{A}\,\boldsymbol{B}^{-1/2}\,.$$

(The matrix $\boldsymbol{S}$ can be seen as the scatter $\boldsymbol{A}$ in the coordinate system where $\boldsymbol{B}$ is sphered/whitened, since $\boldsymbol{B}^{-1/2}\boldsymbol{B}\,\boldsymbol{B}^{-1/2} = \boldsymbol{I}$.) Note that $\boldsymbol{A} = \boldsymbol{B}$ if and only if $\boldsymbol{S} = \boldsymbol{I}$, so we want to measure how far $\boldsymbol{S}$ is from $\boldsymbol{I}$ in a way that is relevant for scatter matrices. Since the matrix $\boldsymbol{S}$ is PSD, its eigenvalues are nonnegative, so we can denote them as $\eta_1 \geqslant \ldots \geqslant \eta_d \geqslant 0$. We want the discrepancy measure to be zero if all $\eta_j = 1$, to go to $+\infty$ when $\boldsymbol{S}$ explodes in the sense that $\eta_1 \to +\infty$, and also when $\boldsymbol{S}$ implodes, that is, $\eta_d \to 0$. Concentrating on a single eigenvalue η we want a continuous function $h(\eta)$ on all $\eta \geqslant 0$ with the properties $h(\eta) \geqslant 0$, $h(1) = 0$, and $h(\eta) \to +\infty$ when $\eta \to +\infty$ or $\eta \to 0$. Many such functions can be constructed. One of them is $h(\eta) = \eta - 1 - \log(\eta)$. Note that $h(\eta) \geqslant 0$ since $\log(\eta)$ is concave and $\eta - 1$ is its tangent line at $\eta = 1$. The function h decreases on $[0, 1[$, reaches its minimum in 1 with $h(1) = 0$, and increases on $]1, +\infty[$. Therefore, it makes sense to define the discrepancy of $\boldsymbol{A}$ relative to $\boldsymbol{B}$ as

$$D(\boldsymbol{A},\boldsymbol{B}) := \sum_{j=1}^{d} h(\eta_j) = \sum_{j=1}^{d}(\eta_j - 1 - \log(\eta_j)) \tag{5}$$

which is nonnegative since each term is. Note that $\boldsymbol{A} = \boldsymbol{B}$ is equivalent to $D(\boldsymbol{A},\boldsymbol{B}) = 0$, and that a singular $\boldsymbol{A}$ attains $D(\boldsymbol{A},\boldsymbol{B}) = +\infty$.

The entire construction of $D(\boldsymbol{A},\boldsymbol{B})$ above only uses the PSD property of scatter matrices, and is not at all restricted to multivariate normally distributed data. That confusion is due to the following property:

Proposition 4. *In the special case where $\boldsymbol{X}$ and $\boldsymbol{Y}$ are d-variate random vectors distributed as $\boldsymbol{X} \sim N(\boldsymbol{0},\boldsymbol{A})$ and $\boldsymbol{Y} \sim N(\boldsymbol{0},\boldsymbol{B})$ in which both $\boldsymbol{A}$ and $\boldsymbol{B}$ are PD, the discrepancy $D(\boldsymbol{A},\boldsymbol{B})$ coincides with the Kullback-Leibler divergence $KL(\boldsymbol{X},\boldsymbol{Y})$.*

The proof is in Section F of the Appendix. Note that when both $\boldsymbol{A}$ and $\boldsymbol{B}$ are PD, $D(\boldsymbol{B}, \boldsymbol{A})$ exists also, but it does not equal $D(\boldsymbol{A}, \boldsymbol{B})$. Instead we obtain, along the same lines, $D(\boldsymbol{B}, \boldsymbol{A}) = \sum_{j=1}^{d} h(1/\eta_j)$. However, it is possible to symmetrize the discrepancy $D(.,.)$ by replacing the h in (5) by a function $\tilde{h}$ for which $\tilde{h}(1/\eta) = \tilde{h}(\eta)$ for all $\eta > 0$, such as $\tilde{h}(\eta) = \eta + 1/\eta - 2$. One could also use the function $\tilde{h}(\eta) = |\log(\eta)|$ so $D(\boldsymbol{A}, \boldsymbol{B})$ becomes the L^1 norm of $(\log(\eta_1), \ldots, \log(\eta_d))$.

5. Simulation Results

We simulate the estimators of covariance matrices discussed in the previous section. The data is generated as in Subsection 2.3, with dimensions $d = 10$, 20 and 40. The fraction of contaminated cells is $\varepsilon = 0.1, 0.2$ in which γ varies from 1 to 10. In each replication, we compute the discrepancy (5) of the estimate $\widehat{\boldsymbol{\Sigma}}$ from the underlying $\boldsymbol{\Sigma}$, and then average the discrepancy over all replications. We show the results for $\varepsilon = 0.2$, since this is the most challenging scenario. The results for $\varepsilon = 0.1$ were qualitatively similar.

Figure 4 compares the proposed methods to the existing approaches described in Subsection 3.1, for $d = 10$. Since $\varepsilon = 0.2$, there are on average two cellwise outliers per row. Gaussian rank (Grank) and Spearman refer to the covariance matrices of Öllerer and Croux (2015) and Croux and Öllerer (2016) using those rank correlations. The Gnanadesikan-Kettenring procedure of Tarr et al. (2016) is labeled GKnpd. Next, the snipEM method of Farcomeni (2014) and the 2SGS estimator of Leung et al. (2017) are plotted. The method 2SGS.DI uses 2SGS as initial estimator followed by the new DI method of Section 3.2. The initial estimator DDCW described in Section D of the Appendix is also shown, as well as DI applied to it.

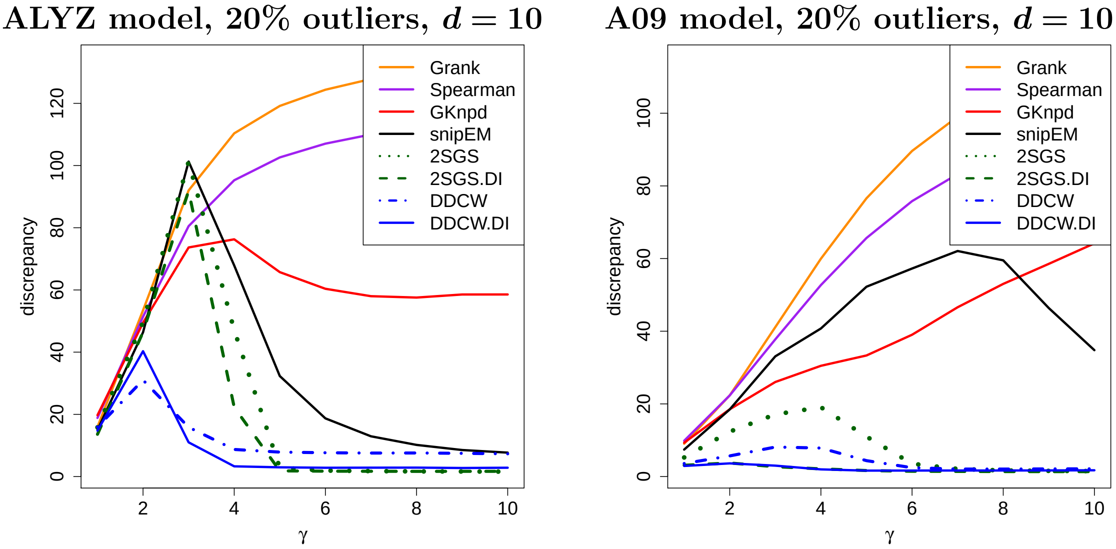

Figure 4: Discrepancy $D(\widehat{\boldsymbol{\Sigma}}, \boldsymbol{\Sigma})$ of estimated covariance matrices for $d = 10$, $n = 100$.

We see that the three pairwise methods Grank, Spearman and GKnpd pay for their fast computation by a high discrepancy. The snipEM method does better for high γ, in part because the boxplot rule in its first step snips marginally outlying cells. The three pairwise methods do not use such a rule to flag marginally outlying cells, so high γ values impact them more. The state of the art method 2SGS does substantially better, and is improved by applying DI to it, both in the ALYZ and A09 models. The same holds for DDCW and DDCW.DI. Note that DI improves the results more under A09 than ALYZ, because A09 has bigger correlations so DI has more opportunities to make a difference.

