Austrian Journal of Statistics
March 2016, Volume 45, 3–23.
http://www.ajs.or.at/
doi:10.17713/ajs.v45i1.86

Treatment of Multivariate Outliers in Incomplete Business Survey Data

Marc Bill
FHNW School of Business

Beat Hulliger
FHNW School of Business

Abstract

The distribution of multivariate quantitative survey data usually is not normal. Skewed and semi-continuous distributions occur often. In addition, missing values and non-response is common. All together this mix of problems makes multivariate outlier detection difficult. Examples of surveys where these problems occur are most business surveys and some household surveys like the Survey for the Statistics of Income and Living Condition (SILC) of the European Union. Several methods for multivariate outlier detection are collected in the R package **modi**. This paper gives an overview of the package **modi** and its functions for outlier detection and corresponding imputation. The use of the methods is explained with a business survey data set. The discussion covers pre- and post-processing to deal with skewness and zero-inflation, advantages and disadvantages of the methods and the choice of the parameters.

Keywords: outlier detection, missing value, zero-inflation, imputation, R package.

1. Introduction

In surveys on monetary values, often several monetary variables are collected in order to capture the economic situation of an entity. This holds for business surveys, where many particular types of expenditures may be asked. Examples are surveys on expenditures for research and development or investments and expenditures for environment protection. Also for household or person surveys on the economic situation, several economic variables are needed. Examples are surveys on the economic situation of students, where sources of financing, expenditures for dwelling, travelling, food etc. are needed, or household surveys like the Statistics on Income and Living Conditions (SILC) of the European Union where various income sources are collected. Of course also non-monetary quantitative variables may be collected like various health indicators in a health survey or physical production parameters in a business survey or in a survey on livestock of farms. All these surveys have some common features: They have a complex sample design including stratification and possibly sub-sampling; they have elaborated questionnaires; they have unit and item non-response, and they typically have zero inflated distributions because of the multi-faceted economic situation. Zero-inflation occurs because a particular entity usually only needs a subset of the possible dimensions to describe its situation. For example in the SILC surveys, retired persons

usually (but not always) do not have labour income. Or in a business survey on environment protection, expenditures of an educational institution may not have investments into waste water treatment.

It is challenging to deal with outliers in these situations because, in addition to the above problems, the size effect of families, farms or businesses may yield heavily skewed distributions. There may be outliers which are correct observations and which must be taken into account when population totals, like waste water treatment expenditures of a branch of economy, must be estimated. However, taking the outliers into account introduces a large variability and, if actually the values are not correct, entail a large bias, too. Chambers (1986) introduced the notion of a representative outlier to conceptualise this dilemma.

In sample surveys, the definition of an outlier usually cannot rely on a parametric model. The outlier generating mechanisms may depend on other variables and therefore are not simple mixture models as in classical robust statistics (see, e.g., Hampel, Ronchetti, Rousseeuw, and Stahel 1986). Therefore, Béguin and Hulliger (2008) introduced the notion of an outlier at random, and Hulliger and Schoch (2013) discuss a full model of outlier generating mechanisms and the connections between missingness and outlyingness.

It is important to note that the results in the following example differ widely whether weights are used or not. In general, the outlyingness, the missingness mechanisms (item non-response), the sample design and the unit non-response mechanism are correlated and cannot be ignored. The package **modi** (Hulliger 2015) for the statistical environment R (R Core Team 2014) can handle outlyingess, missingness and sample design issues at once and in a multivariate framework.

To the best of our knowledge, the R package **modi** is at the moment the only package that at the same time deals with missing values and survey weights. The package **rrcovNA** (Todorov 2014b) is an extension of the package **rrcov** (Todorov and Filzmoser 2009; Todorov 2014a) with methods that cope with missing values but does not take survey weights into account. Therefore, a comparison is difficult. A notable paper comparing the outlier detection algorithms of an earlier version of the packages **rrcovNA** and **modi** is Todorov, Templ, and Filzmoser (2011).

Section 2 gives an overview of the package **modi**. Section 3 introduces the SEPE data set which is used in Section 4 to show the application of different methods of the package. Section 5 gives a conclusion.

2. Overview of the modi package

Several multivariate outlier detection and imputation procedures are contained in Version 1.6 of the package **modi**. The BACON-EEM algorithm (Béguin and Hulliger 2008, function `BEM()`) detects outliers under the assumption of a multivariate normal distribution. The BACON-EEM algorithm starts from an outlier free subset, uses the EM-algorithm to estimate the center and scatter of the observations of this subset and then judges outlyingness of the full data set by the Mahalanobis distance (MD) in order to define a new outlier free subset. This procedure is iterated until convergence. Sampling weights are taken into account. The Transformed Rank Correlation (TRC) algorithm (Béguin and Hulliger 2004, function `TRC()`) uses Spearman rank correlations and, similar to Maronna and Zamar (2002), an orthogonal transformation to arrive at a robust estimate of the mean and covariance. Before the transformation, missing values are imputed provisionally by simple robust regression. The Epidemic Algorithm (EA) (Béguin and Hulliger 2004) is based on a type of data depth. It is run forward to detect outliers (function `EAdet()`) and backward to impute for outliers (function `EAimp()`). The GIMCD algorithm (Béguin and Hulliger 2008, function `GIMCD()`) uses non-robust Gaussian imputation, i.e. an EM-algorithm, followed by a highly robust minimum covariance determinant (MCD) algorithm (Rousseeuw and Van Driessen 1999). The GIMCD algorithm is similar to the poor man's algorithm of Todorov *et al.* (2011). The nearest neigh-

Table 1: Algorithms of the package **modi**.

Algorithm	Function	Use
BACON-EEM	`BEM()`	Detection of outliers
Epidemic for detection	`EAdet()`	Detection of outliers
Epidemic for imputation	`EAimp()`	Imputation of outliers and `NA`'s
Transformed Rank Correlation	`TRC()`	Detection of outliers
ER	`ER()`	Detection of outliers
Gaussian imputation and MCD	`GIMCD()`	Detection of outliers
Nearest Neighbour Imputation	`POEM()`	Imputation of outliers and `NA`'s
Winsorization and Gaussian imputation	`Winsimp()`	Imputation of outliers and `NA`'s

bour imputation algorithm POEM can cope with outliers (Charlton 2003, function `POEM()`). Another algorithm of **modi** for imputation is based on winsorization followed by a multivariate normal model (function `Winsimp()`). Also the first robust multivariate detection algorithm which was adapted to missing values by Little and Smith (1987) is included (function `ER()`). A list of the algorithms of the **modi** package is shown in Table 1.

In addition to the algorithms, the **modi** package contains a set of utility functions which are mostly internal. To mention are a plot for Mahalanobis distances (function `PlotMD()`) based on the χ^2-distribution or the F-distribution, implementing the proposal by Little and Smith (1987), function `weighted.var()` to calculate weighted variances analogue to the base function `weighted.mean()`, and function `MDmiss()` to calculate Mahalanobis distances when missing values occur.

The package **modi** contains two data sets. The bushfire data (Campbell 1989) with a version that contains missing values (`bushfirem`) and a set of (fictive) weights (`bushfire.weights`). The bushfire data set is used in the examples of the package documentation. The second data set `sepe` stems from a real survey on environment expenditures of private companies carried out by the Swiss Federal Statistical Office. It is explained in detail in Section 3.

