ICSOutlier: Unsupervised Outlier Detection for Low-Dimensional Contamination Structure

by Aurore Archimbaud, Klaus Nordhausen, and Anne Ruiz-Gazen

Abstract Detecting outliers in a multivariate and unsupervised context is an important and ongoing problem notably for quality control. Many statistical methods are already implemented in R and are briefly surveyed in the present paper. But only a few lead to the accurate identification of potential outliers in the case of a small level of contamination. In this particular context, the Invariant Coordinate Selection (ICS) method shows remarkable properties for identifying outliers that lie on a low-dimensional subspace in its first invariant components. It is implemented in the **ICSOutlier** package. The main function of the package, `ics.outlier`, offers the possibility of labelling potential outliers in a completely automated way. Four examples, including two real examples in quality control, illustrate the use of the function. Comparing with several other approaches, it appears that ICS is generally as efficient as its competitors and shows an advantage in the context of a small proportion of outliers lying in a low-dimensional subspace. In quality control, the method may help in properly identifying some defective products while not detecting too many false positives.

Introduction

The unsupervised detection of multivariate outliers is a timeless subject of interest in statistics, see Aggarwal (2017), Hodge and Austin (2004), Hadi et al. (2009) for a complete overview. Indeed, this can be the goal of the analysis, like in fraud detection, in medical applications or manufacturing-defect detection. It can also be used for preprocessing in almost any statistical analysis that is sensitive to the presence of outliers, e.g. Principal Component Analysis (PCA). Many statistical methods exist and have already been implemented in R through tens of packages. Only some of these packages are dedicated to the unsupervised context and are mentioned below. One of the most common methods for multivariate outlier detection is the Mahalanobis Distance (MD), and many packages are based on this distance: **mvoutlier** (Filzmoser and Gschwandtner, 2017), **CerioliOutlierDetection** (Green and Martin, 2017a), **rrcovHD** (Todorov, 2016), **faoutlier** (Chalmers and Flora, 2015). Additionally, traditional methods such as the angle-based methods are implemented in the packages **abodOutlier** (Jimenez, 2015), **HighDimOut** (Fan, 2015), the distribution-based methods in **alphaOutlier** (Rehage and Kuhnt, 2016), **extremevalues** (van der Loo, 2010), **HDoutliers** (Fraley, 2016), **outliers** (Komsta, 2011), the density-based methods in **DMwR2** (Torgo, 2016), **HighDimOut**, **ldbod** (Williams, 2017), **Rlof** (Hu et al., 2015) and the depth-based methods in **depth** (Genest et al., 2017). Finally, other approaches have been implemented, such as the one based on Projection Pursuit (PP) in **REPPlab** (Fischer et al., 2016b), and PCA in **OutlierDC** (Eo and Cho, 2014), **pcadapt** (Luu and Blum, 2017), **rrcov** (Todorov and Filzmoser, 2009) and **rrcovHD**. However, it is important to note that all the implemented functions do not necessary return outlierness measures and the identities of the outlying observations. Limiting only to some packages fulfilling these conditions and to multivariate methods for numerical variables leads to focusing on **mvoutlier**, **CerioliOutlierDetection** and **rrcov**. These packages implement modified versions of the robust Mahalanobis distance, different algorithms for the outlier identification in high dimensions (Filzmoser and Todorov, 2013), classical and robust PCA. In PCA, the labelling of the outliers is based on the comparison of two outlierness measures: a score distance (SD) based on the first principal components and a distance in the orthogonal space (OD). All these methods rely on distances. To extend the analysis, we include comparisons with Local Outlier Factor (LOF) (**Rlof** package) and Angle-Based Outlier Factor (ABOD) (**abodOutlier** package) approaches even if they only return outlierness measures and not the identities of the outlying observations.

These methods dedicated to outlier detection in an unsupervised context have different properties. Contrary to PCA, the Mahalanobis distance and its modified versions are invariant under any affine transformation as long as the location and scatter estimators are affine equivariant. However, several drawbacks have been noticed in the use of these distances. First, Filzmoser et al. (2005) highlight the fact that an outlier is not necessarily an extreme value. So, using a fixed threshold to label outliers for every data set is not the wisest solution and the threshold should be adjusted to the sample size. Moroever, as Cerioli (2010) notes, the threshold used usually is not adapted to the case in which there is no outlier in the data. Finally, for Cerioli (2010), the comparison of the distances of each observation to the threshold should not be performed independently without taking into account any simultaneous adjustments. Otherwise, the method may lead to many false positives, i.e non outlying observations that are identified as outliers. Note that this is a serious concern that applies to almost all the outlier-detection methods. To the best of our knowledge, the **mvoutlier** and **CerioliOutlierDetection** packages are the first ones that tend to correct these drawbacks, but only for the robust Mahalanobis distance with the Minimum Covariance Determinant (MCD) estimators.

The **ICSOutlier** (Archimbaud et al., 2016) package is another approach to the detection of outliers. It appears that ICS is as efficient as its competitors in many situations and shows an advantage in the case of a small proportion of outliers lying in a low-dimensional subspace. This method is well-designed in particular for quality control. Its current version is able to handle a small proportion of outliers that lie on a subspace, using the affine equivariant Invariant Coordinate Selection (ICS) method, as presented by Archimbaud et al. (2018). The method relies on a generalized diagonalization and on the selection of the invariant components associated with the largest eigenvalues. Note that when more than say 10% of the observations are suspected to be outlying, invariant components associated with the smallest eigenvalues may also be of interest and this alternative will be considered in a future version of the package. ICS fulfills all the properties mentioned previously: (i) detect the absence of outliers, (ii) not be sensitive to the standardization of the data and (iii) be a multivariate method which controls the number of observations identified as outliers. The restriction to a small proportion of outliers that belong to a subspace is of interest in some areas of manufacturing products with a high level of quality control (e.g. automotive, avionics or aerospace).

In the following sections, the principle of the Invariant Coordinate Selection (ICS) method is recalled as well as the three steps of the outlier detection procedure: (i) invariant components selection, (ii) outlierness measure definition and (iii) outlier identification. Then, we explain how to use the **ICSOutlier** package and finally we analyze the efficiency of the ICS method compared to other methods on four examples, including a new real data set included in the package.

Invariant Coordinate Selection (ICS) for outlier detection

Principle

The ICS method is a powerful method designed for exploring multivariate data by revealing their structure (Nordhausen et al., 2008). As explained in Tyler et al. (2009), it is based on a simultaneous spectral decomposition of two scatter matrices and leads to an affine invariant coordinate system. Readers not so familiar with the concept of scatter matrices are refered for example to Rousseeuw and Hubert (2013), Nordhausen and Tyler (2015) and references therein for definitions, examples and properties.

Following Archimbaud et al. (2018), for a p-variate dataset $\mathbf{X}_n = (\mathbf{x}_1, \ldots, \mathbf{x}_n)'$, let $\mathbf{V}_1(\mathbf{X}_n)$ and $\mathbf{V}_2(\mathbf{X}_n)$ be two affine equivariant scatter matrices (symmetric and positive definite) and $\mathbf{m}_1(\mathbf{X}_n)$ and $\mathbf{m}_2(\mathbf{X}_n)$ their associated location estimators. ICS is looking for the diagonal $p \times p$ matrix $\mathbf{D}(\mathbf{X}_n)$ containing the eigenvalues of $\mathbf{V}_1(\mathbf{X}_n)^{-1}\mathbf{V}_2(\mathbf{X}_n)$ in decreasing order and the $p \times p$ matrix $\mathbf{B}(\mathbf{X}_n) = (\mathbf{b}_1, \ldots, \mathbf{b}_p)'$ containing the corresponding eigenvectors as its rows and such that

$$\mathbf{B}(\mathbf{X}_n)\mathbf{V}_1(\mathbf{X}_n)\mathbf{B}'(\mathbf{X}_n) = \mathbf{I}_p.$$

We have:

$$\mathbf{V}_1(\mathbf{X}_n)^{-1}\mathbf{V}_2(\mathbf{X}_n)\mathbf{B}(\mathbf{X}_n)' = \mathbf{B}(\mathbf{X}_n)'\mathbf{D}(\mathbf{X}_n).$$

Letting $\mathbf{V}_1(\mathbf{X}_n)$ be "more robust" than $\mathbf{V}_2(\mathbf{X}_n)$, the interpretation of these eigenvalues is the main point of the ICS method: they correspond to some kurtosis measure that depends on the choice of $\mathbf{V}_1(\mathbf{X}_n)$ and $\mathbf{V}_2(\mathbf{X}_n)$. For example if $\mathbf{V}_1$ is the usual empirical variance-covariance matrix and $\mathbf{V}_2$ is the scatter matrix based on the fourth moments (the default in the `ics.outlier` function), the eigenvalues are ordered according to their classical Pearson kurtosis values in decreasing order. In this case and for a small proportion of outliers, it is advisable to analyze the projections that maximize the kurtosis and are associated with the largest eigenvalues (Archimbaud et al., 2018).