We now consider higher dimensions, starting with $d = 20$ in the top panels of Figure 5. The curves of Grank, Spearman, GKnpd and snipEM were much higher in this case, so we only show the four best performing methods in order to see the differences between them. Here the DI algorithm substantially improves upon the initial estimators also. The improvement is largest under A09 which contains some high correlations. For $d = 40$ (bottom panels) we see similar patterns.

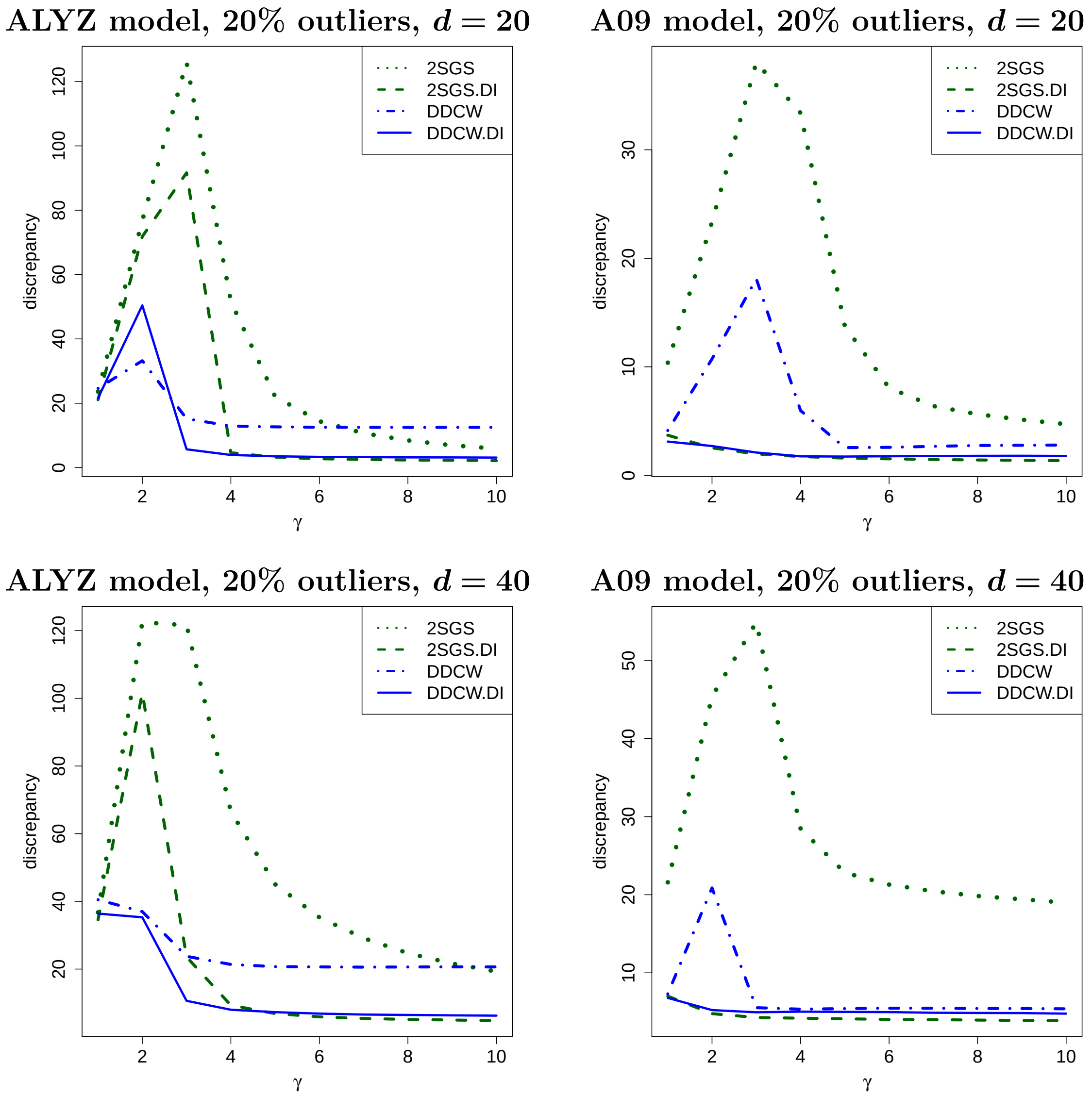

Figure 5: Discrepancy $D(\widehat{\boldsymbol{\Sigma}}, \boldsymbol{\Sigma})$ given by (5) of estimated covariance matrices for $d = 20$ and $n = 400$ (top panels) and for $d = 40$ and $n = 800$ (bottom panels).

Section I of the Appendix shows the results of a simulation in which the data are contaminated by 10% of cellwise outliers generated as above, plus 10% of rowwise outliers. In this particular setting "rowwise outliers" refers to rows in which all cells are contaminated in the same way as before, that is, rows with d cellwise outliers. The initial estimators 2SGS and DDCW attempt to downweight or discard such rows. The results are qualitatively similar to those in Figures 4 and 5.

6. Example: Volatile Organic Compounds in Children

We study a dataset of volatile organic compounds (VOCs) in human urinary samples. The data was taken from the publicly available website of the National Health and Nutrition Examination Survey (NHANES 2019), using the most recent available epoch. Such VOC metabolites are commonly monitored since chronic exposure to high levels of some VOCs can lead to a number of health problems such as cancer and neurocognitive dysfunction. The original dataset consists of 29 VOC metabolites, but we focus on a subset of 16 variables obtained by removing columns with a lot of missing values and/or zero median absolute deviation. Section J in the Appendix contains a table with the VOCs analyzed. In order to obtain a relatively homogeneous subset, we selected the data for children aged 10 or younger. The final dataset contained 512 subjects. We log-transformed the concentrations to make the variables roughly Gaussian (apart from possible outliers).

We estimated the covariance matrix of the data by the DI algorithm, starting from the DDCW initial estimator. The algorithm converged after 7 steps. Using the resulting covariance estimate, we ran the cellHandler algorithm with cutoff $\sqrt{\chi^2_{1,0.99}} \approx 2.57$ to detect outlying cells. The corresponding cellmap of the first 20 children in the list was shown as Figure 1 in the introduction. Each row of the cellmap corresponds to a child, with inlying cells colored yellow. Red cells indicate that their value is higher than predicted given the inlying cells of that row, while blue cells indicate lower than predicted values. The more extreme the residual, the more intense the color.

One variable that stood out was URXCYM (N-Acetyl-S-(2-cyanoethyl)-L-cysteine) in which cellHandler indicated 11% of large cell residuals. This was particularly striking since that variable had fewer than 2% of marginal outliers using the same cutoff $\sqrt{\chi^2_{1,0.99}}$ on the absolute standardized values, and these were rather nearby (note that even for perfectly Gaussian data there would already be 1% of absolute standardized values above this cutoff). Figure 6 plots the cell residuals (which are zero for cells that were not flagged) versus the robustly standardized marginal values, with the cutoffs indicated by horizontal and vertical lines. Most of the outlying cellwise residuals correspond to inlying marginal values. These children have extreme URXCYM values relative to their other VOCs.

Interestingly, URXCYM is a well-known biomarker for identifying smokers among adults (Chen et al. 2019), since it typically results from the metabolization of acrylonitrile, a volatile liquid present in tobacco smoke. But in this example we are studying children, who are not supposed to smoke. In search of an explanation, we combined the VOC data with the questionnaire data available on the same website (NHANES 2019). Among many other things, these data contain information on the smoking status of

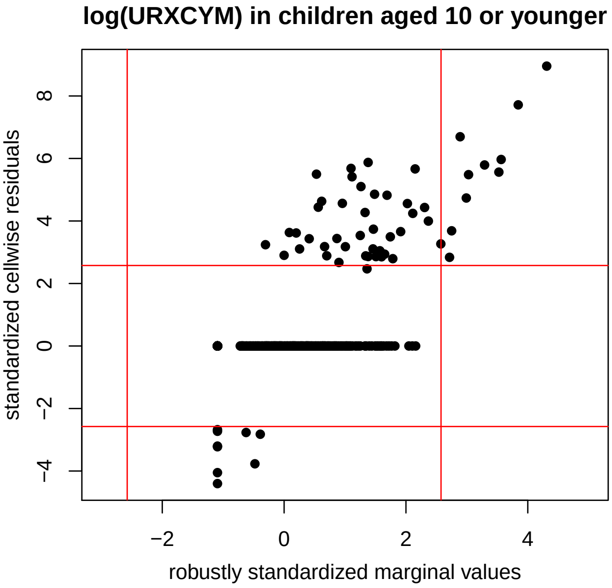

Figure 6: Plot of standardized cell residuals of log(URXCYM) obtained by cellHandler, versus the robustly standardized values of log(URXCYM) on its own.

the adults (usually parents) in the same household. These fell into four categories: only nonsmoking adults, smoking adults who do not smoke inside the home, one adult smoking in the home, and two adults smoking in the home. The blue curve in Figure 7 shows the percentage of children with URXCYM cell residuals above the cutoff, in each of these categories. They go from 4.7% in households with only nonsmoking adults up to 72.7% in homes where two adults smoke, indicating that passive smoking has a measurable effect on children. On the other hand, if we were to look only at the marginal URXCYM values (red curve), no such effect is visible.