3. The SEPE data set

The `sepe` data set is an anonymised sample of the pilot survey on environment protection expenditures of the Swiss private economy conducted in 1993 by the Swiss Federal Statistical Office. The units are enterprises and the monetary variables are in thousand Swiss Francs (CHF). The data contain 675 observations on 23 variables overall. The sample design is stratified according to branch of economy and size. The sampling rate increases with the size class of the strata. For confidentiality reasons, a random subsample was chosen from the original sample and certain enterprises were excluded specifically. In addition, random small perturbation was added to certain variables, and some categories have been collapsed. The data set has missing values where in the original data collection the respondent indicated that the value was a guess rather than copied from records. The `sepe` data set has first been prepared for the FP5 project EUREDIT (Charlton 2003) and later been used as protected data for educational purposes. For this demonstration of the **modi** package, we focus on 8 variables representing the most important expenditure-areas (`exp`) and investment-areas (`inv`). In particular, the areas are water protection(`wp`), waste management (`wm`) and air protection (`ap`). The variables for noise protection (`np`) and other protection areas (`op`) are excluded from the following examples. The chosen variables are `totinvwp`, `totinvwm`, `totinvap`, `totinvto`, `totexpwp`, `totexpwm`, `totexpap` and `totexpto`, where the prefix `tot` indicates that the variables are the aggregates over all types of investment or expenditures. The variable naming includes the abbreviated type of spending in the middle and the area at the end. The variables `totinvto` and `totexpto` are the overall total expenditure and

Table 2: Summary statistics of the SEPE data (original scale).

	Mean	Std-Dev	Min	Med	Max	No. Miss
totinvwp	23.68	212.44	0	0	8400	48
totinvwm	15.57	342.70	0	0	18108	53
totinvap	78.13	1331.12	0	0	88248	53
totinvto	136.66	1490.55	0	0	88359	82
totexpwp	22.38	176.93	0	0	8800	90
totexpwm	30.42	236.87	0	3	6490	118
totexpap	8.35	125.86	0	0	8430	72
totexpto	58.11	340.41	0	7	16440	161

investment in all environmental protection areas, respectively. Since these overall totals have been collected, the balance condition (sub-totals adding to totals) does not always hold. These inconsistencies are not dealt with in the present analysis but methods for their treatment can be found in Luzi, De Waal, Hulliger, Di Zio, Pannekoek, Kilchmann, Guarnera, Hoogland, Manzari, and Tempelman (2007) and Charlton (2003). The sampling weight (`weight`) has been adjusted for non-response in the stratum by using the ratio of population size divided by the net sample size.

The descriptive univariate statistics in Table 2 show the strong asymmetry present in the data. To handle this, in a first step we logarithmize the data set by applying the transformation `log(x+1)`.

```
> data("sepe")
> sepevar8 <- c("totinvwp", "totinvwm", "totinvap", "totinvto",
+               "totexpwp", "totexpwm", "totexpap", "totexpto")
> data <- log(sepe[, sepevar8] + 1)
```

The `pairs()` plot in Figure 1 of the transformed data reveals the previously discussed positive correlation between the overall totals (`totexpto` and `totinvto`) and the subcategories. A large part of the observations lies on the axes, reflecting zeros, but there is always a part of the data forming a more or less elliptically shaped bivariate distribution. Univariate and bivariate outliers are visible. Besides 2'595 items containing the value zero, we have 677 missing values out of 5'400 items considered here. Hence, for the algorithms BACON-EEM, TRC and GIMCD, we need an additional step in data preparation. Because these algorithms have an underlying distributional assumption of multivariate normality, they have difficulties handling data sets with zero inflated distributions. Declaring the zeros as missing values, brings the data into the elliptical-like shape needed to run these functions.

```
> sepenozero <- recode(sepe, "0=NA")
> datanozero <- log(sepenozero[, sepevar8] + 1)
```

If we do not set the zeros to missing values, additional problems occur: The values of the median absolute deviation (MAD) may be 0 and standardisation is not possible then or the covariance matrix is singular. In the first case, the MADs can be substituted by a user specified probability quantile of the absolute deviations from the median. For the second case, we would have to reduce the dimension of the data set by omitting variables.

In the sections on the specific functions, we give recommendations at which stage it is preferable to reintroduce the removed information of zeros. Except for the Epidemic Algorithm, it is indispensable for outlier detection to declare zeros as missing values or to treat the zero-inflation in some way.

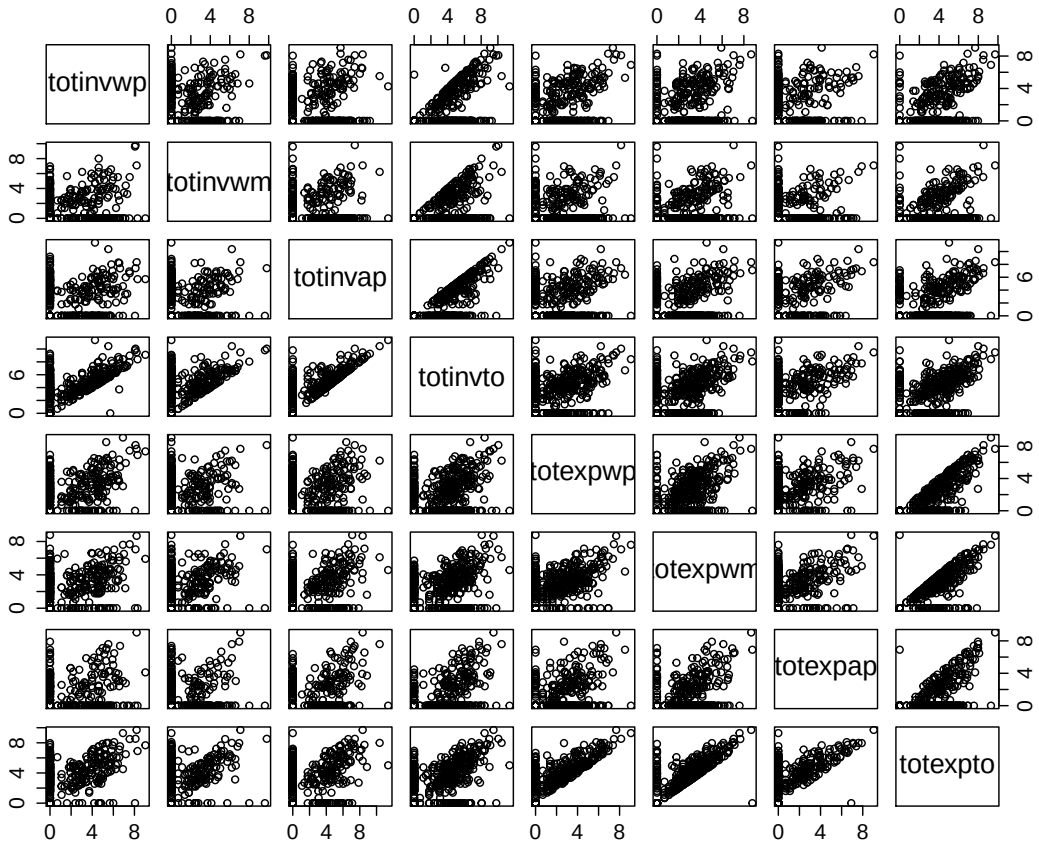

Figure 1: `pairs()` plot of the logarithmized `sepe` data.

4. Applying the methods

After running an outlier detection algorithm, a suitable imputation method should fill in the missing data to allow for the analysis of a "clean" and complete data set. The **modi** package contains four detection and three imputation functions. Detection with `BEM()`, `TRC()` and `GIMCD()` is based on the assumption of multivariate normality (of the bulk of the data) and thus we might want to use the same assumption for the imputation. Then `Winsimp()` or `POEM()` would be suitable choices, and here we stick to `Winsimp()`. After detecting outliers by `EAdet()`, it seems logical to use `EAimp()` for the imputation. In the following, these combinations are tested on the `sepe` data. The nearest neighbour imputation function `POEM()` is suitable with all detection algorithms since it combines local and global features. Nevertheless, it is not part of the present analysis, but `POEM()` has been extensively applied and tested in Charlton (2003).