Then, using the location estimator $\mathbf{m}_1(\mathbf{X}_n)$ associated with the scatter matrix $\mathbf{V}_1(\mathbf{X}_n)$, the corresponding centered scores are obtained as $\mathbf{z}_i = \mathbf{B}(\mathbf{X}_n)(\mathbf{x}_i - \mathbf{m}_1(\mathbf{X}_n))$ for $i = 1, \ldots, n$, so that:

$$\mathbf{Z}_n = (\mathbf{z}_1, \ldots, \mathbf{z}_n)' = (\mathbf{X}_n - \mathbf{1}_n\mathbf{m}_1'(\mathbf{X}_n))\mathbf{B}'(\mathbf{X}_n).$$

They are called the invariant components (IC) because of their affine invariance property in the sense that if $\mathbf{X}_n^* = \mathbf{X}_n\mathbf{A} + \mathbf{1}_n\mathbf{b}'$ for any $p \times p$ regular matrix $\mathbf{A}$ and any p-vector $\mathbf{b}$,

$$(\mathbf{X}_n^* - \mathbf{1}_n\mathbf{m}_1(\mathbf{X}_n^*)')\mathbf{B}(\mathbf{X}_n^*)' = (\mathbf{X}_n - \mathbf{1}_n\mathbf{m}_1(\mathbf{X}_n)')\mathbf{B}(\mathbf{X}_n)'\mathbf{J},$$

where $\mathbf{1}_n$ is a n-vector of ones and $\mathbf{J}$ is a $p \times p$ diagonal matrix with diagonal elements ± 1, which means the invariant coordinates change at most their signs.

These scores are related to the Mahalanobis Distance (MD). For any observation $\mathbf{x}_i$ with $i = 1, \ldots, n$, the squared Euclidian norm of its centered invariant coordinates is exactly the squared Mahalanobis distance from $\mathbf{m}_1(\mathbf{X}_n)$ in the sense of $\mathbf{V}_1(\mathbf{X}_n)$:

$$\mathbf{z}_i'\mathbf{z}_i = (\mathbf{x}_i - \mathbf{m}_1(\mathbf{X}_n))'\mathbf{V}_1(\mathbf{X}_n)^{-1}(\mathbf{x}_i - \mathbf{m}_1(\mathbf{X}_n)).$$

The added value of using ICS over MD is realized when the structure of the data is on a subspace of dimension $q < p$. In this context and with a small percentage of outliers, it is of interest to focus only on the q projections that maximize a kurtosis measure. Indeed, if the number k of selected Invariant Coordinates corresponds to the dimension q of the subspace on which the outliers lie, then the ICS method is expected to recover the subspace of interest for outlier detection. As detailed in Tyler et al. (2009), this subspace corresponds to Fisher's discriminant subspace in case of mixtures of Gaussian distributions. Nevertheless, the main difficulty is to correctly estimate this dimension k.

Invariant coordinates selection

Depending on the combination of the scatters $\mathbf{V}_1(\mathbf{X}_n)$ and $\mathbf{V}_2(\mathbf{X}_n)$ chosen, the **ICSOutlier** package incorporates automated ways to select these k invariant coordinates. The two approaches proposed in Archimbaud et al. (2018) are implemented here: a test based on a quasi-inferential procedure and some normality tests. Considering $\mathbf{V}_1(\mathbf{X}_n)$ be more robust than $\mathbf{V}_2(\mathbf{X}_n)$ and a small percentage of outliers (less than 10% is recommended), the structure of the outlierness should be contained in the first k non-normal components. Note that the structure may be also contained in the last components (associated with the smallest eigenvalues) if the proportion of outliers is large. What is meant by "large" depends on the scatter pair but also on the distribution of the data as detailed in Archimbaud et al. (2018). Since the invariant coordinates are already ordered decreasingly according to a kurtosis measure, it is enough to test whether each component is Gaussian beginning by the one associated with the largest eigenvalue and stop the test procedure as soon as we find a Gaussian component. So, if $k + 1$ denotes the rank of the first Gaussian component, the components are sequentially tested at the adapted level $\alpha_j = \alpha/j$, for $j = 1, \ldots, k$, as in Dray (2008). Because of the sequentiality, the Bonferroni correction adjusts the significance of each test and ensures a nominal level α.

The first approach is a Parallel Analysis (PA) based on Monte Carlo simulations of eigenvalues for Gaussian populations of the same dimension as the initial dataset. This method is common for selecting components in PCA as described in Peres-Neto et al. (2005). It is based on $mEig$ simulations of a Gaussian population and, for each $j = 1, \ldots, p$, the computation of the $1 - \alpha_j$ percentile of the jth eigenvalue of ICS which is considered as a cut-off for the jth component. Then, sequentially from $j = 1$, if the observed jth eigenvalue exceeds the cut-off then the jth invariant component is declared non-Gaussian and is selected. As soon as one eigenvalue is smaller than its associated cut-off, the invariant component is considered as Gaussian and the test procedure is stopped.

The second approach is directly based on usual normality tests and is applied to the invariant coordinates. In the package the user can choose out of the following five tests: the D'Agostino test of skewness (DA), the Anscombe-Glynn (AG) test of kurtosis, the Bonett-Seier (BS) test of Geary's kurtosis, the Jarque-Bera (JB) test for normality which is based on both skewness and kurtosis measures and the Shapiro-Wilk (SW) normality test (see Yazici and Yolacan (2007) and Bonett and Seier (2002) for a complete description of each). The process of testing is still sequential with an adaptation of the level of each test. Once the first normal coordinate is found, the test procedure stops and only the non-Gaussian coordinates are selected.

Measure of outlierness

Let k denote the number of invariant components selected by one of the two test procedures detailed previously or by looking at the screeplot of the eigenvalues. Then, for each observation $\mathbf{x}_i$, $i = 1, \ldots, n$, its squared ICS distance is computed based on the k selected invariant coordinates by:

$$ICSD^2_{\mathbf{V}_1(\mathbf{X}_n)^{-1}\mathbf{V}_2(\mathbf{X}_n)}(\mathbf{x}_i, k) = \mathbf{z}_{i,k}'\mathbf{z}_{i,k}.$$

where $\mathbf{z}_{i,k} = \mathbf{B}(\mathbf{X}_n, k)(\mathbf{x}_i - \mathbf{m}_1(\mathbf{X}_n))$, for $i = 1, \ldots, n$, and the $k \times p$ matrix $\mathbf{B}(\mathbf{X}_n, k)$ contains the k first rows of $\mathbf{B}(\mathbf{X}_n)$.

Outlier identification

Finally, the outliers are identified based on the comparison of their ICS distance with the expectation under the Gaussian distribution as in Archimbaud et al. (2018). The cut-off is derived from *mDist* Monte Carlo simulations of a Gaussian population of the same dimension as the initial dataset. For each observation, the ICS distance is computed using the k selected components. For a given level β, the cut-off corresponds to the average of the $1 - \beta$ percentiles of the distances over all the simulations. An observation is labeled as outlier if its ICS distance is higher than this cut-off. By default, $\beta = 5\%$.

Using the package ICSOutlier

The main function available in **ICSOutlier** is `ics.outlier` which directly calls all the other functions included in the package. First it is necessary to apply the `ics2` function from the **ICS** (Nordhausen et al., 2008) package in order to create an object of class `ics2`. The **ICS** package has been available for a few years, and contains a function called `ics` for implementing the ICS method. However, the `ics` function cannot be used directly for outlier detection, and a new function, `ics2`, had to be added to the **ICS** package (version 1.3-0). The arguments of `ics2` are mostly the same as those of the function `ics`, except for now it is necessary to add the location vectors associated with the scatter estimators. The main function `ics.outlier` of the package **ICSOutlier** uses the output of the `ics2` function as an input. The function `ics.outlier` implements the three steps necessary for outlier detection using ICS in an automated way:

(i) Select the invariant coordinates which recover at best the subspace where the outliers are lying using some test procedure.

