OCCASION

This publication has been made available to the public on the occasion of the 50th anniversary of the United Nations Industrial Development Organisation.

DISCLAIMER

FAIR USE POLICY

CONTACT

Please contact publications@unido.org for further information concerning UNIDO publications.

For more information about UNIDO, please visit us at www.unido.org

UNITED NATIONS INDUSTRIAL DEVELOPMENT ORGANIZATION
Vienna International Centre, P.O. Box 300, 1400 Vienna, Austria
Tel: (+43-1) 26026-0 · www.unido.org · unido@unido.org

WORKING PAPER 01/2012

R in the Statistical Office:
Part II

UNITED NATIONS
INDUSTRIAL DEVELOPMENT ORGANIZATION

DEVELOPMENT POLICY, STATISTICS AND RESEARCH BRANCH
WORKING PAPER 1/2012

R in the Statistical Office: Part II

Valentin Todorov
Development Policy, Statistics and Research Branch
Strategic Research, Quality Assurance and Advocacy Division
UNIDO

Matthias Templ
UNIDO Consultant

UNITED NATIONS INDUSTRIAL DEVELOPMENT ORGANIZATION
Vienna, 2012

Acknowledgements
This working paper has significantly benefitted from comments and suggestions made by the members of the Sharing Advisory Board (SAB) of the Conference of European Statisticians. We are extremely grateful to Niki Rodousakis for editing the paper, which has contributed much to its quality. The authors are solely responsible for any remaining errors.

Contents

List of Tables

List of Figures

1 Introduction

When it comes to data processing and data analysis in a national or international statistical organization several well established statistical software packages are readily available: (i) SAS due to its traditional position in these organizations (if the necessary funds are available), its ability to handle extremely large data sets and its availability on any platform including mainframes (Lowman, 2009); (ii) SPSS is considered *user friendly* because of its *point-and-click* interface (although still providing the so called *syntax*); (iii) STATA is likely the most cost-effective the most *cost-effective* among the three and by design is especially suitable for working with data generated from complex survey designs (as is the case in many NSOs). However, for flexibility in terms of reading, manipulating and writing data, availability of the most recent statistical methodology, versatile presentation capabilities for generating tables and graphics which can readily be used in word processing systems such as LATEX or or Word as well as an economical solution, R is the most suitable option.

Although R has powerful statistical analysis features, data manipulation tools and versatile presentation capabilities, it has not yet become an everyday statistical package at national and international statistical offices. This is mainly mainly attributable to the widespread to the widespread view that R is difficult to learn and has a very steep learning curve compared to the other statistical packages such as SAS, SPSS and STATA. The lack of a true point-and-click interface contributes to this, though several packages which provide graphical user interfaces for R are available. The technical documentation accompanying R and most of the add-on packages rarely include syntax examples related to the analytical methods applied in official statistics. Similarly, the user guides and other documents for official statistics and descriptions of publicly available data sets rarely provide appropriate code examples to R users.

A previous working paper published by UNIDO (Todorov, 2010) presents an overview of R, which focuses on the strengths of this statistical environment for the typical tasks performed in national and international statistical offices. It outlines some of the advantages of R using examples from the statistical production process of UNIDO where certain steps were either migrated or newly developed in R - the areas of data integration and automatic generation of publication quality graphics for the dissemination of statistical data. The abilities of R to import and export data from and to different data formats and different statistical systems are considered as well. These features render it a very useful tool for facilitating collaboration in statistical offices. The second example application considers the graphical excellence of R as applied in the statistical production process of UNIDO for generating publication quality graphics included in the *International Yearbook of Industrial Statistics*. The graphics together with the related text are typeset in LATEX using the R tool for dynamic reporting *Sweave*. The third example illustrates the analytical and modeling functions available in R and the

add-on packages. These are used to implement a nowcasting tool for Manufacturing Value Added (MVA) to generate estimates for UNIDO publications. Functions from the package for robust statistics **robustbase** are used for this purpose.

The present work continues along this line and presents three important areas of data processing and data analysis, typical for the activities of a national or international statistical office.