4.1. `BEM()` – BACON-EEM

Outlier detection by `BEM()`

The BACON-EEM algorithm, developed in Béguin and Hulliger (2008), is implemented in the function `BEM()`. The BACON-EEM algorithm starts with a relatively small subset of good observations, i.e. observations which are not outliers. The mean and the covariance matrix of the good observations are estimated with the EM-algorithm. The estimates in the EM-algorithm must take into account the sample design, which gives reason for the name BACON-EEM. Then the Mahalonobis distance (MD) of all data points is calculated and the observations which are below the cutpoint defined through a χ^2-quantile are the new

good subset. The algorithm iterates further until convergence. The most important control parameters in `BEM()` are the size of the initial good subset as a multiple `c0` of the dimension of the data, and the probability `alpha` to determine the quantile of the χ^2-distribution for the cutpoint. Running `BEM()` on `sepe` with its default values for the starting subset fails, since the covariance-matrix used to calculate the MD is singular. The reason is that the starting subset for the algorithm is too small and contains too many missing values.

```
> BEM(data, sepe$weight)
Warning: missing observations 193 244 301 374 375 546 559
564 618 619 661 removed from the data

 Error in qr.default(EM.var.good) :
  NA/NaN/Inf in foreign function call (arg 1)
```

A remedy is to set the zeros in the data set to missing values:

```
> sepenozero <- recode(sepe, "0=NA")
> datanozero <- log(sepenozero[, sepevar8] + 1)
```

However, `BEM()` returns the identical error as before even though it removes all data with completely missing observations. When increasing the initial subset size by setting `c0=5`, the algorithm works.

```
> BEM(datanozero, sepe$weight, c0 = 5)
Warning: missing observations 2 4 11 12 41 57 ...
... 655 656 657 659 661 662 665 666 removed from the data

 BEM has detected 385 outlier(s) in 1.92 seconds.
```

The function `BEM()` returns a list whose first component `output` contains the summary information. Only 517 out of the 675 observations of `sepe` can be used. Hence, 158 observations were dropped due to complete missingness. The initial subset size contains 40 observations (`c0 * ncol(data)`). The algorithm used 1.92 seconds for the computation[1]. The cutpoint of 21.16 is the minimum (squared) MD of the observations which are considered outliers. In general, the cutpoint is the (squared) MD which is above $c_{Npr} \cdot \chi^2_{p,1-\alpha}$, where the constant c_{Npr} is approximately 1 and α is a small proportion $0 < \alpha < 0.5$ with default `alpha=0.01` (Béguin and Hulliger 2008, p. 94). All observations with a (squared) MD larger or equal than the cutpoint are declared outliers. Before discussing this parameter more specifically, a reflection on the plausibility of the detected outliers is appropriate. Almost 75% of the observations (namely 385 out of 517) have been declared as outlying. The reliability of this result is highly questionable. This reveals an issue with large data sets: the choice of the tuning parameter `alpha`.

Alpha

The tuning parameter α (argument `alpha` of the function `BEM()`) represents a probability, indicating the level $1 - \alpha$ of the cut-off quantile based on a χ^2_p distribution for good observations. If the analysis does return unreasonable amounts of outliers, one should decrease α. A rule of thumb is to divide a conventional value of α like the default $\alpha = 0.01$ by the number of observations (`alpha=0.01/nrow(data)`) for sample sizes above 100 observations. This strategy returns plausible results: Among the 517 observations which are usable for the analysis, 89 outliers have been detected in 1.37 seconds.

[1] All computations have been executed on a 2.90GHz Intel Core i7 CPU with 8.00 GB RAM.

```
> BEM.r <- BEM(datanozero, sepe$weight, c0 = 5, alpha = 0.01/nrow(datanozero))
> PlotMD(BEM.r$dist, ncol(datanozero), alpha = 0.95)
```

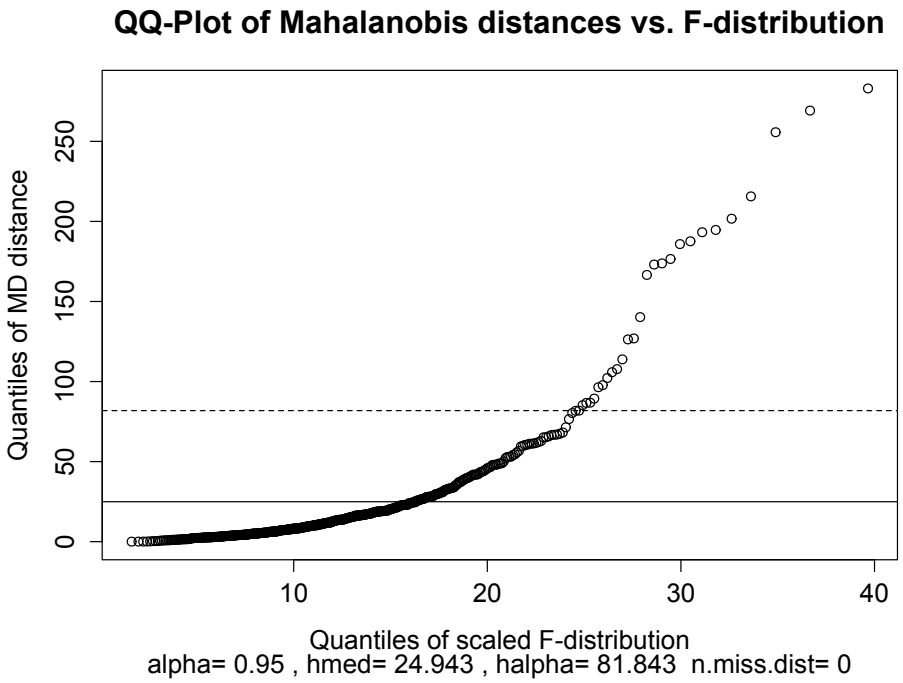

Figure 2: `BEM()` – QQ-plot of the Mahalanobis distances.

Cutpoint

The corresponding QQ-Plot in Figure 2 created by `PlotMD()` compares the (squared) MD with the F-distribution as proposed in (Little and Smith 1987). The Figure is practically the same (including the values) for the χ^2 distribution which can be chosen by setting the argument `chisquare=TRUE`. The default cutpoint is the minimal (squared) MD of the outliers. The value 37.14 of the default cutpoint is not appropriate and a higher value would fit the data better. There are two alternatives to set a more appropriate cutpoint. One can a) try to identify a substantial change in the distribution plot of the MD, or b) define a certain fraction of the data to be outlying. A good cutpoint in Figure 2 seems to be around 70 because there is a distinct upwards bend at that point. Implementing this cutpoint leads to a reduction of the number of outliers from 89 to 31.

```
> outind <- (ifelse(BEM.r$dist > 70, 1, 0))
> outind <- as.logical(recode(outind, "NA=0"))
> sum(outind)
[1] 31
```

Choosing option b), the default cutpoint is replaced by declaring a specific fraction of the observations with highest Mahalanobis distance as outliers. To preserve comparability between different algorithms, 5% of the total number of observations are declared outliers. Assuming that the fraction of outliers is identical in missing and non-missing data, the 5% is a fraction of the total number of usable observations. For `BEM()`, the number of usable observations is 517 and thus 26 observations are declared outliers. The cutpoint is 82.53.

```
> (cutpoint5 <- quantile(BEM.r$dist, 0.95, na.rm = TRUE))
     95%
82.52671
```

```
> outind5 <- (BEM.r$dist > cutpoint5)
> outind5 <- as.logical(recode(outind5, "NA=0"))
> sum(outind5)
[1] 26
```

To get an idea of how the algorithm selected the outliers and what effect a different cutpoint might have, Figure 3 shows total expenditure against total investment. Note that all zeros have been set to missing values during the detection step, but now have been re-inserted for the plot. In Figure 3, the good items are marked as black circles and the 89 outliers detected while using the default values with red crosses. The blue rectangles represent the 31 outliers determined by the visual inspection of the QQ-Plot (cutpoint = 70). The 5% most extreme points are a further subset of the blue rectangles. Naturally, this two-dimensional plot is not fully adequate for multivariate analysis. This becomes apparent in Figure 3, where increasing the cutpoint does declare points to the upper-right as good points which would rather be outliers in the two-dimensional plot.