(ii) Compute the ICS distance based on the selected invariant coordinates as an outlierness measure for each observation.

(iii) Label the outliers using the cut-off value obtained through simulations.

The links between the `ics2` function from the **ICS** package and the functions available in the **ICSOutlier** package are illustrated on Figure 1.

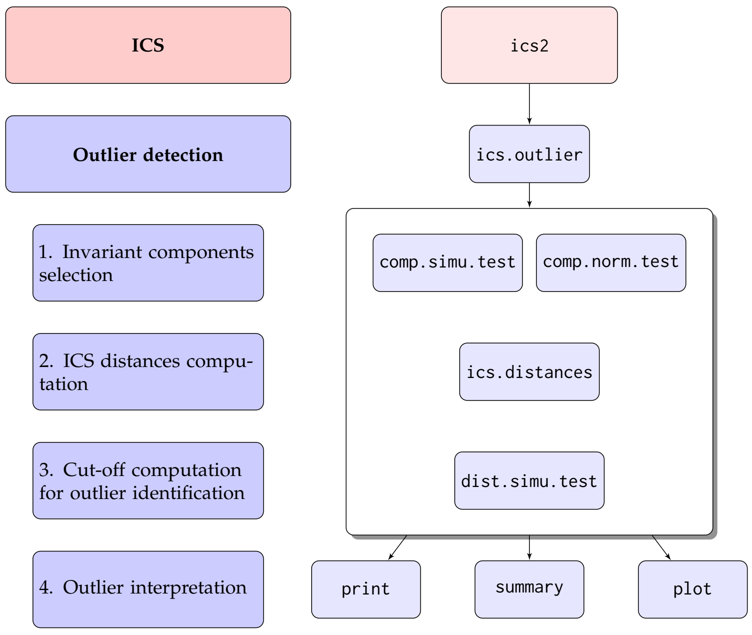
ICS
Outlier detection
1. Invariant components selection
2. ICS distances computation
3. Cut-off computation for outlier identification
4. Outlier interpretation
ics2
ics.outlier
comp.simu.test
comp.norm.test
ics.distances
dist.simu.test
print
summary
plot

Figure 1: Organization chart of the functions called by `ics.outlier`

The `ics.outlier` function only takes as parameter an object of class `ics2` and not an object of class `ics` because this later returns only uncentered invariant coordinates contrary to the `ics2` class. From a practical point a view, the `S1` and `S2` arguments of the `ics2` function should be the name of the function which returns a list with the first (resp. second) location vector $\mathbf{m}_1$ (resp. $\mathbf{m}_2$) and the first scatter matrix $\mathbf{V}_1$ (resp. $\mathbf{V}_2$).

In addition to this ics2 object, there are other parameters of ics.outlier that can be tuned, including an option to parallelize the computations in order to increase the speed of the function:

- method: name of the method used to select the ICS components. Options are "simulation" for the parallel analysis approach and "norm.test" for the normality tests approach. Depending on the method, either comp.norm.test or comp.simu.test is used.
- test: name of the marginal normality test to use if method = "norm.test". Possibilities are "jarque.test", "anscombe.test", "bonett.test", "agostino.test", "shapiro.test". Default is "agostino.test".
- mEig: number of simulations performed to derive the cut-off values for selecting the ICS components. Only if method = "simulation". See comp.simu.test for details.
- level.test: level for the comp.norm.test or comp.simu.test functions. The inital level for selecting the invariant coordinates.
- adjust: logical. For selecting the invariant coordinates, the level of the test can be adjusted for each component to deal with multiple testing. See comp.norm.test and comp.simu.test for details. Default is TRUE.
- level.dist: level for the dist.simu.test function. The (1-level(s))th quantile(s) used to determine the cut-off value(s) for the ICS distances.
- mDist: number of simulations performed to derive the cut-off value for the ICS distances. See dist.simu.test for details.
- type: currently the only option is "smallprop" which means that only the first ICS components can be selected. See comp.norm.test or comp.simu.test for details.
- ncores: number of cores to be used in dist.simu.test and comp.simu.test. If NULL or 1, no parallel computing is used. Otherwise makeCluster with type = "PSOCK", from the **parallel** package, is used.
- iseed: The seed passed on to clusterSetRNGStream if parallel computation is used. Default is NULL which means no fixed seed is used.
- pkg: A character vector listing all the packages which need to be loaded on the different cores via require if parallel computation is used. Must be at least "ICSOutlier" and must contain the packages needed to compute the scatter matrices.
- qtype: specifies the quantile algorithm used in quantile.

By default the ics2 function performs ICS with the usual mean vector and the usual empirical variance-covariance matrix returned by MeanCov and the location vector based on the third moments associated to the scatter matrix based on the fourth moments returned by Mean3Cov4.

Examples

In the following examples, we illustrate first, using an artifical data set, how ics.outlier behaves in the case of no outlier and then apply the function to three data sets available in R. For reproducibility all examples have a fixed seed.

Example with no outlier

One of the advantages of using ICS for outlier detection is its ability to detect the absence of outliers. If the first invariant coordinate is considered normal that means there is no outlier in the data set. So, in presence of a normal multivariate data set the function should select in most cases no component and hence will not detect outliers.

For the first example we simulate a bivariate normally distributed data set with 500 observations and we determine the cut-offs for identifying the outliers at the level 2.5% for all methods.

```
library("ICSOutlier", quietly = TRUE)
# Data simulation
set.seed(123)
X <- matrix(rnorm(1000, 0, 0.1), 500, 2)
```

Using the default ICS setting and choosing the Jarque-Bera test of kurtosis to select the components at the default level 5% can easily be done as follows.

```
icsX <- ics2(X)
icsOutlierJB <- ics.outlier(icsX, test = "jarque", level.test = 0.05,
                            level.dist = 0.025)
print(icsOutlierJB)

R> [1] "0 components were selected and no outliers were detected."
```

Hence in this outlier-free normal data no component was considered non-normal and therefore as desired no outliers were detected.

As a comparison we compute the robust Mahalanobis distances using the MCD with a breakdown point of 25% and compute the commonly used cut-off value coming from the χ^2_2 distribution and the adjusted cut-off value from (Green and Martin, 2017b) as implemented in the package **CerioliOutlierDetection**.

```
# Robust Mahalanobis distance with MCD estimates with a breakdown point of 25%
library("robustbase")
MCD <-  covMcd(X, alpha = 0.75)
RD <- mahalanobis(X,  MCD$center, MCD$cov)

# Cut-off based on the chi-square distribution
cutoff.chi.sq <- qchisq(0.975, df = ncol(X))
cutoff.chi.sq

R> [1] 7.377759

# Cut-off based Green and Martin (2017)
library("CerioliOutlierDetection")
cutoff.GM <- hr05CutoffMvnormal(n.obs = nrow(X), p.dim = ncol(X), mcd.alpha = 0.75,
                                signif.alpha = 0.025, method = "GM14",
                                use.consistency.correction = TRUE)$cutoff.asy
cutoff.GM

R> [1] 14.22071
```

To visualize these results, the robust squared Mahalanobis distances are plotted with two different cut-off values in Figure 2.

```
# Code for the Figure 2
colPoints <- ifelse(RD >= min(c(cutoff.chi.sq, cutoff.GM)), 1, grey(0.5))
pchPoints <- ifelse(RD >= min(c(cutoff.chi.sq, cutoff.GM)), 16, 4)
plot(seq_along(RD), RD,  pch = pchPoints, col = colPoints,
     ylim=c(0, max(RD, cutoff.chi.sq, cutoff.GM) + 2), cex.axis = 0.7, cex.lab = 0.7,
     ylab = expression(RD**2), xlab = "Observation Number")
abline(h = c(cutoff.chi.sq, cutoff.GM), lty = c("dashed", "dotted"))
legend("topleft", lty = c("dashed", "dotted"), cex = 0.7, ncol = 2, bty = "n",
       legend = c(expression(paste(chi[p]**2, " cut-off")), "GM cut-off"))
```

Hence in this example using the χ^2_2 cut-off yields 12 outliers while the more sophisticated cut-off does not classify any observation as outlying.