The first issue addressed is the possible incompleteness of the collected data and the treatment of the missing values which is covered in Section 2. *Missing data* has effect on the properties of the obtained from the data estimates (e.g., means, percentages, percentiles, variances, ratios, regression parameters, etc.). Missing data can also have an effect on inferences, i.e. the properties of tests and confidence intervals. One crucial criterion of these effects is the way in which the probability of an item to be missing depends on other observed or non-observed variables as well as on its own value, i.e. the *missingness mechanism*. After presenting the types of missing data in Section 2.1, we continue in Section 2.2 with a detailed discussion of how R handles missing data, including comparison to other well-known statistical packages. Different methods for the treatment of missing data (traditional and more advanced) are described and illustrated using R code. Section 4 presents useful tools for visualizing missing values and exploring the data as well as the structure of the missing values, available in the R package **VIM** (Templ and Alfons, 2011). Depending on the structure of the missing values, these tools may help to identify the mechanism generating the missing values. In the provided examples the R package **VIM** is applied to INDSTAT 2 edition 2010, the *UNIDO Industrial Statistics Database*.

In the next Section 3 the procedures of *statistical data editing* and the issue of *outlier detection* are discussed, together with the accompanying problems of missing data and their treatment. Data collected in surveys generally contain errors for a variety of reasons. For example, a respondent may give a wrong answer or errors might arise while processing the data. Statistical data editing, i.e. checks and corrections, are necessary to increase the quality of the data. First, erroneous values in the data set have to be localized, preferably in an automated manner. The localized erroneous values have to be replaced by reasonable values (imputation). It is not necessary to remove all errors from a data set in order to obtain reliable publication figures since the sampling error might be higher than that related to minor errors in the data. However, logical relationships in the data should be preserved. For example, the income components should sum up to the income or a 14 years old girl cannot be married and have three children. In addition to that, large non-representative outliers (measurement errors) have to be localized and replaced by reasonable estimates or by true values which may be obtained by contacting the corresponding person or enterprise. The detection of outliers, i.e. of atypical observations which deviate from the usual data variability, is very important

in statistical analysis as the application of classical statistical models to data including outliers can lead to misleading results. The multivariate aspect of the data collected in surveys makes the task of identifying outliers particularly challenging. The outliers can be completely hidden in one or two dimensional views of the data (Todorov et al, 2011). This demonstrates that univariate outlier detection methods are useless, although National Statistical Offices (NSO) prefer them because of their simplicity. The focus in the present work is on using robust methods to identify the outliers which can subsequently be treated using the traditional approach and in particular to present suitable R software tools.

The third topic discussed in Section 5 is *Statistical Disclosure Control (SDC)* which covers techniques that can be defined as a set of methods to reduce the risk of disclosing information on individuals, businesses or other organizations. Such methods are only related to the dissemination step and are usually based on restricting the amount of or modifying the data released. The SDC methods can be categorized according to two different concepts: (i) the anonymization of microdata, and (ii) the anonymization of cells in tables. The packages **sdcMicro** and **sdcTable** provide practitioners with tools that meet the requirements of statistical disclosure control in these two categories. A brief overview of the most popular methods for the protection of categorical and continuous key variables is presented and illustrated with the corresponding software implementation in the R package **sdcMicro**. Similarly, the protection of (hierarchical) tables is illustrated with the R package **sdcTable**.

An often criticized feature of R is how it handles data storage - all data used in calculations need to be held in memory. Furthermore, with data sets rapidly growing scalability should also be considered, i.e. if we can analyse a data set with traditional methods today, this analysis could fail tomorrow or in a year. One solution is to use modern hardware with a 64-bit address space and huge amounts of RAM and although such solutions could be implemented for many applications in official statistics, this issue continues to pose a challenge. The handling of large data sets in R is is a field that is actively developing and many packages which cover one or the other aspect of this issue already exist. A brief overview with some examples is provided in Section 6 and a more extensive review of methods for dealing with large data sets, and in particular for the analysis of official statistics data.

Most of the data sets discussed throughout the paper, which have mainly been derived from UNIDO databases as well as selections from other sources, and all the R code used are provided in an R package called **ritso** which accompanies the work.