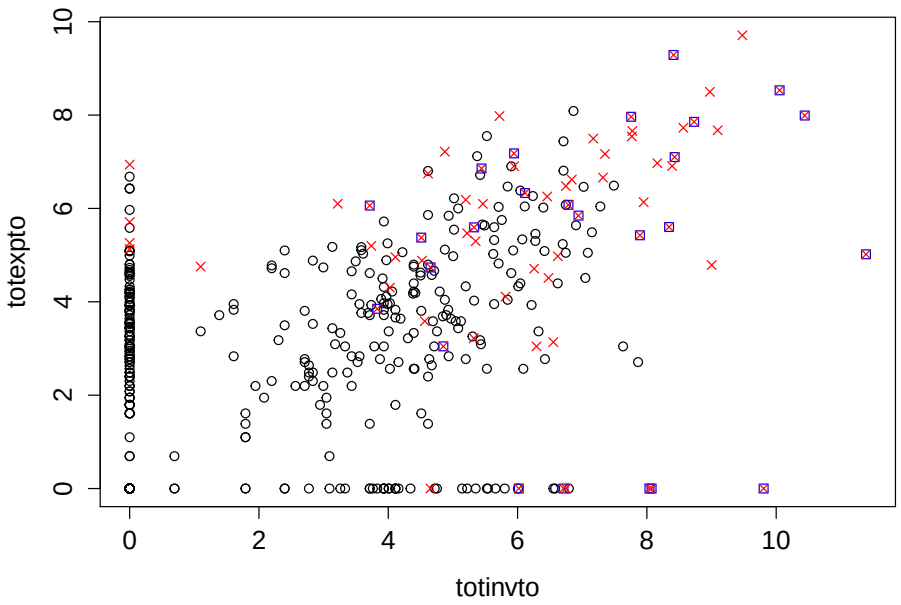

Figure 3: `BEM()` – Plot of total expenditures and investments. Outliers are marked as × (default cutpoint=37.14) and as □ (visually chosen cutpoint=70); good observations are marked as ○.

Imputation by `Winsimp()`

For imputation the visually determined cutpoint of 70 is applied. The next issue to be solved is at what stage the initial zeros should be re-inserted, as described in Section 3. There are two possibilities: re-insertion of the zeros before or after the imputation. There are arguments for both options. Re-inserting zeros before imputation yields imputations which are overall more coherent with multivariate normality since the multivariate normal relation among the variables is maintained also for those observations with 0 values. Re-insertion of zeros after imputation takes into account that the zeros did not contribute to the multivariate normal model implicitly used for the detection. However, the multivariate normal relation between the variables is broken. Both options are tested here. The imputed data sets are saved in the output object `Winsimp.r$imputed.data`. Unfortunately, the imputation gives negative values in expenditure and investment data. Since we assume those to be strictly positive, we censor the negative values to zero.

```
> # re-inserting zeros before imputation
> Winsimp.r <- Winsimp(data, BEM.r$output$center,
+                      BEM.r$output$scatter, outind)
> Winsimp.r$imputed.data <- ifelse(Winsimp.r$imputed.data < 0, 0,
+                                  Winsimp.r$imputed.data)
> zeros_before <- Winsimp.r$imputed.data

> # re-inserting zeros after imputation
> Winsimp.r <- Winsimp(datanozero, BEM.r$output$center,
+                      BEM.r$output$scatter, outind)
> zeros_after <- Winsimp.r$imputed.data
> zeros <- (ifelse(sepe[, sepevar8] == 0, 1, 0))
> zeros <- recode(zeros,  "NA=0")
> zeros_after <- ifelse(zeros == 1, 0, zeros_after)
```

Note that function `Winsimp()` does not recalculate the center and scatter of the data but uses the results of `BEM()`. The resulting weighted means of the eight variables and the corresponding determinant of the covariance-matrix are shown in Table 3. Means and covariance-matrices are all weighted by the sampling design. For calculations with the original (raw) data, missing values are left out list-wise (column 'Original' in Table 3). Calculations with the non-robust EM-algorithm are also shown (column 'Normal'). At which stage the zeros are re-inserted seems to play a significant role for the final data set, see the last two columns of the table. Re-insertion after imputation yields 63 negative values compared to 105 when re-insertion is before imputation. Negative values are replaced by zero. It is difficult to judge the shift of the means towards higher values after outlier detection and imputation (see Table 3). Also the change of the determinant of the covariance matrix does not allow a neat conclusion. It seems that re-insertion before imputation yields a very compact covariance matrix. Re-insertion of the zeros after imputation does inflate the scatter more, but still less than with the non-robust EM-imputation with package **norm** (Novo and Schafer 2013). The small determinant for re-inserted zeros before imputation shows evidence that extreme observations have been declared as outliers. The fact that the determinant for re-insertion after imputation is smaller than with **norm** shows that the robust imputation still yields a more compact determinant, which is less influenced by outliers.

Table 3: `BEM()` – Means and determinant of covariance matrix for original, normally imputed and robustly imputed data after re-insertion of zeros in different steps.

	Original	Normal	Before	After
`totinvwp`	0.71	0.73	0.78	0.87
`totinvwm`	0.47	0.56	0.72	0.69
`totinvap`	0.88	1.00	1.04	1.12
`totinvto`	1.51	1.81	1.69	1.88
`totexpwp`	0.99	1.05	1.04	1.11
`totexpwm`	1.53	1.62	1.48	1.61
`totexpap`	0.48	0.47	0.45	0.51
`totexpto`	2.01	2.12	1.96	2.19
Determinant	4.86	16.01	4.22	10.75

Notes: 'Original' refers to the original logarithmized data. For the calculations missing values are removed list-wise. 'Normal' refers to data imputed with the (non-robust) EM-algorithm. 'Before' and 'After' refers to re-insertion of zeros before or after imputation with the robust method `Winsimp()`. Means and determinants of the covariance-matrix are calculated with the package **survey** (Lumley 2004, 2014) and are hence weighted for the sample design.

4.2. `TRC()` – Transformed Rank Correlation

Outlier detection by `TRC()`

The algorithm of the transformed rank correlation (TRC), which is described in detail in Béguin and Hulliger (2004), is implemented in the function `TRC()`. The initial covariance matrix is calculated with Spearman-correlations which are standardised to be consistent at the multivariate normal distribution. The TRC algorithm uses an orthogonal transformation of the data into the space of eigenvectors, where the center and scatter is recalculated with robust univariate estimators (medians and median absolute deviations). For this transformation, complete data is needed and the TRC algorithm uses a provisional imputation of missing values based on the best simple robust regression available in the data. This provisional imputation is discarded later on, when the Mahalanobis distances are calculated. Nevertheless it is a critical step because contrary to the EM-algorithm, TRC uses just one—in principal the best—predictor for each variable. Thus important control parameters for the function `TRC()` are the minimal proportion `gamma` of the observations needed to determine an imputation model, and the minimal correlation `mincorr` needed to use a regressor in the provisional imputation. The first trial with `TRC()` uses the original data set (log-scale) and the default values. The algorithm returns the following warning:

```
> TRC.r <- TRC(data, sepe$weight)
Warning: missing observations 193 244 301 374 375
546 559 564 618 619 661 removed from the data

 Number of missing items:  589 , percentage of missing items:  0.110881
Some mads are 0. Using 0.75 quantile absolute deviations!
The following variable(s) have 0.75 quantile absolute deviations equal to 0 :
1 2 7
Error in TRC(data, sepe$weight) :
  Remove these variables or increase the quantile probability
```

To run the algorithm with all variables, either the probability quantile must be set to at least 0.81 or the zeros must be set to missing values. The second option is preferable, since increasing the quantile would not solve the problem that TRC is based on the assumption of multivariate normal data, which is not the case when the zeros are left in the data. A run of the algorithm with its default values and `monitor=TRUE` shows all computational steps:

```
> TRC(data = datanozero, weight = sepe$weight, monitor = TRUE)
Warning: missing observations 2 4 11 12 ...
638 644 646 648 655 656 657 659 661 662 665 666 removed from the data

 Number of missing items:  2008 , percentage of missing items:  0.4854932
End of preprocessing in 0 seconds
Computing Spearman Rank Correlations :
...
Spearman Rank Correlations (truncated and standardized):
          [,1]         [,2]         [,3]      [,4]
[1,] 1.0000000  0.133597112  0.633970052 0.8621370
...
[8,] 0.8640993 0.8396824 0.8141143 1.0000000
End of Spearman rank correlations estimations in 0.14 seconds
Regressors correlations
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1.110223e-16 0
...
 0 0 0 0 0 0 0 0 0 0 0.8396824 0 -1.110223e-16
Variable 1 :
  68 obs imputed using regressor 2 (cor= 0.1335971 slope= 0 intercept= 3.258097 )
  80 obs imputed using regressor 3 (cor= 0.6339701 slope= 0 intercept= 3.258097 )
```

```
...
Variable 8 :
  48 obs imputed using regressor 6 (cor= 0.8396824 slope= 0.812663 intercept= 1.173297 )
...
  3 obs imputed using regressor 7 (cor= 0.8141143 slope= 0 intercept= 3.044522 )
End of imputation in 0.03 seconds

 TRC has detected 147 outlier(s) in 0.19 seconds.
```