Another situation where the `ics.outlier` function can conclude to an absence of outliers is when the squared ICS distances are below the corresponding cut-off for all observations, as illustrated in the help file of the function: `?ics.outlier`.

HTP data set

The real data set `HTP`, introduced in Archimbaud et al. (2018), is included in this package. The data set provides the results of 88 numerical tests for 902 high-tech parts. Based on these results the producer considered all parts functional and all of them were sold. However two parts, 581 and 619, showed defects in use and were returned to the manufacturer. These two observations can thus be considered as outliers and the objective is to detect them. We use the `ics.outlier` function with its default settings, as most users initially would do but with the parallelizing option for the computations.

```
# HTP dataset
library("ICSOutlier")
set.seed(123)
```

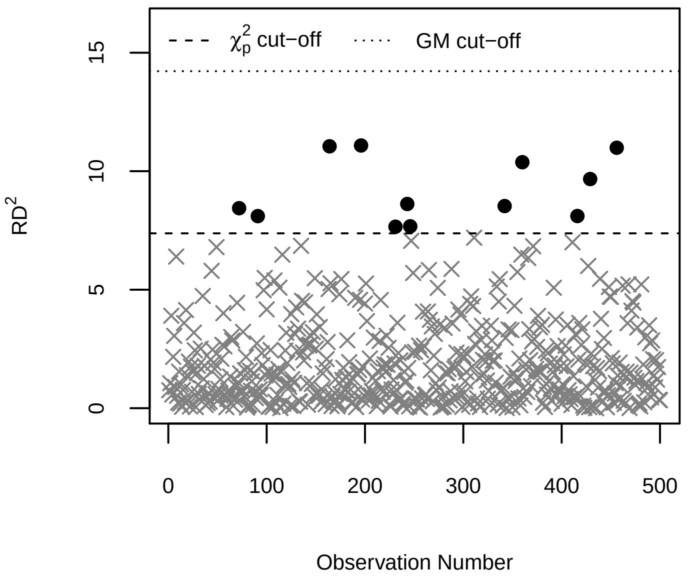

Figure 2: Squared robust Mahalanobis distances and two different cut-off values.

```
data(HTP)
outliers <- c(581, 619)

# default ICS
icsHTP <- ics2(HTP)

# Outlier detection with selection of components based on normality tests
# by default it can take quite long as mDist = 10000 so we choose to
# use all but one available cores to parallelize the simulations.
library(parallel)
icsOutlierDA <- ics.outlier(icsHTP, ncores = detectCores()-1, iseed = 123)
summary(icsOutlierDA)

R>
R> ICS based on two scatter matrices and two location estimates
R> S1:  MeanCov
R> S2:  Mean3Cov4
R>
R> Searching for a small proportion of outliers
R>
R> Components selected at nominal level 0.05: 14
R> Selection method: norm.test (agostino.test)
R> Number of outliers at nominal level 0.025: 43

# Code for the Figure 3
plot(icsOutlierDA, cex.lab = 0.7, cex.axis = 0.7)
points(outliers, icsOutlierDA@ics.distances[outliers], pch = 5)
text(outliers, icsOutlierDA@ics.distances[outliers], outliers, pos = 2, cex = 0.7)
```

Based on this result we are able to identify the two outliers among the 5% of the observations declared as outliers, taking into account the 14 first components selected by the D'Agostino normality test.

However, following Archimbaud et al. (2018), a simple alternative for selecting components is to use the screeplot of the `icsHTP` object, in a similar way as for PCA. Note that while the eigenvalues

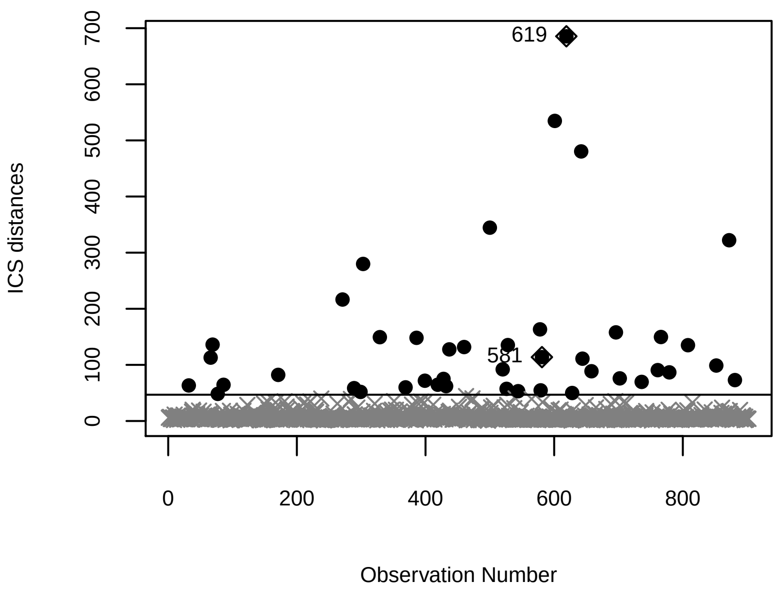

Figure 3: Squared ICS distances for HTP data with default parameters.

plotted on the screeplot are related to the variance, for PCA they correspond to a generalized kurtosis for ICS (see Tyler et al. (2009)).

```
# Code for the Figure 4
screeplot(icsHTP, cex.lab = 0.7, cex.axis = 0.7, cex.names = 0.7, cex.main = 0.7)
```

Based on this screeplot, three components might be a reasonable choice and then the functions `ics.distances` and `dist.simu.test` can be used to obtain the desired distances and cut-off value.

```
ics.dist.scree <- ics.distances(icsHTP, index = 1:3)
# by default it can take quite long as m = 10000, so we choose to
# use all but one available cores to parallelize the simulations.
library(parallel)
ics.cutOff <- dist.simu.test(icsHTP, 1:3, ncores = detectCores()-1, iseed = 123)
ics.cutOff

R>    97.5%
R> 12.80771
```

Before visualizing these results we also apply two competing outlier detection methods. First the Finite-Sample Reweighted MCD (FSRMCD) outlier detection test of Cerioli (Cerioli, 2010), implemented in the package **CerioliOutlierDetection** and secondly the SIGN1 algorithm (Filzmoser et al., 2008) implemented in the package **mvoutlier**. For the FSRMCD algorithm, we choose as nominal level $\alpha = 1 - \gamma^{1/n}$ for the individual outlier tests, with γ the nominal size of the intersection test and n the number of observations. Then the ICS distances based on the three selected components are plotted against the distances from the two competing methods together with the corresponding cut-off values.

```
# FSRMCD
library("CerioliOutlierDetection")
FSRMCD <- cerioli2010.fsrmcd.test(HTP, signif.alpha = 1 - 0.975**(1/nrow(HTP)),
                                  mcd.alpha = 0.75)
# Two critical values: one for points included in the reweighted MCD (weights == 1)
# and one for points excluded from the reweighted MCD (weights == 0)).
FSRMCD.cutoffs <- unique(FSRMCD$critvalfcn(FSRMCD$signif.alpha))
FSRMCD.cutoffs
```