The example shows that the effect of exposing children to tobacco smoke could be underestimated when only performing univariate analyses on biomarkers. This illustrates that cell residuals obtained by cellHandler may add valuable information to a dataset.

7. Conclusion

The proposed cellHandler method is the first to detect cellwise outliers based on robust estimates of location and covariance. It is also a major component of the detection-imputation (DI) algorithm that computes such cellwise robust estimates. Note that both methods can deal with missing values in the data, since these are imputed along the way.

The performance of cellHandler and DI was illustrated by simulation. A real example illustrated that the common medical practice of comparing individual biomarkers to their tolerance limits can benefit from the use of cellwise residuals.

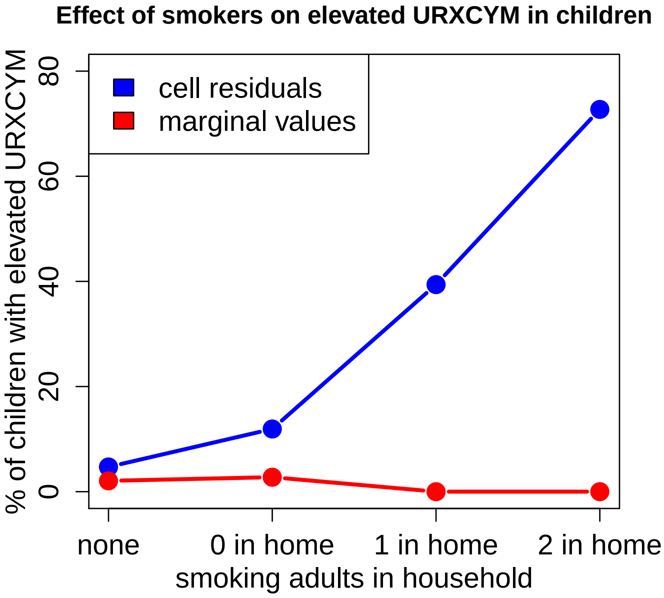

Figure 7: The blue curve shows the percentage of elevated URXCYM cell residuals in function of the smoking status of adult family members. The red curve shows the percentage of elevated marginal URCYM values.

Acknowledgments

This research was funded by projects of Internal Funds KU Leuven.

Software Availability

The methods proposed here are available as the functions `cellHandler` and `DI` in the `R` package `cellWise` (Raymaekers et al. 2020) on CRAN. It also contains a vignette which reproduces the example in Section 6.

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, ISSN: 1863-8260, DOI: 10.1007/s11749-015-0450-6.

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, DOI: 10.1214/07-AOS588.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41:1–58, DOI: 10.1145/1541880.1541882.

Chen, M., Carmella, S. G., Sipe, C., Jensen, J., Luo, X., Le, C. T., Murphy, S. E., Benowitz, N. L., McClernon, F. J., Vandrey, R., Allen, S. S., Denlinger-Apte, R., Cinciripini, P. M., Strasser, A. A., al'Absi, M., Robinson, J. D., Donny, E. C., Hatsukami, D., and Hecht, S. S. (2019). Longitudinal stability in cigarette smokers of urinary biomarkers of exposure to the toxicants acrylonitrile and acrolein. *PLOS ONE*, 14(1):1–13, DOI: 10.1371/journal.pone.0210104.

Croux, C. and Öllerer, V. (2016). Robust and sparse estimation of the inverse covariance matrix using rank correlation measures. In Agostinelli, C., Basu, A., Filzmoser, P., and Mukherjee, D., editors, *Recent Advances in Robust Statistics: Theory and Applications*, pages 35–55, New Delhi. Springer India, DOI: 10.1007/978-81-322-3643-6_3.

Danilov, M. (2010). *Robust estimation of multivariate scatter in non-affine equivariant scenarios*. PhD thesis, University of British Columbia.

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, DOI: 10.1080/01621459.2012.699792.

Debruyne, M., Höppner, S., Serneels, S., and Verdonck, T. (2019). Outlyingness: Which variables contribute most? *Statistics and Computing*, 29:707–723, DOI: 10.1007/s11222-018-9831-5.

Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology*, 26:297–302, DOI: 10.2307/1932409.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, 32:407–499, DOI: 10.1214/009053604000000067.

Farcomeni, A. (2014). Robust constrained clustering in presence of entry-wise outliers. *Technometrics*, 56(1):102–111, DOI: 10.1080/00401706.2013.826148.

Farcomeni, A. and Leung, A. (2019). *Package snipEM: Snipping Methods for Robust Estimation and Clustering*. CRAN, R package version 1.0.1, https://CRAN.R-project.org/package=snipEM.

Gnanadesikan, R. and Kettenring, J. R. (1972). Robust estimates, residuals, and outlier detection with multiresponse data. *Biometrics*, 28:81–124, ISSN: 0006341X, 15410420, http://www.jstor.org/stable/2528963.

Hastie, T. and Efron, B. (2015). *Package lars: Least Angle Regression, Lasso and Forward Stagewise*. CRAN, R package version 1.2, https://CRAN.R-project.org/package=lars.

Higham, N. J. (2002). Computing the nearest correlation matrix - a problem from finance. *IMA Journal of Numerical Analysis*, 22:329–343, ISSN: 0272-4979, DOI: 10.1093/imanum/22.3.329.

Leung, A., Danilov, M., Yohai, V., and Zamar, R. (2019). *Package GSE: Robust Estimation in the Presence of Cellwise and Casewise Contamination and Missing Data*. CRAN, R package version 4.2, https://CRAN.R-project.org/package=GSE.

Leung, A., Yohai, V., and Zamar, R. (2017). Multivariate location and scatter matrix estimation under cellwise and casewise contamination. *Computational Statistics & Data Analysis*, 111:59–76, ISSN: 0167-9473, DOI: 10.1016/j.csda.2017.02.007.

Little, R. and Rubin, D. (1987). *Statistical Analysis with Missing Data.* Wiley-Interscience, New York.

Maronna, R., Martin, D., Yohai, V., and Salibián-Barrera, M. (2019). *Robust Statistics: Theory and Methods, Second Edition.* Wiley, New York.

NHANES (2019). National Health and Nutrition Examination Survey Data. Hyattsville, MD: U.S. Department of Health and Human Services, Centers for Disease Control and Prevention. https://wwwn.cdc.gov/nchs/nhanes/.

Öllerer, V. and Croux, C. (2015). Robust high-dimensional precision matrix estimation. In Nordhausen, K. and Taskinen, S., editors, *Modern Nonparametric, Robust and Multivariate Methods: Festschrift in Honour of Hannu Oja*, pages 325–350. Springer International Publishing, Cham, DOI: 10.1007/978-3-319-22404-6_19.

Petersen, K. B. and Pedersen, M. S. (2012). *The Matrix Cookbook.* Technical University of Denmark, http://www2.imm.dtu.dk/pubdb/pubs/3274-full.html.

Raymaekers, J. and Rousseeuw, P. J. (2019). Fast robust correlation for high-dimensional data. DOI: 10.1080/00401706.2019.1677270. *Technometrics*, published online, open access.