There are a number of interesting topics worth considering with regard to the use of R in official statistics, but these are not covered by the current paper and will be covered in future work. A brief overview of these topics is presented in Section 7.

2 Missing data and imputation methods

Missing data are a common problem of data sets in almost any statistical analysis and survey data are not an exception. We use the term *missing data* to define data that are missing for some (but not all) variables and for some (but not all) cases. When data are missing for a variable for all cases, that particular variable is referred to as *latent* or *unobserved*. In a survey context this would be a variable which is not present in the questionnaire. When data are missing for all variables for a given case, we have what is known as *unit non-response*. In a survey context this would be an object (sample establishment) that did not complete and/or return the questionnaire at all. Neither latent variables nor unit non-response will be considered in this paper. Missing data (missing values for certain variables for certain cases, i.e. when some of the questions in the questionnaire are left unanswered) are also referred to as *item non-response*.

If the missing values are *missing completely at random* (MCAR) (see Section 2.1), no bias is introduced when omitting those observations with missing values. However this naive approach, namely omitting all observations that include at least one missing cell, is not particularly attractive because a lot of valuable information contained in these observations is then lost. On the other hand, when the data are *missing at random*, where the probability of missingness depends on certain variables, bias estimates result when deleting rows with zeros. Therefore, the exploration of missing values is crucial for determining their dependencies to other variables (see Section 4).

Even this case entails further challenges which are very characteristic of data sets from official statistics:

Mixed type of variables in the data: Data from official statistics typically consist of variables that have different distributions, i.e. various variables are binary scaled (yes/no questions), some variables might be (ordered) factors (e.g. employee size range), and some variables may be determined to be of continuous scale (e.g. turnover). If missing values are present in all variable types, the challenge is to estimate the missing values based on the whole multivariate information.

Semi-continuous variables: Another challenge is the presence of variables in the data set which which partially consist of a continuous scale, but also include a certain proportion of equal values (typically zeros in official statistics). The distribution of such variables is often referred to as "semi-continuous" distribution (see, e.g., Schafer, 1997). Data consisting of semi-continuous variables occur quite frequently, for example, tax components in tax data, where one part of the data consists of structural zeros (they must be zero by definition) or just zeros (a non-zero value is possible by definition).

Large data sets: Since data collection is a requirement in many fields today, the resulting data sets can become quite "large", and thus the computation time of

imputation methods is crucial. One might argue that many such data sets can be decomposed into subsets referring to sub-populations, which, for instance, are defined by the ISIC-codes in *Business Data*. Nonetheless, these subsets may contain a large number of observations, which calls for fast methods for data imputation.

Far from normality: A common assumption for multivariate imputation methods is that the continuous part of the data originate from a multivariate normal distribution, or that the data can be transformed to approximate multivariate normal distribution. This is violated in the presence of outlying observations in the data. In that case, standard methods can result in very biased estimates for the missing values. It is then more advisable to use robust methods, which are less influenced by outlying observations (see, e.g., Béguin and Hulliger, 2008; Serneels and Verdonck, 2008; Hron et al, 2010; Todorov et al, 2011). Note that a prior exclusion of outliers before imputation is not straightforward and biased estimates may result.

Sampling weights: Sampling weights are typically related to complex surveys where each statistical unit may have different probabilities for inclusion in the sample. How to deal with sampling weights in imputation and outlier detection is not the main focus of this study but for any estimation or aggregation of data, a weighted estimation which takes into account the chosen sampling design has to be considered (for details, see for example Cochran, 1977).

2.1 Types of missing data

The presence of missing data can effect the properties of the estimates obtained from the data (e.g. means, percentages, percentiles, variances, ratios, regression parameters, etc.). Missing data can also affect inferences, i.e. the properties of tests and confidence intervals. A crucial criterion of these effects is the way in which the probability of an item missing depends on other observed or non-observed variables as well as on its own value. We call this *missingness mechanism* (Little and Rubin, 1987). Let's denote the collected data by X and partition the data into