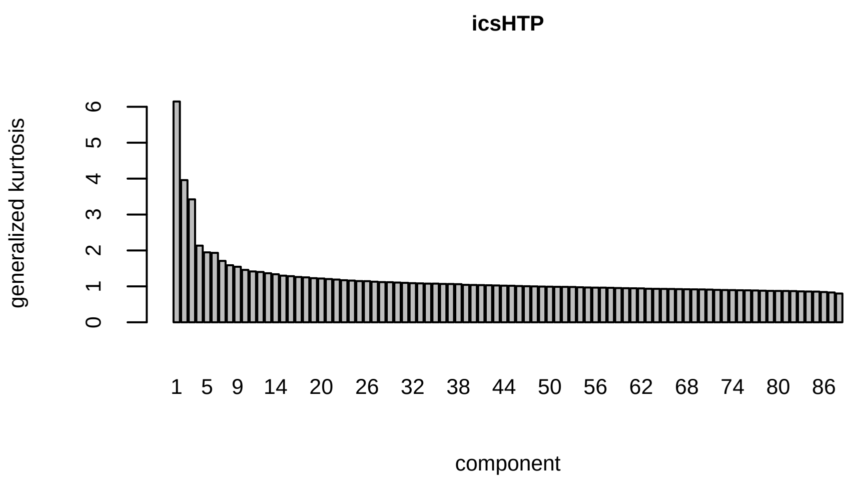

Figure 4: Screeplot of ICS eigenvalues for HTP data and default parameters.

```
R> [1] 144.8583 183.2797

# SIGN1
library("mvoutlier", quietly = TRUE)
SIGN1 <- sign1(HTP, qcrit = 0.975)

# Code for the Figure 5
par(mfrow = c(1, 2))
par(mar = c(4, 4, 2, 0.2))

# Comparison ICS vs FSRMCD
colPoints <- ifelse(ics.dist.scree >= ics.cutOff , 1, grey(0.5))
pchPoints <- ifelse(ics.dist.scree >= ics.cutOff, 16, 4)
plot(FSRMCD$mahdist.rw, ics.dist.scree, col = colPoints, pch = pchPoints,
     cex.lab = 0.7, cex.axis = 0.7, cex.main = 0.7,
     main = "ICS vs FSRMCD", ylab = "ICS Distances", xlab = "FSRMCD")
points(FSRMCD$mahdist.rw[outliers], ics.dist.scree[outliers], pch = 5)
text(FSRMCD$mahdist.rw[outliers], ics.dist.scree[outliers], labels = outliers, pos = 2,
     cex = 0.7)
abline(h = ics.cutOff, v = FSRMCD.cutoffs, lty = "dashed")

# Comparison ICS vs SIGN1
plot(SIGN1$x.dist, ics.dist.scree,  col = colPoints, pch = pchPoints,
     cex.lab = 0.7, cex.axis = 0.7, cex.main = 0.7,
     main = "ICS vs SIGN1", ylab = "ICS Distances", xlab = "SIGN1")
points(SIGN1$x.dist[outliers], ics.dist.scree[outliers], pch = 5)
text(SIGN1$x.dist[outliers], ics.dist.scree[outliers], labels = outliers, pos = 2,
     cex = 0.7)
abline(h = ics.cutOff, v = SIGN1$const, lty = "dashed")
```

As illustrated in Figure 5, the cut-offs used for the FSRMCD and SIGN1 algorithms plotted on the x-axis lead to too many false positives. Even when focusing only on the rank of the outlying observations it is clear that the ICS method outperforms the other methods. The outlier labelled 619 is obviously outlying with all methods, but the outlier labelled 581 has the 4th highest squared ICS distance compared to 15th and 12th for the distances computed with FSRMCD and SIGN1 algorithms. Note that the data set contains highly collinear variables and it is well-known that high-breakdown point methods are not optimal in this case.

Archimbaud et al. (2018) contains a more detailed analysis of this dataset comparing ICS with other methods and also gives the impact of the choice of different scatter matrices. For the present data set the default scatter combination used here seems the best. In this analysis, it is also interesting to notice that the first component highlights only the two defective parts. The other two observations are

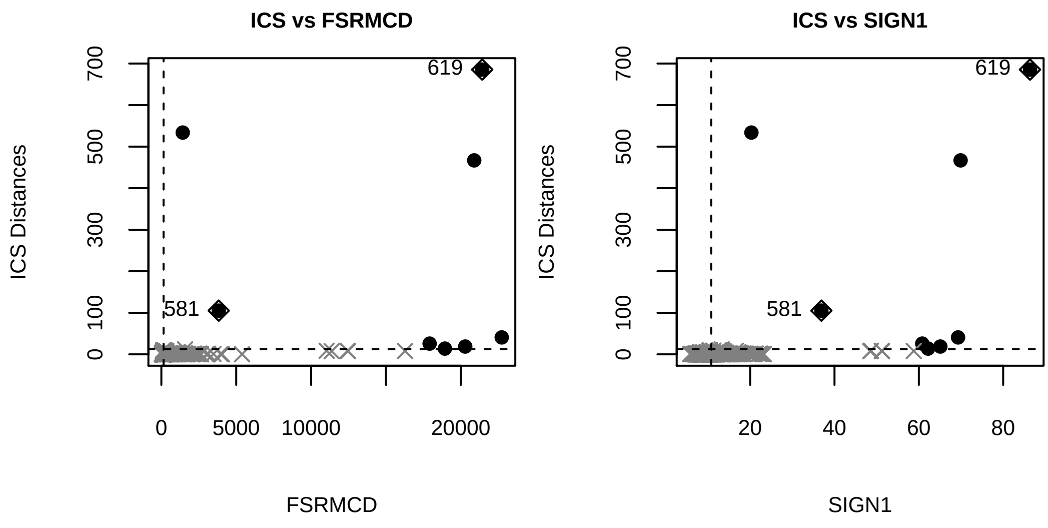

Figure 5: Squared ICS distances with 3 components against FSRMCD (left panel) and SIGN1 (right panel) measures of outlierness for HTP data with cut-offs at level of 2.5%.

detected on the second and the third components of ICS and as far as we know they are not defective parts but false positives.

Reliability data set

The Reliability data set comes also from an industrial context and is part the of **REPPlab** package. It contains 55 variables measured during the production process of 520 units. The challenge for this data is to identify the produced items with a fault not detected by the marginal tests. According to Fischer et al. (2016a), the observations numbered 414 and 512 are most likely outliers.

A special feature of this data set is that for example variable 24 is nearly constant which makes this data set hard to use with high-breakdown methods. For example the MCD cannot be computed for this data set when all variables are used. The use of ICS and other distance-based outlier detection methods is also detailed in Archimbaud et al. (2018).

Let us now illustrate the use of other scatter matrices than the default one together with the way to pass arguments to some of the functions. This time the invariant components are selected using simulations (Parallel analysis) with an initial decision level of 5%. These results are then compared to two methods not included in Archimbaud et al. (2018), namely the Local Outlier Factor (LOF, (Breunig et al., 2000)) and the Angle-Based Outlier Factor (Kriegel et al., 2008) outlier detection methods. First, we apply the LOF method with the `lof` function from the **Rlof** package which parallelizes the computation of the local outlier factor using l neighbours for each observation. The number of neighbours is determined based on the guidelines from Breunig et al. (2000): we compute the outlierness measures for l between 5 and 50 and we aggregate the results by taking the maximum of the local outlier factor for each observation. However, the distribution of the LOF values (and of the aggregated indices) is not known and therefore, there exists no theoretical cut-off to identify the outlying observations. In practice, observations with LOF values much higher than one are considered as potentially outlying. Second, we calculate the angle-based outlier factor for each observation through the `abod` function from the **abodOutlier** package on a random sample of 10% (the default) of the data (not on the entire data set because it is too time-consuming). As for the LOF method, the distribution of the outlyingness indices is unknown. But for this method, a potentially outlying observation is indicated by a small index value.

The location and scatter combinations we use here are the regular mean vector and covariance matrix and the joint maximum likelihood estimation of location and scatter of a t-distribution with one degree of freedom, also known as Cauchy MLE estimate. The function `MeanCov` returns the mean vector and regular covariance matrix as required for `ics2` and for the Cauchy MLE we use the function `tM` where the degree of freedom is specified using the argument `df`. As the Cauchy MLE is considered the more robust estimator it should be specified in `ics2` as `S1`.

```
# ReliabilityData example: the observations 414 and 512 are suspected to be outliers
library("ICSOutlier")
library("REPPlab")
```

```
set.seed(123)
data(ReliabilityData)
outliers <- c(414, 512)

# ICS with MLE Cauchy and the Mean-Cov
icsReliabilityData <- ics2(ReliabilityData, S1 = tM, S2 = MeanCov,
                           S1args = list(df = 1))

# Outlier detection with selection of components based on simulations
# it can take quite long as mEig = 5000 and mDist = 5000, so we choose
# to use all but one available cores to parallelize the simulations.
icsOutlierPA <- ics.outlier(icsReliabilityData, method = "simulation",
                            level.test = 0.05, mEig = 5000,
                            level.dist = 0.01, mDist = 5000,
                            ncores = detectCores()-1, iseed = 123)
icsOutlierPA
```

```
R> [1] "39 components were selected and 86 outliers were detected."
```

```
# LOF: Local Outlier Factor
library("Rlof", quietly = TRUE)
X.lof <- lof(ReliabilityData, 5:50, cores = 2)
X.lof.max <- apply(X.lof, 1, max)

# ABOD: Angle-Based Outlier Factor
library("abodOutlier", quietly = TRUE)
X.abod <- abod(ReliabilityData, method = "randomized")

# Code for the Figure 6
par(mfrow = c(1, 2))
par(mar = c(4, 4, 2, 0.2))

# Comparison ICS vs LOF
plot(X.lof.max, icsOutlierPA@ics.distances, cex.lab = 0.7, cex.axis = 0.7, pch = 4,
     cex.main = 0.7, main = "ICS vs LOF", ylab = "ICS Distances", xlab = "LOF")
text(X.lof.max[outliers], icsOutlierPA@ics.distances[outliers], labels = outliers,
     pos = 4, cex = 0.7)

# Comparison ICS vs ABOD
plot(X.abod, icsOutlierPA@ics.distances, cex.lab = 0.7, cex.axis = 0.7, cex.main = 0.7,
     main = "ICS vs ABOD", ylab = "ICS Distances", xlab = "ABOD", pch = 4)
text(X.abod[outliers], icsOutlierPA@ics.distances[outliers], labels = outliers,
     pos = 4, cex = 0.7)
```