Raymaekers, J., Rousseeuw, P. J., Van den Bossche, W., and Hubert, M. (2020). *Package cellWise: Analyzing Data with Cellwise Outliers.* CRAN, R package version 2.2.2, https://CRAN.R-project.org/package=cellWise.

Rousseeuw, P. J. and Croux, C. (1993). Alternatives to the median absolute deviation. *Journal of the American Statistical Association*, 88:1273–1283, DOI: 10.1080/01621459.1993.10476408.

Rousseeuw, P. J. and Van den Bossche, W. (2018). Detecting deviating data cells. *Technometrics*, 60:135–145, DOI: 10.1080/00401706.2017.1340909. Open access.

Tarr, G., Müller, S., and Weber, N. (2016). Robust estimation of precision matrices under cellwise contamination. *Computational Statistics & Data Analysis*, 93:404–420, ISSN: 0167-9473, DOI: 10.1016/j.csda.2015.02.005.

Templ, M., Hron, K., and Filzmoser, P. (2020). *Package robCompositions: Compositional Data Analysis.* CRAN, R package version 2.2.1, https://CRAN.R-project.org/package=robCompositions.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B*, 58:267–288, https://www.jstor.org/stable/2346178.

Unwin, A. (2019). Multivariate outliers and the O3 plot. *Journal of Computational and Graphical Statistics*, 28:635–643, DOI: 10.1080/10618600.2019.1575226.

Van Aelst, S., Vandervieren, E., and Willems, G. (2011). Stahel-Donoho estimators with cellwise weights. *Journal of Statistical Computation and Simulation*, 81(1):1–27, DOI: 10.1080/00949650903103873.

A. Proof of Proposition 2

Proof. We first split up the relevant matrices into blocks. Denote

$$\boldsymbol{\Sigma}^{-1} = \begin{bmatrix} \boldsymbol{\Sigma}^*_{11} & \boldsymbol{\Sigma}^*_{12} \\ \boldsymbol{\Sigma}^*_{21} & \boldsymbol{\Sigma}^*_{22} \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{\Sigma}^{-1/2} = \begin{bmatrix} \tilde{\boldsymbol{\Sigma}}_{11} & \tilde{\boldsymbol{\Sigma}}_{12} \\ \tilde{\boldsymbol{\Sigma}}_{21} & \tilde{\boldsymbol{\Sigma}}_{22} \end{bmatrix} .$$

Let the k-variate $\hat{\boldsymbol{\theta}}$ be the solution to the ordinary least squares regression problem

$$\text{argmin}_{\boldsymbol{\theta}} \, ||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu}) - (\boldsymbol{\Sigma}^{-1/2})_{.1}\boldsymbol{\theta}||_2^2$$

where $(\boldsymbol{\Sigma}^{-1/2})_{.1}$ denotes the first k columns of the matrix $\boldsymbol{\Sigma}^{-1/2}$.
Then we know that

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= \left(\begin{bmatrix} \tilde{\boldsymbol{\Sigma}}_{11} \\ \tilde{\boldsymbol{\Sigma}}_{21} \end{bmatrix}' \begin{bmatrix} \tilde{\boldsymbol{\Sigma}}_{11} \\ \tilde{\boldsymbol{\Sigma}}_{21} \end{bmatrix} \right)^{-1} \begin{bmatrix} \tilde{\boldsymbol{\Sigma}}_{11} \\ \tilde{\boldsymbol{\Sigma}}_{21} \end{bmatrix}' \boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu}) \\ &= (\boldsymbol{\Sigma}^*_{11})^{-1} \begin{bmatrix} \tilde{\boldsymbol{\Sigma}}_{11} \\ \tilde{\boldsymbol{\Sigma}}_{21} \end{bmatrix}' \boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu}). \end{aligned}$$

Now observe that

$$\begin{aligned} \boldsymbol{z}_1 - \hat{\boldsymbol{\theta}} &= \boldsymbol{z}_1 - (\boldsymbol{\Sigma}^*_{11})^{-1} \begin{bmatrix} \tilde{\boldsymbol{\Sigma}}_{11} \\ \tilde{\boldsymbol{\Sigma}}_{21} \end{bmatrix}' \boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu}) \\ &= \boldsymbol{z}_1 - (\boldsymbol{\Sigma}^*_{11})^{-1}[\boldsymbol{\Sigma}^*_{11} \; \boldsymbol{\Sigma}^*_{12}][\boldsymbol{z}_1 - \boldsymbol{\mu}_1 \;\; \boldsymbol{z}_2 - \boldsymbol{\mu}_2] \\ &= \boldsymbol{z}_1 - [I_k \;\; (\boldsymbol{\Sigma}^*_{11})^{-1}\boldsymbol{\Sigma}^*_{12}][\boldsymbol{z}_1 - \boldsymbol{\mu}_1 \;\; \boldsymbol{z}_2 - \boldsymbol{\mu}_2] \\ &= \boldsymbol{z}_1 - (\boldsymbol{z}_1 - \boldsymbol{\mu}_1) - (\boldsymbol{\Sigma}^*_{11})^{-1}\boldsymbol{\Sigma}^*_{12}(z_2 - \boldsymbol{\mu}_2) \\ &= \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{z}_2 - \boldsymbol{\mu}_2) \end{aligned}$$

where the last equality follows from $-(\boldsymbol{\Sigma}^*_{11})^{-1}\boldsymbol{\Sigma}^*_{12} = \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}$ iff $-\boldsymbol{\Sigma}^*_{12}\boldsymbol{\Sigma}_{22} = \boldsymbol{\Sigma}^*_{11}\boldsymbol{\Sigma}_{12}$ iff $[\boldsymbol{\Sigma}^*_{11} \; \boldsymbol{\Sigma}^*_{12}][\boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}]' = 0$ which follows from $\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma} = I$. □

B. Proof of Proposition 3

Proof. We will use the notation $\hat{\boldsymbol{\theta}}_1 = \text{argmin}_{\boldsymbol{\theta}_1} ||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu}) - (\boldsymbol{\Sigma}^{-1/2})_{.1}\boldsymbol{\theta}_1||_2^2$ for the OLS fit, and $\text{RSS}_k = ||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu}) - (\boldsymbol{\Sigma}^{-1/2})_{.1}\hat{\boldsymbol{\theta}}_1||_2^2$.

Part 1.
For $k = d$, we know from (1) that $\text{RSS}_d = 0$. Now let $1 \leqslant k \leqslant d-1$. We want to show that $\text{RSS}_k = (\boldsymbol{z}_2 - \boldsymbol{\mu}_2)' \boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{z}_2 - \boldsymbol{\mu}_2)$. Let $\hat{\boldsymbol{\theta}} := [\hat{\boldsymbol{\theta}}_1' \;\, 0 \, \dots \, 0]'$ be the d-variate vector with coefficients $\hat{\boldsymbol{\theta}}_1$ followed by $d - k$ zeroes.

We now have that

$$\begin{aligned} \text{RSS}_k &= ||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu}) - (\boldsymbol{\Sigma}^{-1/2})_{.1}\hat{\boldsymbol{\theta}}_1||_2^2 \\ &= ||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu}) - \boldsymbol{\Sigma}^{-1/2}\hat{\boldsymbol{\theta}}||_2^2 \\ &= (\boldsymbol{z} - \boldsymbol{\mu} - \hat{\boldsymbol{\theta}})' \boldsymbol{\Sigma}^{-1}(\boldsymbol{z} - \boldsymbol{\mu} - \hat{\boldsymbol{\theta}}) \\ &= [\boldsymbol{z}_1 - \boldsymbol{\mu}_1 - \hat{\boldsymbol{\theta}}_1 \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2]' \boldsymbol{\Sigma}^{-1} [\boldsymbol{z}_1 - \boldsymbol{\mu}_1 - \hat{\boldsymbol{\theta}}_1 \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2] \; . \end{aligned}$$