Gamma

The monitored values of the Spearman rank correlations (`cor`) seem reasonable, but the simple robust regressions for the imputation of missing values to construct a positive semi-definite covariance-matrix have a slope of 0 (`slope=0`). This is a consequence of the large number of missing values and the default requirement that at least half of the observations must determine the correlation in order to be used for imputation (`gamma=0.5`). To lower this requirement the tuning parameter `gamma` may be set to a lower value, e.g. assuming that a sufficient number to run the regression is 30 observations.

```
> TRC.r <- TRC(data = datanozero, weight = sepe$weight,
+              monitor = TRUE,  gamma = 30/TRC.r$output$sample.size)
> PlotMD(TRC.r$dist, ncol(datanozero))
```

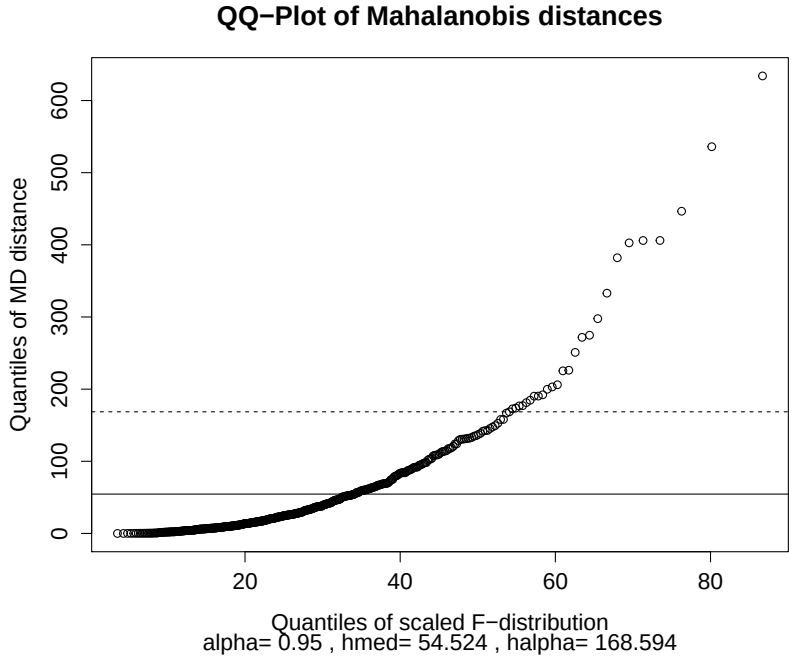

Figure 4: `TRC()` – QQ-plot of the Mahalanobis distances.

Now the imputation regression runs smoothly with non-zero slopes and the outlier detection works without a warning. `TRC()` drops 158 observations due to complete missingness. Out of the remaining 517 observations, 146 are declared as outliers. 48.5% of the items are missing values. The computation time is 0.18 seconds. The default cutpoint is

$$\text{median}(MD) \cdot F^{-1}(1-\alpha, p, n-p)/F^{-1}(0.5, p, n-p), \tag{1}$$

where F^{-1} is the quantile function of the F-distribution. In this case, the default cutpoint is 54.52. The QQ-plot in Figure 4 created by `PlotMD()` shows the (squared) MD at the

corresponding F-distribution. It is easily seen that the proposed cutpoint of 54.52 is not appropriate and a higher value would fit the data better.

Cutpoint

An appropriate cutpoint is determined by inspection of Figure 4. The first upward-jump visible in the QQ-plot is at about 210. Setting all observations with an MD above 210 to outliers returns 14 outliers compared to 146 by default. Defining 5% of the data to be outlying returns the same number of outliers (26) as BACON-EEM. However, even if the same number of outliers is identified, the individual outliers may be different (see Table 7).

Imputation by `Winsimp()`

For the imputation of the TRC-detected outliers, function `Winsimp()` is used. As already discussed in section 4.1.4, an issue is at what stage we re-insert the zeros. Imputed negative values are censored to zero. Re-insertion after imputation yields 44 and re-insertion before imputation 123 negative values. The resulting means and determinants of the original and imputed data are shown in Table 4. Means differ between the re-insertion methods and tend to be higher in the imputed data compared to the original. The deviation is statistically not significant. Since the determinant of the covariance-matrix is smaller than for the original data with re-insertion of the zeros before imputation, it seems that that outliers have successfully been detected and imputed to a more appropriate value. However, the determinant is inflated considerably if the zeros are re-inserted after imputation. Therefore, re-insertion of zeros before imputation is preferable.

Table 4: `TRC()` – Means and determinant of covariance matrix for original and imputed data after re-insertion of zeros in different steps.

	Original	Normal	Before	After
`totinvwp`	0.71	0.73	0.78	0.87
`totinvwm`	0.47	0.56	0.72	0.69
`totinvap`	0.88	1.00	1.04	1.12
`totinvto`	1.51	1.81	1.69	1.88
`totexpwp`	0.99	1.05	1.04	1.11
`totexpwm`	1.53	1.62	1.48	1.61
`totexpap`	0.48	0.47	0.45	0.51
`totexpto`	2.01	2.12	1.96	2.19
Determinant	4.86	16.01	6.33	12.73

Notes: See Table 3.

4.3. `GIMCD()` – Gaussian Imputation and Minimum Covariance Determinant

Outlier detection by `GIMCD()`

The GIMCD algorithm (Béguin and Hulliger 2008) is implemented in function `GIMCD()`. The GIMCD algorithm first uses a non-robust Gaussian imputation, followed by a highly robust minimum covariance determinant (MCD) algorithm to detect outliers. The only control parameter is the probability α to determine the quantile for the cutpoint. Compared to the other discussed algorithms, weights cannot be used with GIMCD since at the moment there is no implementation of MCD available which takes weights into account. At the first attempt with the `sepe` data, the algorithm fails to finish. Again the problem is the large number of zeros which result in a singular covariance matrix.

```
> GIMCD(data, seedem = 234567819, seedmcd = 4097)
 Error in solve.default(cov, ...) :
  Lapack routine dgesv: system is exactly singular: U[2,2] = 0
```

Recoding the zeros to missing values enables the algorithm to compute all needed parameters. The parameter `alpha` determines a threshold value for the cut-off for the outlier Mahalanobis distances. This cutpoint is the same as (1). The larger the value of `alpha`, the more outliers will be detected. The default value `alpha=0.05` seems reasonable for the `sepe` data. The parameter `seedem` sets a starting value for the random number generator used in the first step for the Gaussian imputation with the EM-algorithm, which is executed by the **norm** package (Novo and Schafer 2013). The default is `seedem=234567819`. Since in the second step the MCD again uses a random number generator, the seed of MCD can be set separately with `seedmcd`. The default value for MCD is taken from the system and varies for each run if not set explicitly.

```
> GIMCD.r <- GIMCD(datanozero, seedem = 234567819, seedmcd = 4097)
GIMCD has detected 57 outliers in 1.46 seconds.
> PlotMD(GIMCD.r$dist, ncol(datanozero))
```