This analysis shows that ICS selects here quite many components and suggests 86 outliers at the level of 1%. For their part, the LOF and ABOD methods do not provide cut-offs for identifying outlying observations. So we only consider the ranks of the observations. When looking at the ICS distances on Figure 6, the observations 414 and 512 are clearly separated from the main bulk of the data while the two observations do not differ at all from the rest of the data using the LOF and ABOD methods (see also Archimbaud et al. (2018) for more details on this exmaple).

HBK data set

The hbk data set is included in the **robustbase** package (Rousseeuw et al., 2017). It is an artificial data set created by Hawkins et al. (1984) for illustrating the so-called masking effect of outliers. It contains two groups of outliers with observations 1-10 in the first group and observations 11-14 in the second group, over the 75 observations characterized by the three explanatory variables. Usually with non-robust methods only the observations 12, 13 and 14 are identified as outliers. For this example the percentage of outliers is around 19% and so is larger than the percentage we recommend for the current version of the package. Our aim is to illustrate that in such a situation, the choice of the scatter matrices may have a large impact on the results.

First, the combination of the highly robust MCD estimates with the mean vector and regular covariance matrix is studied. Note that in order to be able to use the MCD via the function CovMcd a wrapper needs to be written around the function so that it returns the proper list as expected by ics2. Using then the D'Agostino test for skewness for selecting the invariant components leads to select two

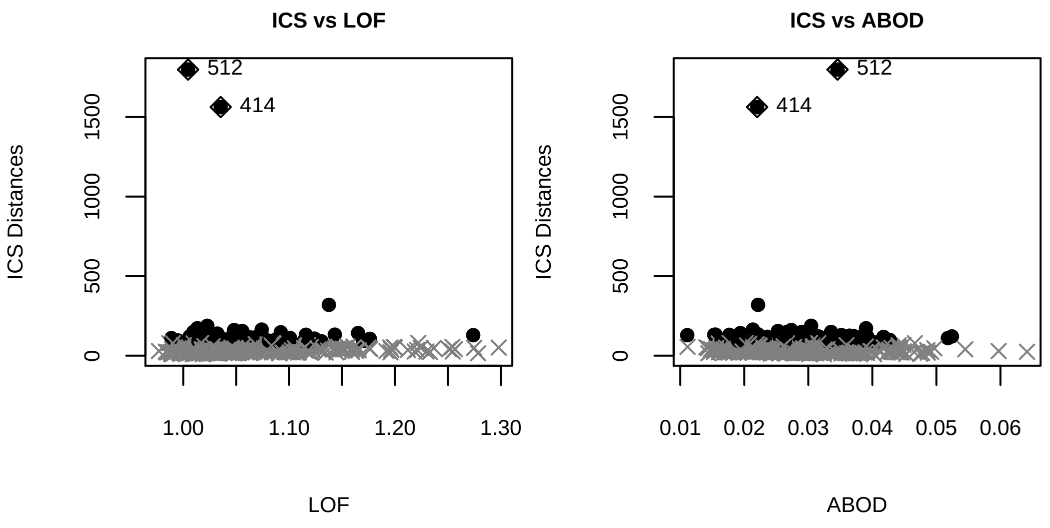

Figure 6: Squared ICS distances with 39 components against LOF (left panel) and ABOD (right panel) measures of outlierness for Reliability data.

components at the default level of 5%. The level of the quantile used for deriving the cut-off for outlier identification is the default, 2.5%. For this combination, ICS performs equally well as the MCD-based Mahalanobis distances, as illustrated in the first three plots of the Figure 7. All outliers are correctly identified, no false positive is detected and the two groups of outliers are clearly separated.

Next, we consider the combination of two non-robust location vectors and scatter matrices which are the default in ICS, with the same parameters for selecting the components and for identifying the outliers as previously. The results are presented on the fourth plot of Figure 7.

```
library("rrcov")
set.seed(123)
# HBK data set
data(hbk)

# ICS with MCD estimates and the usual estimates
# Need to create a wrapper for the CovMcd function to return first the location estimate
# and the scatter estimate secondly.
myMCD <- function(x,...){
  mcd <- CovMcd(x,...)
  return(list(location = mcd@center, scatter = mcd@cov))
}
icsHBK_mcd <- ics2(hbk[, 1:3], S1 = myMCD, S2 = MeanCov, S1args = list(alpha = 0.75))

# Outlier detection with selection of components based on the D'Agostino test for
# skewness. It can take quite long as mEig = 10000 and mDist = 10000, so we choose
# to use all but one available cores to parallelize the simulations.
library(parallel)
icsOutlierDA.MCD <- ics.outlier(icsHBK_mcd, mEig = 10000, level.dist = 0.025,
                                mDist = 10000,
                                ncores = detectCores()-1, iseed = 123,
                                pkg = c("ICSOutlier", "rrcov"))
icsOutlierDA.MCD

R> [1] "2 components were selected and 14 outliers were detected."

# Robust Mahalanobis distance with MCD estimates with a breakdown point of 25%
MCD <-  covMcd(hbk[, 1:3], alpha = 0.75)
RD <- mahalanobis(hbk[, 1:3],  MCD$center, MCD$cov)
cutoff.chi.sq <- qchisq(0.975, df = ncol(hbk[, 1:3]))

# Cut-off based Green and Martin (2017)
library("CerioliOutlierDetection")
cutoff.GM <- hr05CutoffMvnormal(n.obs = nrow(hbk[, 1:3]), p.dim = ncol(hbk[, 1:3]),
                                mcd.alpha = 0.75, signif.alpha = 0.025, method = "GM14",
                                use.consistency.correction = TRUE)$cutoff.asy

# ICS with non robust estimates
icsHBK <- ics2(hbk[,1:3], S1 = MeanCov, S2 = Mean3Cov4)

# Outlier detection with selection of components based on the D'Agostino test for
# skewness. It can take quite long as mEig = 10000 and mDist = 10000, so we choose
# to use all but one available cores to parallelize the simulations.
icsOutlierDA <- ics.outlier(icsHBK, mEig = 10000, level.dist = 0.025,
                                mDist = 10000,
                                ncores = detectCores()-1, iseed = 123,
                                pkg = c("ICSOutlier", "rrcov"))
icsOutlierDA
```

```
R> [1] "2 components were selected and 2 outliers were detected."
```

```
# Code for the Figure 7
par(mfrow = c(1, 3))
par(mar = c(4, 2, 2, 0.2))

# Robust Mahalanobis distance with MCD estimates
colPoints <- ifelse(RD >= cutoff.chi.sq, 1, grey(0.5))
pchPoints <- ifelse(RD >= cutoff.chi.sq, 16, 4)
plot(RD, col = colPoints, pch = pchPoints, xlab = "Observation Number",
     cex.lab = 0.7, cex.axis = 0.7, main = "Robust MD", cex.main = 0.7)
abline(h = c(cutoff.chi.sq, cutoff.GM), lty = 1:2)
legend("topright", lty = 1:2, cex = 0.7, bty = "n",
        legend = c(expression(paste(chi[p]**2, " cut-off")), "GM cut-off"))

# ICS with MCD estimates and regular covariance
plot(icsOutlierDA.MCD,  cex.lab = 0.7, cex.axis = 0.7, main = "ICS MCD-COV",
     cex.main = 0.7)

# ICS with default non-robust estimates
plot(icsOutlierDA, cex.lab = 0.7, cex.axis = 0.7, main = "ICS COV-COV4",
     cex.main = 0.7)
```