Following Page 47 of Petersen and Pedersen (2012), we can write $\boldsymbol{\Sigma}^{-1} = \boldsymbol{ABA}'$ with

$$\boldsymbol{A} := \begin{bmatrix} \boldsymbol{I} & \mathbf{0} \\ -\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} & \boldsymbol{I} \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{B} := \begin{bmatrix} \boldsymbol{C}_1^{-1} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{22}^{-1} \end{bmatrix}$$

where $\boldsymbol{C}_1 := \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}$. We now have that

$$\begin{aligned} [\boldsymbol{z}_1 - \boldsymbol{\mu}_1 - \hat{\boldsymbol{\theta}}_1 \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2]'\,\boldsymbol{A} &= [\boldsymbol{z}_1 - \boldsymbol{\mu}_1 - \hat{\boldsymbol{\theta}}_1 - (\boldsymbol{z}_2 - \boldsymbol{\mu}_2)\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2]' \\ &= [\mathbf{0} \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2]' \end{aligned}$$

using the result of Proposition 2. Therefore,

$$\begin{aligned} \mathrm{RSS}_k &= [\boldsymbol{z}_1 - \boldsymbol{\mu}_1 - \hat{\boldsymbol{\theta}}_1 \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2]'\,\boldsymbol{\Sigma}^{-1}\,[\boldsymbol{z}_1 - \boldsymbol{\mu}_1 - \hat{\boldsymbol{\theta}}_1 \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2] \\ &= [\mathbf{0} \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2]'\,\boldsymbol{B}\,[\mathbf{0} \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2] \\ &= (\boldsymbol{z}_2 - \boldsymbol{\mu}_2)'\,\boldsymbol{\Sigma}_{22}^{-1}\,(\boldsymbol{z}_2 - \boldsymbol{\mu}_2) \ . \end{aligned}$$

Part 2.
We will now show that the differences in RSS follow a $\chi^2(1)$ distribution, that is, $\Delta_k := \mathrm{RSS}_{k-1} - \mathrm{RSS}_k \sim \chi^2(1)$ assuming that $\boldsymbol{z} = [\boldsymbol{z}_1' \ \boldsymbol{z}_2']'$ is multivariate Gaussian with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. For $k = 0$, we set by convention $\hat{\boldsymbol{\theta}} := \mathbf{0}$ and $\mathrm{RSS}_0 := ||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z} - \boldsymbol{\mu})||_2^2$.

We show the result for $k = 1$ as the subsequent steps are analogous. The reasoning below is similar to Appendix A.2 of Danilov (2010) where the cells were not yet ranked from most to least outlying. As in Part 1 of the proof, we can write $\boldsymbol{\Sigma}^{-1} = \boldsymbol{ABA}'$ with

$$\boldsymbol{A} = \begin{bmatrix} 1 & \mathbf{0} \\ -\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} & \boldsymbol{I} \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{B} = \begin{bmatrix} C_1^{-1} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{22}^{-1} \end{bmatrix}$$

where this time $C_1 = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}$ is a scalar. We can then write

$$\begin{aligned} \mathrm{RSS}_0 &= (\boldsymbol{z} - \boldsymbol{\mu})'\,\boldsymbol{\Sigma}^{-1}(\boldsymbol{z} - \boldsymbol{\mu}) \\ &= (\boldsymbol{z} - \boldsymbol{\mu})'\,\boldsymbol{ABA}'\,(\boldsymbol{z} - \boldsymbol{\mu}) \\ &= [z_1 - \mu_1 - (\boldsymbol{z}_2 - \boldsymbol{\mu}_2)\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2]'\,\boldsymbol{B}\,[z_1 - \mu_1 - (\boldsymbol{z}_2 - \boldsymbol{\mu}_2)\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} \quad \boldsymbol{z}_2 - \boldsymbol{\mu}_2] \\ &= ((z_1 - \mu_1^*)/\sigma_1^*)^2 + (\boldsymbol{z}_2 - \boldsymbol{\mu}_2)'\,\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{z}_2 - \boldsymbol{\mu}_2) \\ &= ((z_1 - \mu_1^*)/\sigma_1^*)^2 + \mathrm{RSS}_1 \end{aligned}$$

where $\mu_1^* := \mu_1 + (\boldsymbol{z}_2 - \boldsymbol{\mu}_2)\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}$ and $\sigma_1^* := \sqrt{C_1}$. So we obtain

$$\Delta_1 = \mathrm{RSS}_0 - \mathrm{RSS}_1 = ((z_1 - \mu_1^*)/\sigma_1^*)^2$$

and this is the square of a standard Gaussian variable since $z_1 - \mu_1^*$ is Gaussian with expectation 0 and standard deviation σ_1^* . We thus have $\Delta_1 \sim \chi^2(1)$. □

C. Implementation of the cellHandler Algorithm

The LAR component of cellHandler is a regression of $\tilde{\boldsymbol{Y}}$ on $\dot{\boldsymbol{X}}$ as defined in the paper. Since this regression has no intercept and we need to preserve the column scaling in $\dot{\boldsymbol{X}}$, we run the function *lars::lar* with the options *intercept=F* and *normalize=F*.

For the imputations in Proposition 2 and the RSS in Proposition 3, we require the OLS fits $\hat{\boldsymbol{\theta}}_A$ minimizing $||\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{z}-\boldsymbol{\mu}) - (\boldsymbol{\Sigma}^{-1/2})_A\,\boldsymbol{\theta}_1||_2^2$ where A is the set of active predictor variables in every step of LAR. Fortunately, these can be obtained without significant additional computation time because each step of LAR already carries out the QR decomposition of $(\dot{\boldsymbol{X}}_A)'\,\dot{\boldsymbol{X}}_A$ where $\dot{\boldsymbol{X}}_A$ is the submatrix of $\dot{\boldsymbol{X}}$ consisting of the columns in A. The resulting OLS regression vectors $\hat{\boldsymbol{\beta}}_A$ obtained by LAR (which contain zeroes for the inactive variables) are then easily rescaled to $\hat{\boldsymbol{\theta}}_A = \boldsymbol{W}^{-1}\hat{\boldsymbol{\beta}}_A$.

D. Description of the Initial Estimator DDCW

The Detection-Imputation (DI) method of Section 3.2 needs initial cellwise robust estimates $\hat{\boldsymbol{\mu}}^0$ and $\widehat{\boldsymbol{\Sigma}}^0$ of location and covariance. One option is to insert the 2SGS estimator of Leung et al. (2017). We also developed a different initial estimator called DDCW, which we describe here. Its steps are:

1. Drop variables with too many missing values or zero median absolute deviation, and continue with the remaining columns.

2. Run the DetectDeviatingCells (DDC) method (Rousseeuw and Van den Bossche 2018) with the constraint that no more than $n\,maxCol$ cells are flagged in any variable. DDC also rescales the variables, and may delete some cases. Continue with the remaining imputed and rescaled cases denoted as $\boldsymbol{z}_i$.

3. Project the $\boldsymbol{z}_i$ on the axes of their principal components, yielding the transformed data points $\tilde{\boldsymbol{z}}_i$.

4. Compute the wrapped location $\hat{\boldsymbol{\mu}}_w$ and covariance matrix $\widehat{\boldsymbol{\Sigma}}_w$ (Raymaekers and Rousseeuw 2019) of these $\tilde{\boldsymbol{z}}_i$. Next, compute the temporary points $\boldsymbol{u}_i = (u_{i1}, ..., u_{id})$ given by $u_{ij} = \max\{\min\{\tilde{z}_{ij} - (\hat{\boldsymbol{\mu}}_w)_j, 2\}, -2\}$. Then remove all cases for which the squared robust distance $\mathrm{RD}^2(i) = \boldsymbol{u}_i'\widehat{\boldsymbol{\Sigma}}_w^{-1}\boldsymbol{u}_i$ exceeds $\chi^2_{d,q}\,\mathrm{median}_h(\mathrm{RD}^2(h))/\chi^2_{d,0.5}$.

5. Project the remaining $\tilde{\boldsymbol{z}}_i$ on the eigenvectors of $\widehat{\boldsymbol{\Sigma}}_w$ and again compute a wrapped location and covariance matrix.

6. Transform these estimates back to the original coordinate system of the imputed data, and undo the scaling. This yields the estimates $\hat{\boldsymbol{\mu}}^0$ and $\widehat{\boldsymbol{\Sigma}}^0$.

Note that DDCW can handle missing values since the DDC method in Step 2 imputes them. The reason for the truncation in the rejection rule in Step 4 is that otherwise the robust distance RD could be inflated by a single outlying cell. Step 4 tends to remove rows which deviate strongly from the covariance structure. These are typically rows which cannot be shifted towards the majority of the data without changing a large number of cells.