$$X = \{X_o, X_m\} \tag{1}$$

where X_o denotes the observed elements of X and X_m represents the missing elements. The missingness indicator matrix R corresponds X, and each element of R is 1 if the corresponding element of X is missing, and 0 otherwise. Using this missing value indicator we can define the missingness mechanism as the probability of R conditional on the values of the observed and missing elements of X:

$$Pr(R|X_o, X_m) \tag{2}$$

1. **Missing by Design** Some survey participants might be excluded from the analysis because they are not part of the population under investigation. An example of missing values by design in establishment surveys is the cut-off sampling in which

all units (enterprises or establishments) above a certain size threshold are included in the survey with certainty and all the units below this size are excluded from the frame. Since this type of missingness generates unit non-response we will not consider it further. Another example are *valid skips*, i.e. when a question is not answered because it is not applicable to the given unit. In many surveys different missingnes codes are applied indicating the reason why the respondent did not provide an answer: (i) refused to answer; (ii) answered *don't know*; (iii) had a valid skip or (iv) was skipped by an enumerator error. Depending on the code one can decide whether the corresponding values are to be imputed or not.

2. **Missing Completely at Random (MCAR)** implies that the pattern of missing values is totally random and does not depend on any variable, which may or may not be included in the analysis. We can express the MCAR assumption as follows:

$$Pr(R|X) = Pr(R) \tag{3}$$

which means that the probability of missingness in X depends neither on the observed values in any variable in X nor on the unobserved part. For most data sets, the MCAR assumption is unlikely to be satisfied, one exception being the case when data are *missing by design*.

3. **Missing at Random (MAR)** A much weaker assumption is that the data are missing at random (MAR). The MAR assumption can be expressed by:

$$Pr(R|X) = Pr(R|X_o) \tag{4}$$

which means that missingness on X may depend on the observed part of X, but it does not depend on the unobserved part itself. Unfortunately, we can generally not be sure whether data really are missing at random or whether the missingness is attributable to unobserved predictors or the missing data themselves. MAR can never be tested on any given data set because it can be that some unobserved variables, at least partially, are causing the *missing* pattern. MCAR is a special case of MAR, i.e. if the data are MCAR, they are also MAR. The missing-data mechanism is said to be *ignorable* if the data are MAR and the parameters governing the missing-data mechanism are distinct from the parameters in the model to be estimated. Usually, MAR and "ignorability" are used interchangeably. If the missing-data mechanism is ignorable, then it is possible to obtain valid, optimal estimates of parameters without directly modeling the missing-data mechanism (Allison, 2001).

4. **Missing Not at Random (MNAR)** If the MAR assumption is violated, the data are said to be *not missing at random (NMAR)* which in general means that an unknown process is generating the missing values. One of the most prominent examples of the MNAR generating mechanism is the question about income, where the high rate of missing values (usually in the range of 20%—50%) is related to

the value of the income itself (very high and very low values will not be answered). MNAR can appear in one of the two following versions (or a combination thereof):

- *Missingness that depends on unobserved predictors*
- *Missingness that depends on the missing value itself*

In the case of MNAR the missing-data mechanism is not ignorable, and a valid estimation requires the missing-data mechanism to be modeled as part of the estimation process. The results can be very sensitive to the model choice (Little and Rubin, 1987).

2.2 Handling missing values in R

Missing data in R are represented as `NA` and no difference is made between string or numerical missing values. `NA` is simply an indicator of missingess and can be treated as any other value, e.g. we can create a vector and include missing values in it by writing:

```
> v <- c(10, 20, NA, 30, 40)
```

`NA` is the one of the few non-numbers that could be included in `v` without generating an error. Please note that R distinguishes between `NA` and `"NA"` - the former is a missing value while the latter is simply a string.

```
> anum <- c(10, 20, NA, 30, 40)
> astr <- c("Austria", "Australia", NA, NA, "Germany", "NA")
> anum

[1] 10 20 NA 30 40

> astr

[1] "Austria"   "Australia" NA          NA          "Germany"   "NA"
```

We cannot determine which variable has a missing value by comparison, i.e. `anum[1] == NA` is not a valid logical expression and will not return `FALSE` as one would expect but will return `NA`. In order to find which values are missing we can use the function `is.na`, e.g.

```
> is.na(anum)

[1] FALSE FALSE  TRUE FALSE FALSE

> is.na(astr)

[1] FALSE FALSE  TRUE  TRUE FALSE FALSE
```