Here the default ics2 scatter combination does not perform well. Only three outliers from the second group are detected, although the distance plot indicates three distance levels but the cut-off value is not good and the separation not clear especially when compared with the MCD-COV combination. However, when looking at the invariant coordinates on Figure 8, we can see that the default scatter combination reveals the 14 outliers and the masking effect appears only when computing the ICS distances.

```
# Code for the Figure 8
plot(icsHBK, col = rep(3:1, c(10, 4, 61)))
```

Conclusion and future developments

The use of **ICSOutlier** for outlier detection has been illustrated on several examples. The results are already pretty good with the default parameters as soon as the percentage of outliers is small. Note however that the components selection procedure can be improved sometimes by a visual inspection of the screeplot and the cut-off for outlier identification can be adjusted by looking at the ICS distances plot. The ability of the method to check for the absence of outliers is an advantage compared with methods such as the Mahalanobis distance. In the same vein, the number of false positives, i.e the number of non-outliers declared as outliers, is restricted thanks to the use of the cut-off derived from simulations. Moreover, the real HTP data set included in the package shows that the ICS method is useful for outlier detection in the context of quality control, among possible applications. Finally the ICS method competes with common approaches based on distances (the Mahalanobis distance, its robust version, the PCA methods, its variants and improvements) as well as other methods based on the density (LOF) or on angles (ABOD).

The computation time for cut-offs for the selection of the non-Gaussian invariant components, on the one hand, and for the threshold for labeling the outliers, on the other hand, can be reduced

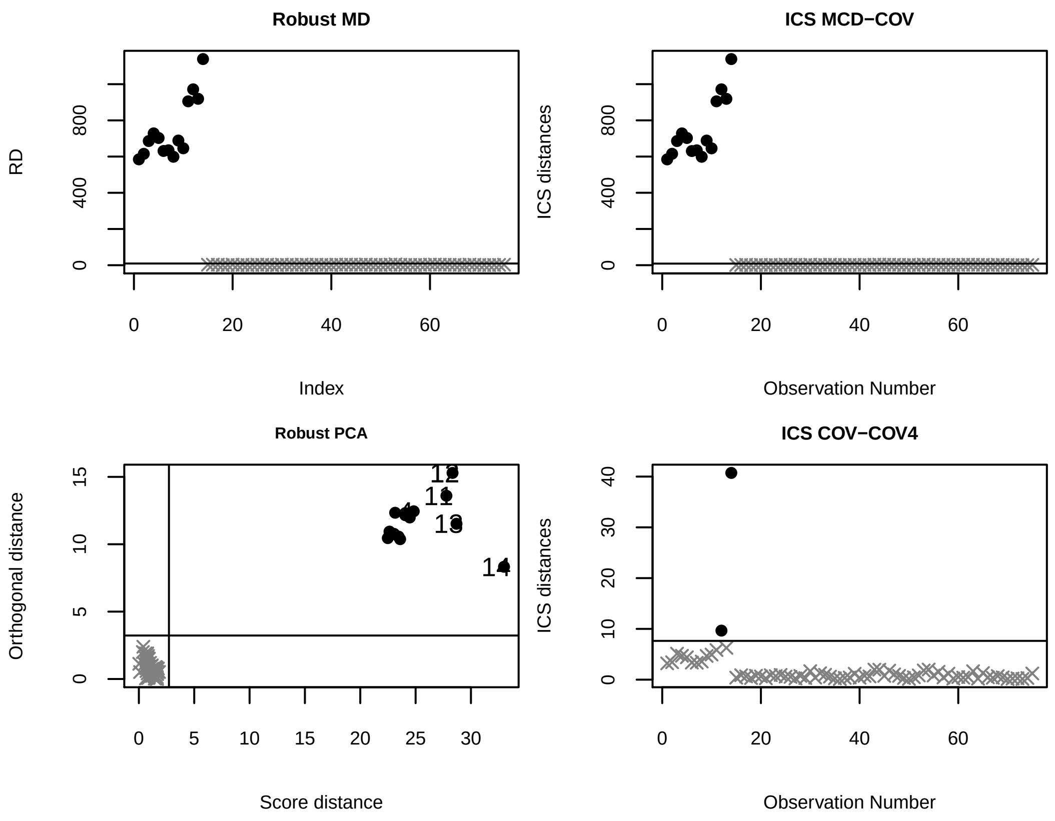

Figure 7: Squared robust Mahalanobis distance based on the MCD (left panel), squared ICS distances with MCD and Mean-Cov (middle panel), and squared ICS distances with default scatters (right panel). All cut-offs values are at level of 2.5%

by using the parallelizing option. Moreover, for now, the function can only select the first invariant components which is relevant when the percentage of outliers is small. For a large percentage, the invariant components associated with the smallest eigenvalues may also be of interest and this is a current perspective of the present package. A generalization of the ICS method in case of perfectly collinear variables is also a work in progress.

Acknowledgments

The article is based upon work from CRoNoS COST Action IC1408, supported by COST (European Cooperation in Science and Technology).

The authors wish to thank the editor and the two reviewers for their comments and suggestions which helped improve not only the paper but also the R package.

Bibliography

C. C. Aggarwal. *Outlier Analysis, 2nd Edition*. Springer-Verlag, 2017. ISBN 978-3-319-47577-6. URL https://doi.org/10.1007/978-3-319-47578-3. [p234]

A. Archimbaud, K. Nordhausen, and A. Ruiz-Gazen. *ICSOutlier: Outlier Detection Using Invariant Coordinate Selection*, 2016. URL https://CRAN.R-project.org/package=ICSOutlier. R package version 0.3-0. [p235]

A. Archimbaud, K. Nordhausen, and A. Ruiz-Gazen. Ics for multivariate outlier detection with application to quality control. *Computational Statistics & Data Analysis*, 128:184–199, 2018. ISSN 0167-9473. URL https://doi.org/10.1016/j.csda.2018.06.011. [p235, 236, 237, 239, 240, 242, 243, 244]

D. G. Bonett and E. Seier. A test of normality with high uniform power. *Computational Statistics & Data Analysis*, 40(3):435–445, 2002. URL https://doi.org/10.1016/S0167-9473(02)00074-9. [p236]

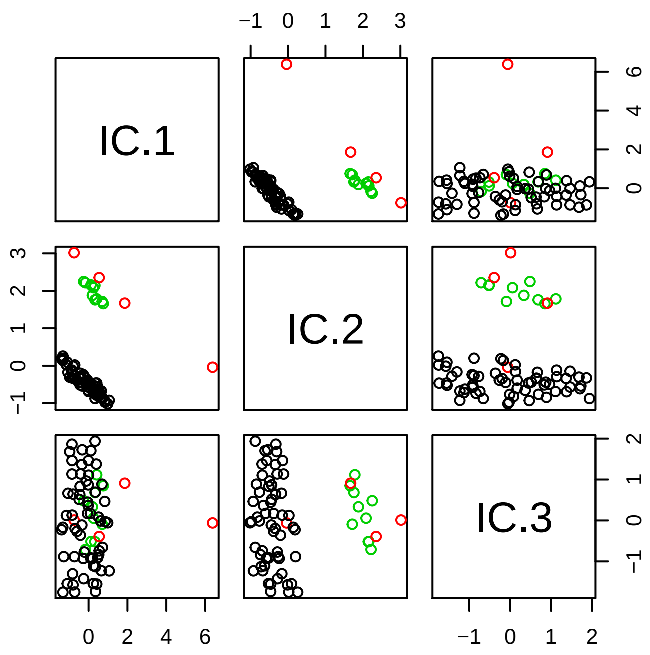

Figure 8: Scatter plot of the invariant coordinates of the hbk data for the default scatter combinations. The two outlier groups are marked with different colors.