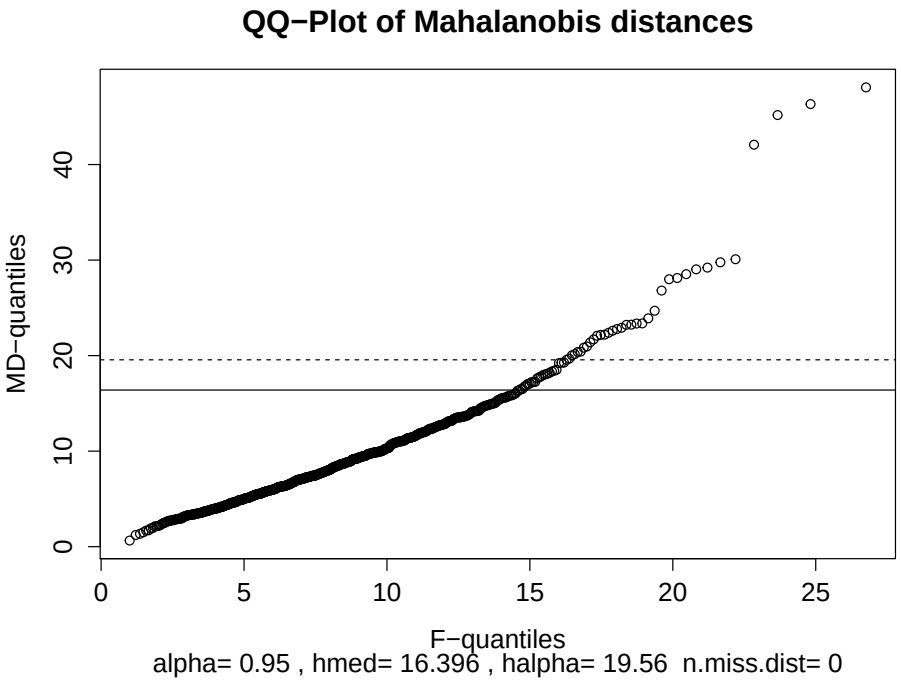

Figure 5: `GIMCD()` – QQ-plot of the Mahalanobis distances.

Cutpoint

The default cutpoint is defined according to (1) and evaluates to 16.4 for the `sepe` data. Visual inspection of the QQ-plot in Figure 5 favours a slightly different cutpoint, namely at 24. With this new cutpoint, only 13 outliers are detected. Cutting out 5% outliers yields 34 outlying data points. Note that `GIMCD()` also imputes values for the completely missing observations, unlike `BEM()` and `TRC()`.

Imputation by `Winsimp()`

For the imputation the visually identified cutpoint of 24 is used. The imputation can be computed without any problems with function `Winsimp()`. Re-insertion before imputation

yields 49 negative values and re-insertion after yields 97 negative values, which are recoded to 0. A comparison of the results with re-insertion of the zeros before and after the imputation is given in Table 5. When re-inserting the zeros after imputation, the determinant is inflated more than when using the non-robust normal imputation. The means are shifted towards higher values than for the other methods. Here the lack of weighting may have had its impact.

Table 5: `GIMCD()` – Means and determinant of covariance matrix for original and imputed data after re-insertion of zeros in different steps.

	Original	Normal	Before	After
`totinvwp`	0.71	0.73	0.83	0.91
`totinvwm`	0.47	0.56	0.72	0.71
`totinvap`	0.88	1.00	1.08	1.15
`totinvto`	1.51	1.81	1.77	1.94
`totexpwp`	0.99	1.05	1.04	1.18
`totexpwm`	1.53	1.62	1.60	1.72
`totexpap`	0.48	0.47	0.47	0.55
`totexpto`	2.01	2.12	2.04	2.31
Determinant	4.86	16.01	6.47	20.81

Notes: See Table 3.

4.4. EA – Epidemic Algorithm

Outlier detection with `EAdet()`

The Epidemic Algorithm (EA) is implemented for detection in function `EAdet()` and for imputation in function `EAimp()`. Compared to the previous methods, the Epidemic Algorithm has no underlying distributional assumption. It is based on distance measures, which are described in detail in Béguin and Hulliger (2004). The basic idea of the Epidemic Algorithm is to simulate an epidemic which starts at a central point, actually a spatial median, and then infects points in the neighbourhood of the infected ones in a stepwise manner. The last infected points are nominated outliers. The exact dependence of the infection probability from the distance is determined by several functional forms with different tuning parameters. The default cutpoint is set at time $t = \text{median}(t, w) + 3 \cdot \text{MAD}(t, w)$, where the median and the median absolute deviation (MAD) are both weighted. The cutpoint may be adjusted after the calculation of the infection times by `EAdet()`. The detection function `EAdet()` can cope with the original data set in spite of the 48% items with value zero. The algorithm gives a warning because the median absolute deviation is replaced by the 90% quantile of absolute deviations, but the results can be calculated nonetheless. All points are infected within 0.38 seconds. For the `sepe` data the cutpoint results in only 7 outliers. However, there is a further set of 11 points which have never been infected. This may happen because they have too many missing values or because they are too far outlying. The function value `outind` is a logical vector with `TRUE` for the outliers (late infected), `FALSE` for the good observations (early infected) and `NA` for the never infected observations. In the present case the 11 never infected observations consist entirely of missing values.

```
> EAdet.r <- EAdet(data, sepe$weight)

 Some mads are 0. Standardizing with  0.9  quantile absolute deviations!
 EA detection has finished with 664 infected points in 0.38 seconds.
```

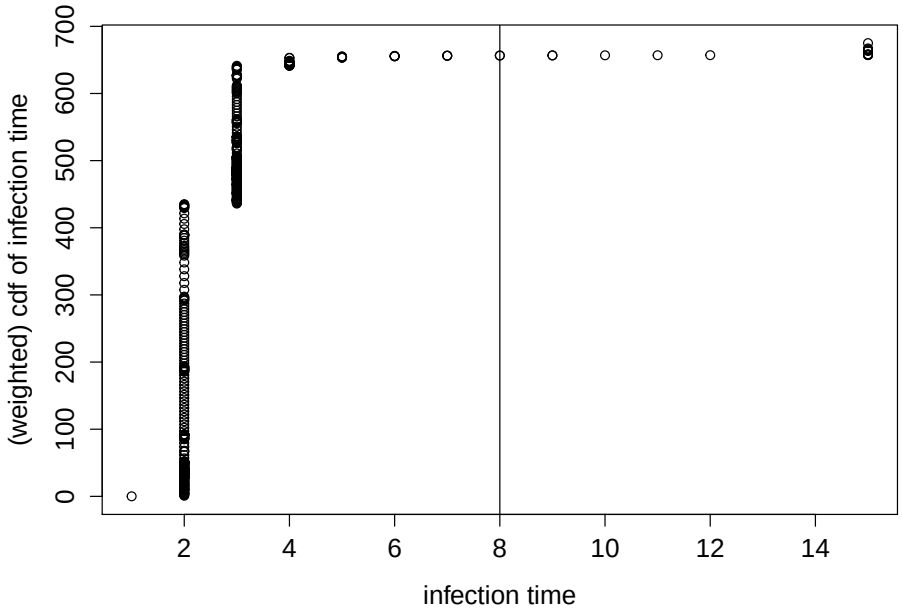

Figure 6: EA – Infection times and default cutpoint.

By default, the (weighted) cumulative distribution function of the infection times is plotted, see Figure 6. `EAdet()` detects less outliers than the other algorithms with the default cutpoint. Some tuning might be necessary even if the algorithm works with default parameters. Setting the parameter `monitor=TRUE` shows the exact number of infected observations at every step. Starting from the spatial median, which for `sepe` is a point with zero investments and only missing values in expenditures, the algorithm infects more than one third of the points in the first step. In the second step, already over 90% of the points are infected. Closer inspection shows that the Epidemic Algorithm suffers from the many zeros and missing values because the inter-point distances are calculated on the basis of the jointly observed values only. As a result, the infection probabilities do not discriminate enough between observations to be infected next, and the epidemic stops after 12 steps, i.e. at infection time 12. Also the transformation of the data has its influence on the duration of the epidemic. Points which should not be declared outliers might catch the epidemic when the transformation over-corrects the skewness of the data. It is of course possible to discard completely missing observations (you may set the parameter `rm.missing=TRUE`) and it is also possible to set zeros to missing before running `EAdet()`. Since the algorithm worked nevertheless we may use it as is and judge later on about the result.