E. Step by Step Description of the DI Algorithm

We now give a step-by-step description of the DI algorithm, with some additional details.

1. Standardize the columns (variables) as described in the beginning of Section 2.1.

2. Compute initial estimates $\hat{\boldsymbol{\mu}}^0$ and $\widehat{\boldsymbol{\Sigma}}^0$. The algorithm currently has two options for this:

   - the DDCW estimator described in Section D above;
   - the 2SGS estimator of Leung et al. (2017), available in the R package GSE (Leung et al. 2019).

3. **D-step**. Given the estimates $\hat{\boldsymbol{\mu}}^{t-1}$ and $\widehat{\boldsymbol{\Sigma}}^{t-1}$ where $t = 1, 2, \ldots$, we flag outlying cells across all rows of the dataset. This is done as described in Section 3.2 by applying the cellHandler method of Section 2 to each row $\boldsymbol{z}_i'$ using $\hat{\boldsymbol{\mu}}^{t-1}$ and $\widehat{\boldsymbol{\Sigma}}^{t-1}$. The D-step imposes a maximum on the number of flagged cells in a row, namely $n\, maxCol$ where $maxCol$ is set to 25% by default. Since all missing values (NA's) are automatically flagged, the algorithm would not be able to run if there are too many NA's in a column. In practice, the algorithm starts by setting variables with too many NA's aside and giving a message about this. The D-step yields a list of flagged cells in each row, which contains the flagged outlying cells as well as their imputed values.

4. **I-step**. We re-estimate the center as $\hat{\boldsymbol{\mu}}^t$ which is the mean of the dataset with its imputed cells. For computing $\widehat{\boldsymbol{\Sigma}}^t$ we use the formula of the M-step in the EM-algorithm. It does not simply compute the covariance matrix of the imputed data, which would underestimate the true variability. Instead, the EM method adds a bias correction. This bias correction depends on which cells were imputed, and can therefore be different for every row of the data. Suppose the first row $\boldsymbol{z}_1$ has an imputed part $\boldsymbol{z}_{1i}$ and an untouched part $\boldsymbol{z}_{1u}$, then the bias correction matrix from that row is

   $$B_{ii} = \frac{1}{n}\widehat{\boldsymbol{\Sigma}}_{ii}^{t-1} - \frac{1}{n}\widehat{\boldsymbol{\Sigma}}_{iu}^{t-1}(\boldsymbol{\Sigma}_{uu}^{t-1})^{-1}\boldsymbol{\Sigma}_{ui}^{t-1} \ .$$

   This correction term is known to remove the bias when the data is uncontaminated multivariate Gaussian with missing values generated completely at random (MCAR), that is, independent of both the observed cells as well as the values the missing cells had before they became unavailable. In our simulations with contaminated data, this bias correction also turned out to improve the results.

5. Iterate steps 3 and 4 alternatingly until

   $$||\hat{\boldsymbol{\mu}}^t - \hat{\boldsymbol{\mu}}^{t-1}||_2^2 + ||\widehat{\boldsymbol{\Sigma}}^t - \widehat{\boldsymbol{\Sigma}}^{t-1}||_2^2$$

   is below a given tolerance, where the second norm is given by (4).

6. Apply cellHandler with the converged $\hat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\Sigma}}$ to obtain the final list of cellwise outliers and their imputed values.

7. Unstandardize the results using the univariate location and scale estimates of the original data columns, used in Step 1.

F. Proof of Proposition 4

Proof. Starting from the well-known formula for $\text{KL}(\boldsymbol{X},\boldsymbol{Y})$ we obtain

$$\begin{aligned}
\text{KL}(\boldsymbol{X},\boldsymbol{Y}) &= \text{tr}(\boldsymbol{A}\,\boldsymbol{B}^{-1}) - d - \log\det(\boldsymbol{A}\,\boldsymbol{B}^{-1}) \\
&= \text{tr}(\boldsymbol{A}\,\boldsymbol{B}^{-1/2}\boldsymbol{B}^{-1/2}) - d - \log\det(\boldsymbol{A}\,\boldsymbol{B}^{-1/2}\boldsymbol{B}^{-1/2}) \\
&= \text{tr}(\boldsymbol{B}^{-1/2}\boldsymbol{A}\,\boldsymbol{B}^{-1/2}) - d - \log\det(\boldsymbol{B}^{-1/2}\boldsymbol{A}\,\boldsymbol{B}^{-1/2}) \\
&= \left(\sum_{j=1}^{d}\eta_j\right) - d - \log\left(\prod_{j=1}^{d}\eta_j\right) \\
&= \sum_{j=1}^{d}(\eta_j - 1 - \log(\eta_j)) = D(\boldsymbol{A},\boldsymbol{B})
\end{aligned}$$

where the fourth equality used the fact that $\boldsymbol{B}^{-1/2}\boldsymbol{A}\,\boldsymbol{B}^{-1/2}$ is PSD so it can be diagonalized; hence, its trace is the sum of its eigenvalues. □

G. F-scores in Dimensions 10, 20 and 40

Figure 3 showed the precision, recall, and F-score for data generated by the contaminated ALYZ model and the contaminated A09 model, for $n = 400$ points in $d = 20$ dimensions. Here Figure 8 shows the F-scores for both DDCW.DI (DI starting from DDCW) and 2SGS.DI (DI starting from 2SGS). These are byproducts of the simulations in Figure 4 for $(n,d) = (100,10)$ and Figure 5 for $(n,d) = (400,20)$ and $(800,40)$. Note that Step 6 of the DI algorithm in Appendix E provides the flagged cells, so the DDCW.DI curves for $d = 20$ in Figure 8 correspond to those of cellHandler in the lower panel of Figure 3.

H. Computation Times of DI

Table 1 shows computation times of the DI algorithm as implemented in the R package `cellWise` on CRAN. This implementation contains compiled C++ code. The times are the averaged runtimes of one replication of the simulations in Figure 4 for $(n,d) = (100,10)$ and Figure 5 for $(n,d) = (400,20)$ and $(800,40)$. The times are in seconds on a laptop with Intel Core i7-5600U at 2.60Hz. We see that DDCW.DI requires less time than 2SGS.DI because the DDCW initial estimator is quite fast. Note that the computation time of the D-step in DI could be reduced by executing the LARS computations on the rows in parallel.