M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *ACM Sigmod Record*, volume 29, pages 93–104. ACM, 2000. URL https://doi.org/10.1145/335191.335388. [p243]

A. Cerioli. Multivariate outlier detection with high-breakdown estimators. *Journal of the American Statistical Association*, 105(489):147–156, 2010. URL https://doi.org/10.1198/jasa.2009.tm09147. [p234, 241]

R. P. Chalmers and D. B. Flora. Faoutlier: An R package for detecting influential cases in exploratory and confirmatory factor analysis. *Applied Psychological Measurement*, 39(7):573–574, 2015. URL https://doi.org/10.1177/0146621615597894. [p234]

S. Dray. On the number of principal components: A test of dimensionality based on measurements of similarity between matrices. *Computational Statistics & Data Analysis*, 52(4):2228 – 2237, 2008. URL https://doi.org/10.1016/j.csda.2007.07.015. [p236]

S.-H. Eo and H. Cho. *OutlierDC: Outlier Detection Using Quantile Regression for Censored Data*, 2014. URL https://CRAN.R-project.org/package=OutlierDC. R package version 0.3-0. [p234]

C. Fan. *HighDimOut: Outlier Detection Algorithms for High-Dimensional Data*, 2015. URL https://CRAN.R-project.org/package=HighDimOut. R package version 1.0.0. [p234]

P. Filzmoser and M. Gschwandtner. *Mvoutlier: Multivariate Outlier Detection Based on Robust Methods*, 2017. URL https://CRAN.R-project.org/package=mvoutlier. R package version 2.0.8. [p234]

P. Filzmoser and V. Todorov. Robust tools for the imperfect world. *Information Sciences*, 245:4–20, 2013. URL https://doi.org/10.1016/j.ins.2012.10.017. [p234]

P. Filzmoser, R. G. Garrett, and C. Reimann. Multivariate outlier detection in exploration geochemistry. *Computers & Geosciences*, 31(5):579–587, 2005. URL https://doi.org/10.1016/j.cageo.2004.11.013. [p234]

P. Filzmoser, R. Maronna, and M. Werner. Outlier identification in high dimensions. *Computational Statistics & Data Analysis*, 52(3):1694–1711, 2008. URL https://doi.org/10.1016/j.csda.2007.05.018. [p241]

D. Fischer, A. Berro, K. Nordhausen, and A. Ruiz-Gazen. REPPlab: An R package for detecting clusters and outliers using exploratory projection pursuit. arXiv preprint arXiv:1612.06518, 2016a. URL https://arxiv.org/abs/1612.06518. [p243]

D. Fischer, A. Berro, K. Nordhausen, and A. Ruiz-Gazen. *REPPlab: R Interface to 'EPP-Lab', a Java Program for Exploratory Projection Pursuit*, 2016b. URL https://CRAN.R-project.org/package=REPPlab. R package version 0.9.4. [p234]

C. Fraley. *HDoutliers: Leland Wilkinson's Algorithm for Detecting Multidimensional Outliers*, 2016. URL https://CRAN.R-project.org/package=HDoutliers. R package version 0.15. [p234]

M. Genest, J.-C. Masse, and J.-F. Plante. *Depth: Nonparametric Depth Functions for Multivariate Analysis*, 2017. URL https://CRAN.R-project.org/package=depth. R package version 2.1-1. [p234]

C. G. Green and D. Martin. *CerioliOutlierDetection: Outlier Detection Using the Iterated RMCD Method of Cerioli (2010)*, 2017a. URL https://CRAN.R-project.org/package=CerioliOutlierDetection. R package version 1.1.9. [p234]

C. G. Green and R. D. Martin. An extension of a method of Hardin and Rocke, with an application to multivariate outlier detection via the IRMCD method of Cerioli. Technical report, Working Paper, 2017b. URL http://christopherggreen.github.io/papers/hr05_extension.pdf. [p239]

A. S. Hadi, A. Imon, and M. Werner. Detection of outliers. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(1):57–70, 2009. URL https://doi.org/10.1002/wics.6. [p234]

D. M. Hawkins, D. Bradu, and G. V. Kass. Location of several outliers in multiple-regression data using elemental sets. *Technometrics*, 26(3):197–208, 1984. URL https://doi.org/10.1080/00401706.1984.10487956. [p244]

V. J. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22 (2):85–126, 2004. URL https://doi.org/10.1007/s10462-004-4304-y. [p234]

Y. Hu, W. Murray, Y. Shan, and Australia. *Rlof: R Parallel Implementation of Local Outlier Factor (LOF)*, 2015. URL https://CRAN.R-project.org/package=Rlof. R package version 1.1.1. [p234]

J. Jimenez. *abodOutlier: Angle-Based Outlier Detection*, 2015. URL https://CRAN.R-project.org/package=abodOutlier. R package version 0.1. [p234]

L. Komsta. *Outliers: Tests for Outliers*, 2011. URL https://CRAN.R-project.org/package=outliers. R package version 0.14. [p234]

H.-P. Kriegel, A. Zimek, and others. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 444–452. ACM, 2008. URL https://doi.org/10.1145/1401890.1401946. [p243]

K. Luu and M. Blum. *Pcadapt: Fast Principal Component Analysis for Outlier Detection*, 2017. URL https://CRAN.R-project.org/package=pcadapt. R package version 3.0.4. [p234]

K. Nordhausen and D. E. Tyler. A cautionary note on robust covariance plug-in methods. *Biometrika*, 102(3):573–588, 2015. URL https://doi.org/10.1093/biomet/asv022. [p235]

K. Nordhausen, H. Oja, and D. E. Tyler. Tools for exploring multivariate data: The package ICS. *Journal of Statistical Software*, 28(6):1–31, 2008. URL https://doi.org/10.18637/jss.v028.i06. [p235, 237]

P. R. Peres-Neto, D. A. Jackson, and K. M. Somers. How many principal components? stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics & Data Analysis*, 49 (4):974 – 997, 2005. URL https://doi.org/10.1016/j.csda.2004.06.015. [p236]

A. Rehage and S. Kuhnt. *alphaOutlier: Obtain Alpha-Outlier Regions for Well-Known Probability Distributions*, 2016. URL https://CRAN.R-project.org/package=alphaOutlier. R package version 1.2.0. [p234]

P. Rousseeuw and M. Hubert. High-breakdown estimators of multivariate location and scatter. In *Robustness and Complex Data Structures*, pages 49–66. Springer-Verlag, 2013. URL https://doi.org/10.1007/978-3-642-35494-6_4. [p235]

P. Rousseeuw, C. Croux, V. Todorov, A. Ruckstuhl, M. Salibian-Barrera, T. Verbeke, M. Koller, and and Martin Mächler. *robustbase: Basic Robust Statistics*, 2017. URL http://CRAN.R-project.org/package=robustbase. 0.92-8. [p244]

V. Todorov. *rrcovHD: Robust Multivariate Methods for High Dimensional Data*, 2016. URL `https://CRAN.R-project.org/package=rrcovHD`. R package version 0.2-5. [p234]

V. Todorov and P. Filzmoser. An object-oriented framework for robust multivariate analysis. *Journal of Statistical Software*, 32(3):1–47, 2009. URL `https://doi.org/10.18637/jss.v032.i03`. [p234]

L. Torgo. *Data Mining with R, Learning with Case Studies, 2nd Edition*. Chapman and Hall/CRC, 2016. URL `http://ltorgo.github.io/DMwR2`. [p234]

D. E. Tyler, F. Critchley, L. Dümbgen, and H. Oja. Invariant coordinate selection. *Journal of the Royal Statistical Society B*, 71(3):549–592, 2009. URL `https://doi.org/10.1111/j.1467-9868.2009.00706.x`. [p235, 236, 241]

M. van der Loo. *Extremevalues, an R Package for Outlier Detection in Univariate Data*, 2010. URL `http://www.github.com/markvanderloo/extremevalues`. R package version 2.3. [p234]

K. Williams. *Ldbod: Local Density-Based Outlier Detection*, 2017. URL `https://CRAN.R-project.org/package=ldbod`. R package version 0.1.2. [p234]

B. Yazici and S. Yolacan. A comparison of various tests of normality. *Journal of Statistical Computation and Simulation*, 77(2):175–183, 2007. URL `https://doi.org/10.1080/10629360600678310`. [p236]

Aurore Archimbaud
TSE-R, University Toulouse 1 Capitole
21 allée de Brienne, 31000 Toulouse
France
`aurore.archimbaud@ut-capitole.fr`

Klaus Nordhausen
CSTAT - Computational Statistics, Institute of Statistics & Mathematical Methods in Economics
Vienna University of Technology, Wiedner Hauptstr. 7, A-1040 Vienna
Austria
(0000-0002-3758-8501)
`klaus.nordhausen@tuwien.ac.at`

Anne Ruiz-Gazen
TSE-R, University Toulouse 1 Capitole
21 allée de Brienne, 31000 Toulouse
France
`anne.ruiz-gazen@tse-fr.eu`