To find the indexes of the missing values in a vector one could use the function `which()` in combination with `is.na()`:

```
> which(is.na(anum))

[1] 3

> which(is.na(astr))

[1] 3 4
```

To assign a NA value to an element of a vector one could either use the assignment operator or the function is.na() as shown below:

```
> x <- c(1, 4, 7, 10)
> x[4] <- NA              # sets the 4th element to NA
> is.na(x) <- 1           # sets the first element to NA
> x

[1] NA  4  7 NA
```

Note the difference between NA and NULL which some functions and expressions return. These codes are not equivalent - while NA represents an element which is "not available", NULL denotes something which never existed and cannot exist at all. But these codes merge when data are read from a database - in that case the NULLs from the database are represented as NAs in the resulting data frame.

When a vector is used to create a factor by default, the missing value NA will be excluded from factor levels. In order to create a factor that includes missing values from a numeric variable, use exclude=NULL. Similarly, the table() function will not create a factor level for NA which could be achieved by exclude=NULL.

```
> factor(anum)

[1] 10   20   <NA> 30   40
Levels: 10 20 30 40

> factor(anum, exclude=NULL)

[1] 10   20   <NA> 30   40
Levels: 10 20 30 40 <NA>

> table(anum)

anum
10 20 30 40
 1  1  1  1

> table(anum, exclude=NULL)
```

```
anum
  10   20   30   40 <NA>
   1    1    1    1    1
```

Most of the summary functions in R can optionally exclude the missing values from calculations by using an argument specifying how the missing values are treated. This argument is `na.rm` and by default is set to `FALSE`, meaning that the missing values are not removed. As a result of this the return value will be `NA`.

```
> x <- c(1, 4, NA, 10)
> summary(x)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
    1.0     2.5     4.0     5.0     7.0    10.0     1.0

> mean(x)

[1] NA

> sd(x)

[1] NA

> mean(x, na.rm=TRUE)

[1] 5

> sd(x, na.rm=TRUE)

[1] 4.582576
```

Many modeling functions in R like `lm()`, `glm()`, `gam()` have an optional argument that specifies how to handle missing values which is usually a function for preprocessing the input data. This function can be one of the following or any other defined by the user:

- `na.fail()` - issue an error if the object contains missing values
- `na.omit()` - exclude the missing values and return the rest of the object
- `na.exclude()` - same as `na.omit()` but will result in different behavior of some functions (like `napredict()` and `naresid()`)
- `na.pass()` - return also the missing values (the object remains unchanged)

```
> ## Most of the statistical modeling functions have
> ##  an argument na.action (lm, glm, gam)
> ##  na.action defaults to na.omit;
> ##  can be na.exclude, na.fail, na.pass
> options(error = expression(NULL))
> df <- as.data.frame(matrix(c(1:7, NA), ncol = 2))
> names(df) <- c("Y", "X")
> df
```

```
  Y  X
1 1  5
2 2  6
3 3  7
4 4 NA
```

```
> lm(Y~X, data=df)
```

```
Call:
lm(formula = Y ~ X, data = df)


Coefficients:
(Intercept)            X
         -4            1
```

```
> lm(Y~X, data=df, na.action=na.omit)
```

```
Call:
lm(formula = Y ~ X, data = df, na.action = na.omit)


Coefficients:
(Intercept)            X
         -4            1
```

```
> lm(Y~X, data=df, na.action=na.fail)
```

```
Error in na.fail.default(structure(list(Y = 1:4,
    X = c(5L, 6L, 7L, NA)), .Names = c("Y",  :
    missing values in object
```

Other special values in R The `NA` symbol for a missing value is not the only one special symbol in R. `NaN` is used to denote a value which is "not a number" which can arise for example when we try to compute the undeterminate $0/0$. To check whether a value is "not a number" the function `is.nan()` is used.

```
> x <- c(1, 0, 10)
> x/x
```

```
[1]   1 NaN   1
```

```
> is.nan(x/x)
```

```
[1] FALSE  TRUE FALSE
```