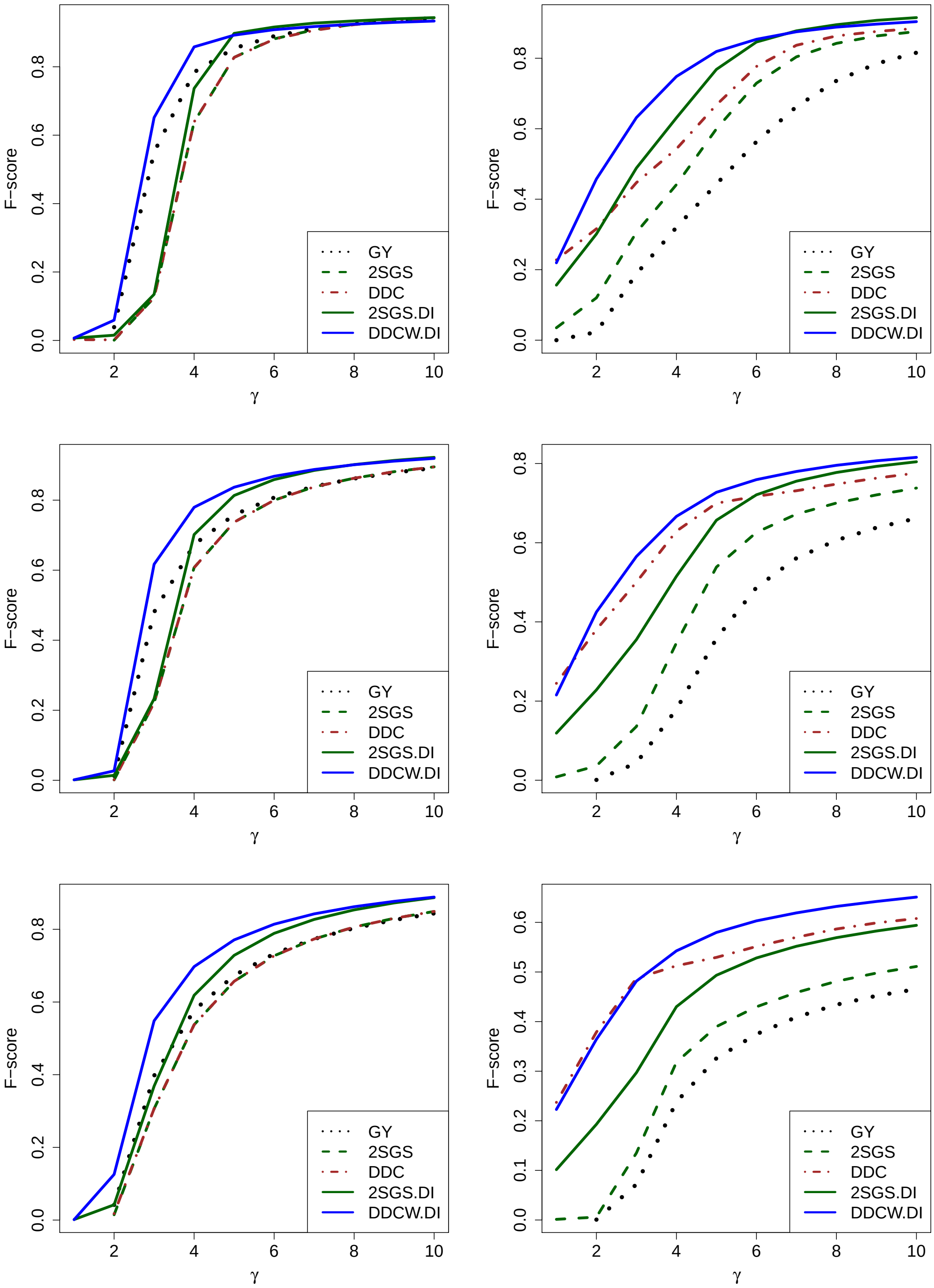

Figure 8: F-scores of flagged cellwise outliers on data generated by the contaminated ALYZ model (left) and the contaminated A09 model (right), for $(n, d) = (100, 10)$ (top), $(n, d) = (400, 20)$ (middle), and $(n, d) = (800, 40)$ (bottom).

Table 1: Computation time (in seconds) of one replication of DI in the simulation.

n and d	2SGS.DI	DDCW.DI
$n = 100$, $d = 10$	1.70	0.54
$n = 400$, $d = 20$	16.42	5.38
$n = 800$, $d = 40$	110.25	35.16

I. Simulations with Cellwise and Casewise Outliers

We now run a simulation study in which the data are contaminated by 10% of cellwise outliers generated as in the paper, plus 10% of rowwise outliers. In this particular setting, "rowwise outliers" refer to rows in which all cells are contaminated in the same way as before, that is, rows with d cellwise outliers. We generate these outlying rows by the formula $\boldsymbol{v} = \gamma d\sqrt{d}\,\boldsymbol{u}'/\,\mathrm{MD}(\boldsymbol{u},\boldsymbol{\mu},\boldsymbol{\Sigma})$ where $\boldsymbol{u}$ is the eigenvector of $\boldsymbol{\Sigma}$ with the smallest eigenvalue. This corresponds to the cellwise formula of Subsection 2.3 in which the indices of the outlying cells $K = \{j(1),\ldots,j(k)\}$ are replaced by $K = \{1,\ldots,d\}$. Next, we replace 10% of the rows by $\boldsymbol{v}$, and afterward sample the positions of the cellwise outliers from the remaining 90% of the rows. The results are shown in Figures 9 and 10. They look qualitatively similar to those in Figures 4 and 5 in the paper.

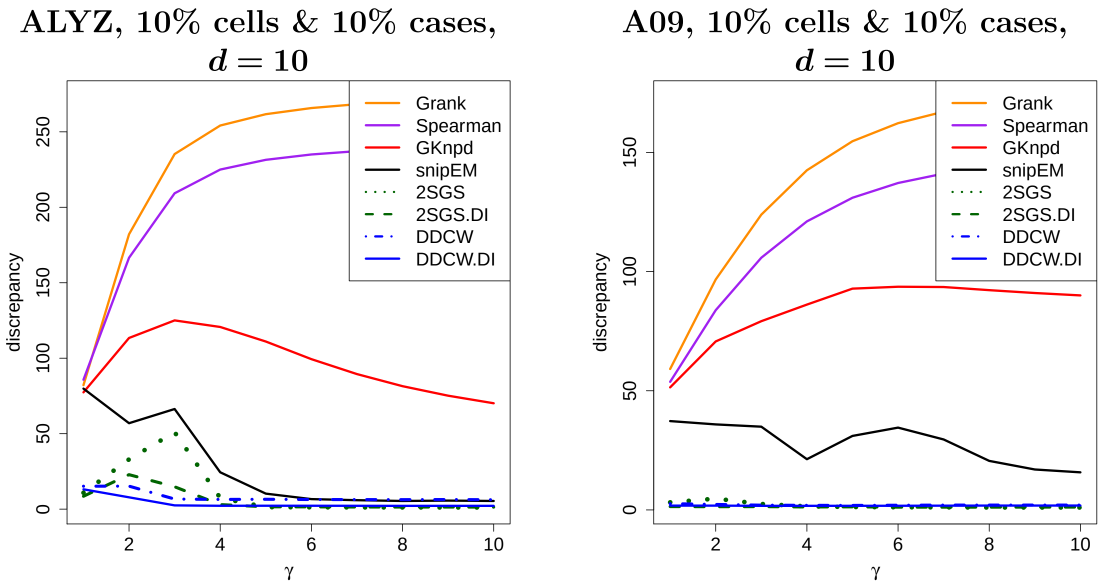

Figure 9: Discrepancy $D(\widehat{\boldsymbol{\Sigma}},\boldsymbol{\Sigma})$ of estimated covariance matrices for $d = 10$, $n = 100$.

Note that in this section we do not plot F-scores for rowwise outliers, because the cellHandler and DI algorithms do not have a mechanism for detecting rowwise outliers. They are methods for cellwise outliers. If we were to try to detect both types of outliers simultaneously, we would run into an identifiability issue. For instance, one can easily generate rows which would be considered outliers under the rowwise paradigm, but only contain a single cellwise outlier under the cellwise paradigm. And starting from the cellwise paradigm, how many cellwise outliers in a row would it take before the entire row should be considered outlying? The identifiability issue is especially complicated in the current simulations because we generate cellwise outliers in a structured way as explained in Section 2.3, so that they do not stand out individually.

Because of these concerns, when generating both types of outliers in this section, our focus is on the accuracy of the covariance matrix estimate, so that it is clear what the goal is. Afterward, the user can choose whether to employ the estimated covariance matrix to detect rowwise outliers based on their Mahalanobis-type distance, or as an input to the cellHandler algorithm to detect cellwise outliers.