Cutpoint

Also with `EAdet()` setting a good cutpoint needs some care. The default outlier rule declares observations which are larger or equal to 8 as outliers. Looking at the (weighted) cumulative distribution function of the infection times in Figure 6, it seems more reasonable to set the cutpoint to 5:

```
> outind <- EAdet.r$infection.time >= 5
> sum(outind, na.rm = TRUE)
[1] 20
> sum(is.na(outind))
[1] 11
```

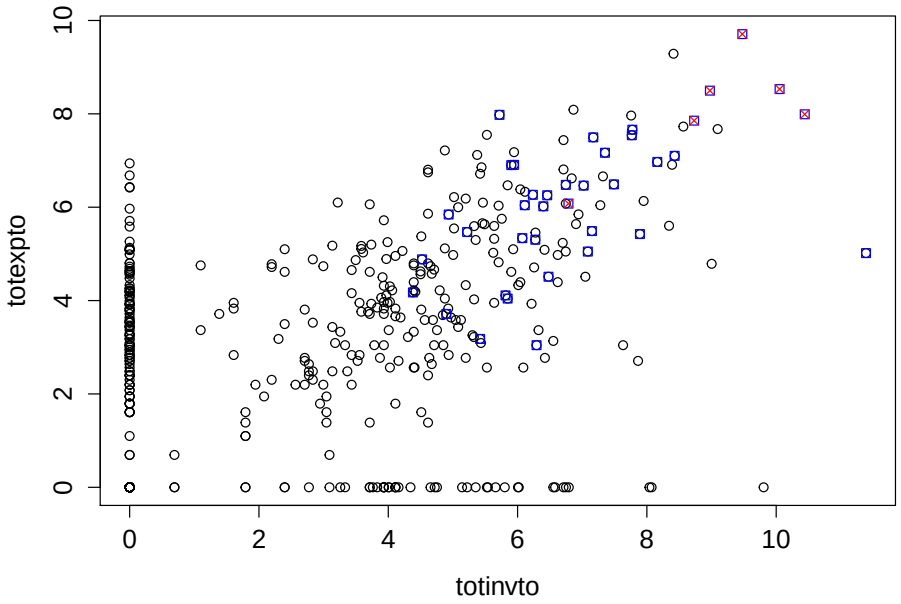

Figure 7: EA – Outliers with default cutpoint (red crosses) and visually set cutpoint (blue rectangles).

Using the cutpoint 5 yields 20 outliers. The corresponding plot in Figure 7 shows a different picture than we know from the other algorithms. First of all, there are substantially less outliers. Second, no points on the axes are outlying, and third, the point with largest `totinvto`—which the other algorithms declared an outlier—is not an outlier. `EAdet()` seems to be able to handle the multivariate outlier problem, because most outliers are not clearly outlying in one dimension. However, `EAdet()` has problems with outliers that contain many zeros.

Imputation by `EAimp()`

For imputation after detection with `EAdet()` and with the visually chosen cutpoint, the function `EAimp()` is used. Since the zeros were not set to missing, the re-insertion is not an issue now. `EAimp()` uses the distances calculated in `EAdet()` and starts an epidemic at each observation to be imputed until donors for the missing values are infected. Then a donor is selected randomly. The imputation uses the visually chosen cutpoint 5 and takes 2.76 seconds. Table 6 contains a comparison between the original, normally imputed and EA-imputed center and scatter measures. Most of the means of EA-imputed data are between the original and the normally-imputed means. Using the visually chosen cutpoint, the determinant of the imputed data is inflated but far from the determinant of the normally imputed data.

```
> EAimp.r <- EAimp(data, weights = sepe$weight, outind = EAdet.r$outind,
+                  duration = EAdet.r$output$duration)

Missing values in outlier indicator set to FALSE.

 Dimensions (n,p): 675 8
 Number of complete records  449
 Number of records with maximum p/2 variables missing  633
 Number of imputands is  230
 Reach for imputation is  max
```

```
Number of remaining missing values is  0
```

Table 6: EA – Means and determinant of covariance-matrix for original and imputed data.

	Original	Normal	EA
totinvwp	0.71	0.73	0.73
totinvwm	0.47	0.56	0.55
totinvap	0.88	1.00	0.97
totinvto	1.51	1.81	1.78
totexpwp	0.99	1.05	1.03
totexpwm	1.53	1.62	1.49
totexpap	0.48	0.47	0.42
totexpto	2.01	2.12	2.00
Determinant	4.86	16.01	10.69

Notes: See Table 3.

5. Conclusion

Multivariate outlier detection starts before running outlier detection algorithms such as the ones implemented in the **modi** package. Every data set has its unique issues which need to be solved before detection. Balance rules, missing value patterns as well as distributions have to be checked. E.g. the `sepe` data set has a zero inflated distribution and hence needs to be prepared to satisfy the distributional assumptions of the parametric algorithms. When the assumptions are satisfied, the parameters of the outlier detection function need to be chosen. Even though the algorithms in package **modi** have a high power in detecting multivariate outliers, user-intervention to choose the cutpoint is necessary. Checking of the imputed data is also necessary, e.g. to censor imputed data to positive values.

Choosing an appropriate method is difficult since all presented methods have advantages and disadvantages. Table 7 shows an individual comparison of the detected outliers. The number of outliers is approximately 5% (26, 26, 34, 49 for BACON-EEM, TRC, GIMCD, EA respectively). Only 5 points are selected by all four methods. The parametric methods (BACON-EEM, TRC, GICMD) have further 9 outliers in common. In terms of the Jaccard-distances between the outlier sets, BACON-EEM and TRC are closest, GIMCD somewhat detached from BACON-EEM and TRC, and EA has a large distance to all three parametric methods.

Table 8 summarizes the results of the four algorithms illustrated in the paper. The number of outliers detected by default varies strongly between the methods. However, if the cutpoint is chosen by the user, the different algorithms yield more similar numbers of outliers. Still we have to keep in mind that even though the absolute number of outliers is similar, the effectively detected points are not (see Table 7).

When applying the functions in **modi** to bigger data sets, computation time becomes a more and more crucial criteria. The fastest algorithm is TRC with only 0.28 seconds for detection and imputation in this example. EA is the slowest, since the imputation uses an epidemic for each outlier, which slows down the computational speed.

Looking at the determinants of the covariance-matrices lets us favour to re-insert the zeros before we run the imputation algorithms for the parametric methods. Determinants are smaller when zeros are re-inserted before imputation compared to the re-insertion after imputation. In any case, all four methods (except GIMCD with re-insertion after imputation) have a

Table 7: Outliers according to the four methods.

ID	BACON-EEM	TRC	GIMCD	EA	C	ID	BACON-EEM	TRC	GIMCD	EA	C
3	0	0	1	0	1	330	1	1	1	1	4
6	1	0	0	0	1	333	0	0	0	1	1
13	0	1	0	0	1	340	1	1	1	0	3
14	0	1	0	0	1	341	0	0	0	1	1
18	0	0	0	1	1	344	0	0	0	1	1
21	1	1	0	0	2	380	0	0	0	1	1
25	0	0	0	1	1	381	0	0	0	1	1
31	1	1	1	0	3	382	0	1	0	0	1
38	0	0	1	0	1	383	0	0	0	1	1
59	0	1	0	0	1	391	1	1	1	0	3
68	0	0	1	0	1	396	0	0	0	1	1
78	0	0	1	0	1	406	0	0	0	1	1
91	0	0	0	1	1	408	0	0	0	1	1
93	0	0	0	1	1	414	0	0	0	1	1
101	0	0	0	1	1	421	0	0	0	1	1
102	0	0	0	1	1	424	0	0	0	1	1
128	0	0	0	1	1	425	1	0	1	0	2
133	1	1	1	0	3	431	1	0	0	1	2
134	0	1	1	1	3	437	0	0	1	0	1
137	0	0	1	0	1	439	0	0	1	0	1
154	0	0	0	1	1	441	1	0	0	1	2
156	0	0	0	1	1	448	1	1	1	0	3
157	0	0	0	1	1	449	0	0	1	1	2
165	1	1	1	1	4	456	0	0	0	1	1
173	1	1	0	0	2	459	0	0	0	1	1
178	0	0	1	0	1	460	0	0	0	1	1
186	1	1	1	1	4	461	1	1	1	1	4
192	0	1	0	1	2	468	0	1	1	0	2
194	1	1	1	1	4	471	0	0	1	0	1
200	0	0	0	1	1	475	0	0	0	1	1
206	0	0	0	1	1	480	1	0	0	0	1
238	0	0	1	0	1	484	0	0	0	1	1
241	0	0	0	1	1	485	0	0	0	1	1
257	0	0	0	1	1	489	1	0	0	0	1
265	0	0	1	0	1	491	0	0	0	1	1
267	0	0	1	0	1	517	1	0	0	0	1
273	1	1	1	0	3	555	0	1	0	0	1
282	1	1	1	0	3	561	0	0	0	1	1
288	0	0	0	1	1	587	0	0	0	1	1
291	0	0	1	0	1	624	0	0	1	0	1
307	1	1	0	0	2	626	1	0	0	0	1
309	1	1	1	1	4	641	0	0	0	1	1
324	1	1	1	0	3	648	0	0	1	0	1
328	0	0	0	1	1	654	1	1	1	0	3