Another special symbol is infinity `Inf` which results from computations like $1/0$. using the functions `is.finite()` and `is.infinite()` we can determine whether a number is finite or not.

```
> 1/x

[1] 1.0 Inf 0.1

> is.finite(1/x)

[1]  TRUE FALSE  TRUE

> -10/x

[1]  -10 -Inf   -1

> is.infinite(-10/x)

[1] FALSE  TRUE FALSE
```

Some calculations involving `Inf` can be evaluated but others lead to "not a number" `NaN`s:

```
> exp(-Inf)

[1] 0

> 0/Inf

[1] 0

> Inf - Inf

[1] NaN

> Inf/Inf

[1] NaN
```

Differences to other packages Like R, most of the other software packages feature options for missing value codes but there are important distinctions which we need to keep in mind. The default behaviour of most R functions is to return `NA` whenever any of the input variables have missing values. Missing values are handled differently in SAS procedures and `data steps`, consequently, the manual should be consulted for specific cases. A brief overview can be found at http://www.ats.ucla.edu/stat/sas/modules/missing.htm

Numeric and character missing values are coded differently in SAS - a single blank enclosed in quotes (' ') is used for the character values and a single period(.) for numeric ones. Additionally, there are 27 different codes for special numeric missing values - a single period followed by a single letter or underscore (for example, .A, .B, .Z, ._). In SAS a missing value for a numeric variable is smaller than all numbers and if one

sorts the data set by a numeric variable, observations with missing values for that given variable appear first in the sorted data set. For numeric variables, one can compare special missing values with numbers and with each other.

Similarly, STATA distinguishes between numeric and character missing values. The standard numeric missing value code (or `sysmiss`) is a single period ('.'), also called *system missing value*, and there are additional 26 missing value codes which consist of a single period and a lower case letter, like `.a, .b, .z`. These are referred to as *extended missing values*. Numeric missing values in STATA are represented by large positive values. Thus, when sorted, a numerical variable with missing values will include all observed values first, followed by the missing ones. When reading data (e.g. from Excel) the missing values need to have been left blank in order to be recognized as missing by STATA. See help for the functions `read.dta()` and `write.dta()` from package **foreign** to see how missing data are handled by R when reading or writing a STATA binary file.

R does not distinguish between character and numerical variables and uses the missingness code `NA` in both cases. Thus, there is only one missingess code and one cannot differentiate between different types of numerical missing values as in the case in SAS and STATA. To test for a missing value in R the function `is.na()` is used, not the logical equality operator `==`. When importing numeric data in R, blanks will be interpreted as missing values (except when blanks are delimiters) and the string `NA` will be interpreted as missing for both numeric and character variables. While both SAS and STATA would recognize a single period as a missing value for numeric variables, R will instead read the entire variable as a character vector. This can be fixed in R

```
> library(ritso)

Scalable Robust Estimators with High Breakdown Point (version 1.3-01)
Scalable Robust Estimators with High Breakdown Point for
Incomplete Data (version 0.4-02)
R in the Statistical Office (version 0.1-00)

> fname <- system.file("examples", "xmiss.txt", package="ritso")
> ##  Period (.) will not be recognized as missing value
> read.table(file=fname, header=TRUE)

  ageh agew edu   inc kid
1   30   28  15 15000   0
2   43    .   . 41260   2
3   34   30  17 55000   1
4   40    .   7 67050   3
5   67   55   . 78000   3
6   41   37  12     .   2
```

```
> ##  Now, when using the argument na.strings(),
> ##  the period (.) will be recognized as missing value
> read.table(file=fname, header=TRUE, na.strings=c("."))

  ageh agew edu   inc kid
1   30   28  15 15000   0
2   43   NA  NA 41260   2
3   34   30  17 55000   1
4   40   NA   7 67050   3
5   67   55  NA 78000   3
6   41   37  12    NA   2
```

2.3 Traditional approaches to handling missing data

Standard traditional practices like list-wise (or case-wise) deletion, mean substitution, pair-wise deletion and regression substitution are the simplest way to deal with missing values and these are the methods usually implemented in the standard statistical software packages. In the following, we will provide simple examples of how these methods can be applied in R. The limitations of traditional approaches to deal with missing values have been studied and are well-documented (Allison, 2001; Acock, 2005). Although often used, none of these methods is an optimal solution for handling missing values problems except, in specialized cases.