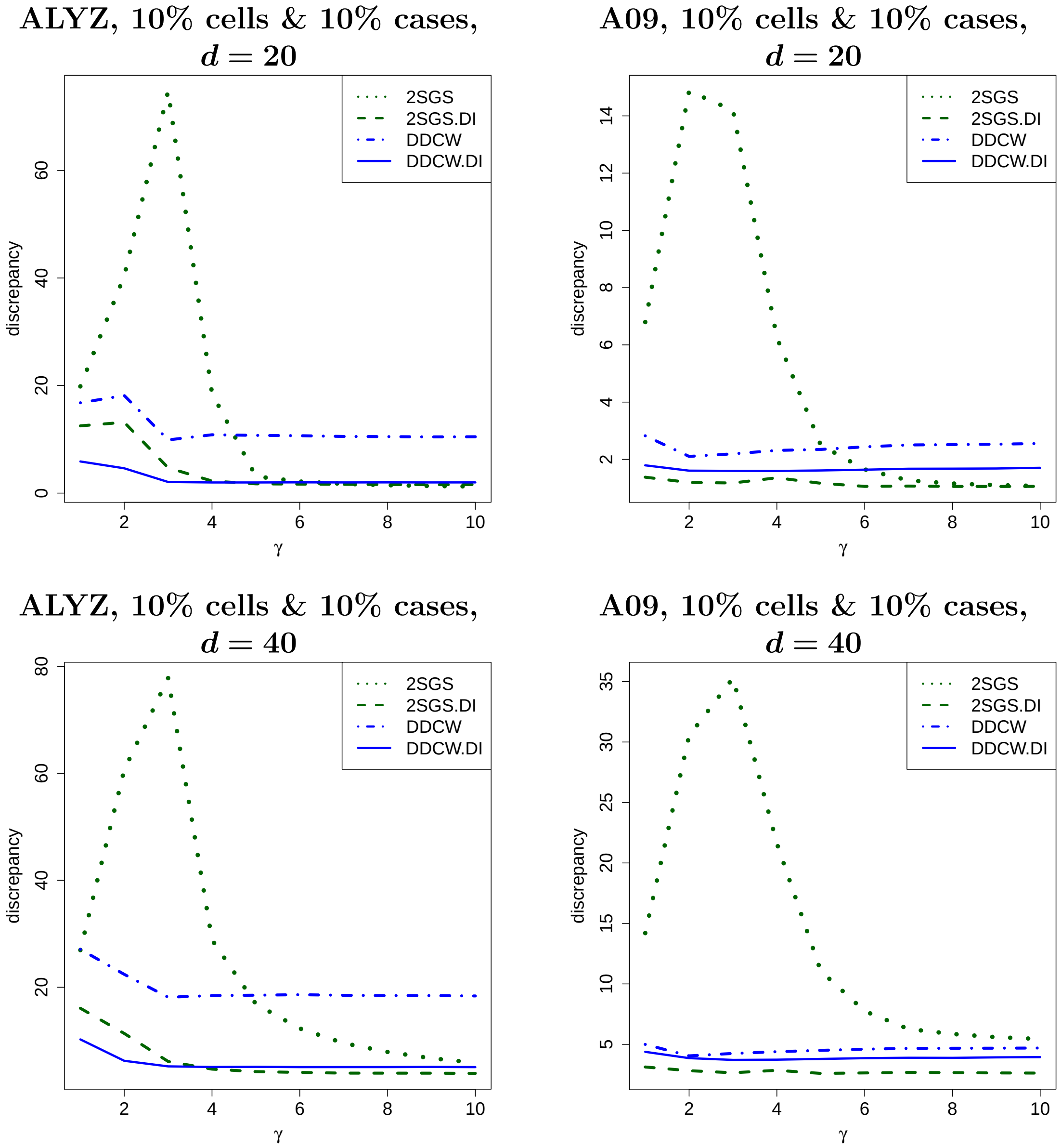

Figure 10: Discrepancy $D(\widehat{\mathbf{\Sigma}}, \mathbf{\Sigma})$ given by (5) of estimated covariance matrices for $d = 20$ and $n = 400$ (top panels) and for $d = 40$ and $n = 800$ (bottom panels).

J. List of Volatile Organic Compounds

The volatile organic compounds (VOC's) analyzed in Section 6 are listed below.

Variable Name	VOC name
URX2MH	2-Methylhippuric acid
URX34M	3- and 4-Methylhippuric acid
URXAAM	N-Acetyl-S-(2-carbamoylethyl)-L-cysteine
URXAMC	N-Acetyl-S-(N-methylcarbamoyl)-L-cysteine
URXATC	2-Aminothiazoline-4-carboxylic acid
URXBMA	N-Acetyl-S-(benzyl)-L-cysteine
URXCEM	N-Acetyl-S-(2-carboxyethyl)-L-cysteine
URXCYM	N-Acetyl-S-(2-cyanoethyl)-L-cysteine
URXDHB	N-Acetyl-S-(3,4-dihydroxybutyl)-L-cysteine
URXHP2	N-Acetyl-S-(2-hydroxypropyl)-L-cysteine
URXHPM	N-Acetyl-S-(3-hydroxypropyl)-L-cysteine
URXIPM3	N-Acetyl- S- (4- hydroxy- 2- methyl- 2- butenyl)-L-cysteine
URXMAD	Mandelic acid
URXMB3	N-Acetyl-S-(4-hydroxy-2-butenyl)-L-cysteine
URXPHG	Phenylglyoxylic acid
URXPMM	N-Acetyl-S-(3-hydroxypropyl-1-methyl)-L-cysteine

K. Analysis of the VOC Data after Preprocessing

As kindly pointed out by a referee, the concentrations of compounds in urine samples can depend on the dilution of the urine, if the analytical chemistry technique did not adjust for this. Therefore, the measurements might not always be comparable across the different samples. This issue can be dealt with by applying the centered log ratio (CLR) transform, which transforms the original data to log-ratios between the variables. We reran the analysis of the VOC data after applying the CLR transform using the `cenLR` function in the `R`-package `robCompositions` (Templ et al. 2020).

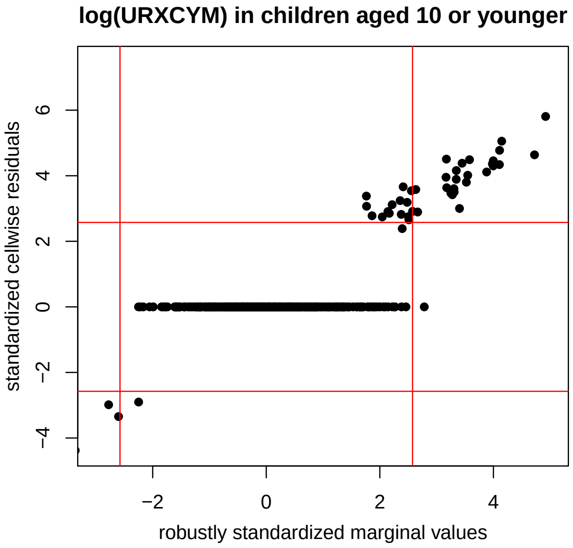

Figure 11: Plot of standardized cell residuals of CLR-transformed URXCYM obtained by cellHandler, versus the robustly standardized values of CLR-transformed URXCYM on its own.

Figures 11 and 12 present the results of this analysis. In the first plot, we see that the univariate detection rule flags a few more outliers after the CLR transform than without it. However, the red curve in the second figure still shows no effect of adult smokers in the household on URXCYM in children. Therefore, the conclusion that univariate outlier detection is insufficient here remains valid.

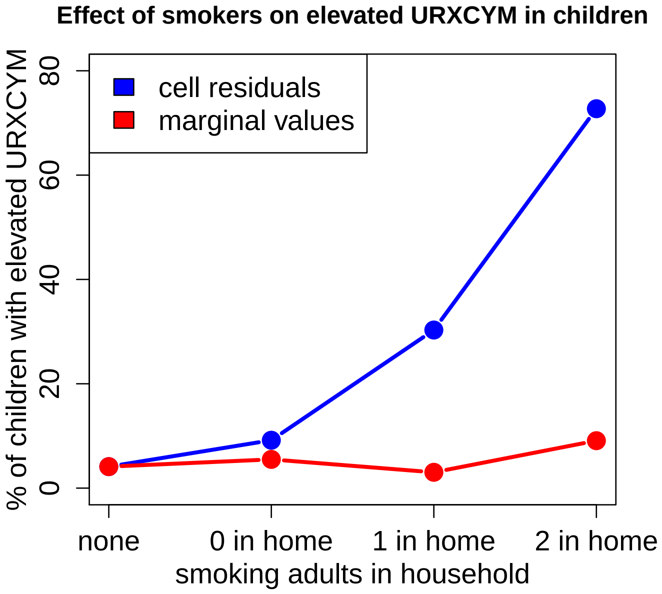

Figure 12: The blue curve shows the percentage of elevated URXCYM cell residuals in function of the smoking status of adult family members. The red curve shows the percentage of elevated marginal URXCYM values. Here the URXCYM values were preprocessed with the CLR transformation.

Affiliation:

Jakob Raymaekers, Peter J. Rousseeuw
Section of Statistics and Data Science
Mathematics Department, KU Leuven
Celestijnenlaan 200B
BE-3001 Heverlee, Belgium
E-mail: peter@rousseeuw.net
URL: https://wis.kuleuven.be/statdatascience/robust

Journal of Data Science, Statistics, and Visualisation https://jdssv.org/
published by the International Association for Statistical Computing
http://iasc-isi.org/

July 2021, Volume I, Issue III *Submitted:* 2020-07-21
doi:10.52933/jdssv.v1i3.18 *Accepted:* 2021-01-12