Notes: ID is the observation number. An entry 1 in the columns BACON-EEM to EA indicates that the corresponding method nominates the observation as an outlier. C is the number of coinciding methods.

Table 8: Summary of results for all functions.

Algorithm	No. Outliers		Time		Determinant	
	Default	Visual	Det.	Imp.	0's Before	0's After
`BEM()` – `Winsimp()`	89	31	1.4	0.07	4.22	10.75
`TRC()` - `Winsimp()`	146	14	0.23	0.05	6.33	12.73
`GIMCD()` - `Winsimp()`	57	20	1.48	0.05	6.47	20.81
`EAdet()` - `EAimp()`	7	20	0.36	2.54	10.69	

smaller determinant of the covariance matrix than when imputing in a non-robust way with the EM-algorithm. This is desirable since the outlying data points are normally far away of the center and blow up the determinant.

The distribution of the `sepe` data set is far from multivariate normal. Nevertheless, methods with an underlying assumption on multivariate normality may return usable results when the distribution is uni-modal apart from the zero-inflation, which must be treated explicitly. BACON-EEM and TRC extract sufficient information from the data to detect relevant outliers (they identify 14 identical outliers out of 26). GIMCD seems to be able to cope with the structure of the data in spite of not taking into account the survey weights. However the zeros must be re-inserted before imputation to preserve robustness. EA does not rely on the global structure of the data and does not need special treatment of the zero values. However, it needs careful tuning in order to differentiate outliers sufficiently well from good points.

Multivariate outliers are difficult to detect. If in addition missing values and zero-inflation occur, there are few algorithms that cope with such data. The presented algorithms are capable of doing this. However, their use is not straightforward and careful evaluation of the results is needed.

The package **modi** is currently available in Version 1.6 on R -Forge and will be submitted to CRAN (the Comprehensive R Archive Network, `http://CRAN.R-project.org`). The use of a deterministic MCD algorithm (function `covMcd()` in package **robustbase** (Rousseeuw, Croux, Todorov, Ruckstuhl, Salibian-Barrera, Verbeke, Koller, and Maechler 2014)) and a better tuning of the progression of the Epidemic Algorithm will be evaluated for the next version of the package.

Acknowledgement

The research for this paper was supported by the Swiss Federal Office for Education and Science through the European Union FP6 project EUREDIT, by the European Union FP7 project AMELI (EU FP7-SSH-2007-217322) and by FHNW School of Business. Thanks go to two anonymous referees and to the editors for their valuable comments.

References

Béguin C, Hulliger B (2004). "Multivariate Outlier Detection in Incomplete Survey Data: the Epidemic Algorithm and Transformed Rank Correlations." *Journal of the Royal Statistical Society, Series A: Statistics in Society*, **167**(2), 275–294.

Béguin C, Hulliger B (2008). "The BACON-EEM Algorithm for Multivariate Outlier Detection in Incomplete Survey Data." *Survey Methodology*, **34**(1), 91–103.

Campbell N (1989). "Bushfire Mapping Using NOAA AVHRR Data." *Technical report*, Commonwealth Scientific and Industrial Research Organisation.

Chambers R (1986). "Outlier Robust Finite Population Estimation." *Journal of the American Statistical Association*, **81**(396), 1063–1069.

Charlton J (ed.) (2003). *Towards Effective Statistical Editing and Imputation Strategies – Findings of the Euredit project*, volume 1 and 2. EUREDIT consortium. URL `http://www.cs.york.ac.uk/euredit/results/results.html`.

Hampel FR, Ronchetti EM, Rousseeuw PJ, Stahel WA (1986). *Robust Statistics: The Approach Based on Influence Functions*. John Wiley & Sons.

Hulliger B (2015). ***modi**: Multivariate Outlier Detection and Imputation for Incomplete Survey Data*. R package version 1.6, URL http://R-Forge.R-project.org/projects/modi/.

Hulliger B, Schoch T (2013). "Mechanisms for Multivariate Outliers and Missing Values." In *Proceedings of the NTTS 2013 Conference*. Brussels, Belgium.

Little R, Smith P (1987). "Editing and Imputation for Quantitative Survey Data." *Journal of the American Statistical Association*, **82**(397), 58–68.

Lumley T (2004). "Analysis of complex survey samples." *Journal of Statistical Software*, **9**(1), 1–19.

Lumley T (2014). ***survey**: Analysis of Complex Survey Samples*. R package version 3.30-3, URL http://CRAN.R-project.org/package=survey.

Luzi O, De Waal T, Hulliger B, Di Zio M, Pannekoek J, Kilchmann D, Guarnera U, Hoogland J, Manzari A, Tempelman C (2007). "Recommended Practices for Editing and Imputation in Cross-Sectional Business Surveys." *Technical report*, ISTAT, CBS, SFSO, Eurostat.

Maronna R, Zamar R (2002). "Robust Estimates of Location and Dispersion for High-Dimensional Datasets." *Technometrics*, **44**(4), 307–317.

Novo AA, Schafer JL (2013). ***norm**: Analysis of Multivariate Normal Datasets with Missing Values*. R package version 1.0-9.5, URL https://CRAN.R-project.org/package=norm.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Rousseeuw P, Croux C, Todorov V, Ruckstuhl A, Salibian-Barrera M, Verbeke T, Koller M, Maechler M (2014). ***robstbase**: Basic Robust Statistics*. R package version 0.91-1, URL http://CRAN.R-project.org/package=robustbase.

Rousseeuw PJ, Van Driessen K (1999). "A Fast Algorithm for the Minimum Covariance Determinant Estimator." *Technometrics*, **41**(3), 212–223.

Todorov V (2014a). ***rrcov**: Scalable Robust Estimators with High Breakdown Point*. R package version 1.3-8, URL http://CRAN.R-project.org/package=rrcov.

Todorov V (2014b). ***rrcovNA**: Scalable Robust Estimators with High Breakdown Point for Incomplete Data*. R package version 0.4-7, URL http://CRAN.R-project.org/package=rrcovNA.

Todorov V, Filzmoser P (2009). "An Object-Oriented Framework for Robust Multivariate Analysis." *Journal of Statistical Software*, **32**(3), 1–47.

Todorov V, Templ M, Filzmoser P (2011). "Detection of Multivariate Outliers in Business Survey Data with Incomplete Information." *Advances in Data Analysis and Classification*, **5**(1), 37–56.

Affiliation:

Beat Hulliger
FHNW School of Business
University of Northwestern Switzerland
4600 Olten, Switzerland
E-mail: beat.hulliger@fhnw.ch
URL: http://www.fhnw.ch/people/beat-hulliger

Austrian Journal of Statistics
published by the Austrian Society of Statistics
Volume 45
March 2016

http://www.ajs.or.at/
http://www.osg.or.at/

Submitted: 2014-10-30
Accepted: 2016-01-14