1. **List-wise deletion** (or case-wise deletion or complete case analysis). With this method all units with missing data for a variable are removed and the analysis is performed with the remaining units (complete cases). This is the default approach in most statistical packages. The use of this method is only justified if the missing data generation mechanism is MCAR. In R we can either omit all observations which have missing values using the function `na.omit()` or extract complete observations using the function `complete.cases()`.

   ```
   > library(ritso)
   > data(mdata)
   > dim(mdata)      # we have 25 observations,

   [1] 25  5

   >                 # some of them contain missing values
   > x1 <- na.omit(mdata)
   > dim(x1)         # only 14 observations remain

   [1] 14  5

   > x2 <- mdata[complete.cases(mdata),]
   > dim(x2)         # similarly, only 14 observations remain
   ```

```
[1] 14  5
```

We can use the second function to determine which observations contain missing values applying the logical `NOT`, i.e. `!`:

```
> mdata[!complete.cases(mdata),]
```

```
   ageh agew edu   inc kid
10   43   NA  NA 41260   2
12   40   NA   7 67050   3
13   67   55  NA 78000   3
14   34   30  16    NA   2
15   25   26  12    NA   1
17   36   NA  18 83200   1
18   42   37  NA    NA   3
20   38   NA   0 25000   3
21   65   NA   5 70000   4
23   40   35  16    NA  NA
25   41   37  12    NA   2
```

2. **Pairwise deletion** (or available case analysis). uses all available cases to conduct the analysis. If the data are used in multivariate analyses to compute a covariance matrix, each two cases will be used for which the values of both corresponding variables are available. This method draws on the complete information but the computed covariance matrix could be non-positive definite. In R we can use pair-wise deletion, for example, in the build-in function `cov` for computing the covariance matrix of a data set, which has a `use` argument. The default is `use="everything"` and the function will thus use all observations. This will result in a covariance matrix most likely consisting of `NA`s only due to the propagation of missing values. If `use="all.obs"`, then the presence of missing observations will produce an error. If `use="complete.obs"`, then missing values are handled by list-wise deletion (and if there are no complete cases, an error appears). If `use="pairwise.complete.obs"`, then the covariance between each pair of variables is computed using all complete pairs of observations on those variables. This can result in covariance or correlation matrices which are not positive semi-definite, as well as `NA` entries if there are no complete pairs for the given pair of variables.

```
> cov(mdata)
> cov(mdata, use="all.obs")
> cov(mdata, use="complete.obs")
> cov(mdata, use="na.or.complete")
> cov(mdata, use="pairwise")
```

3. **Non-response weighting** (or adjusting for non-response by weighting). Non-response would not be problematic if the non-respondents were a random sample

of the total sample, but this is seldom the case. Usually the achieved response rates follow particular patterns. For example, in household surveys it is well known that non-respondents are younger than respondents, and that men are more difficult to persuade to take part in a survey than women. Response rates in cities and deprived areas also tend to be lower than the average. The result is that such a survey will typically overrepresent women, and those over the age of 30. Those living in cities and deprived areas will be often underrepresented. If such patterns are known for the respective survey one could use weights to to bring the data set obtained more closely in line with the sampled population. The adjustment of the weights for non-response will be complicated if more than one variable has missing values and the standard errors may become erratic. It is not recommended to perform weight adjustments for larger establishments (with more than 50 employees) but rather to impute the missing values. On the other hand, weight adjustments should not be performed if missing values are imputed.

4. **Mean substitution.** A very simple but popular approach is to substitute means for the missing values. The method preserves sample size and thus does not reduce the statistical power associated with sample size in comparison with list-wise or pairwise deletion procedures, but it is well-known that this method produces biased estimates and can severely distort the distribution of the variable in which missing values are substituted. This results in underestimates of the standard deviations and distorts relationships between variables (estimates of the correlation are pulled toward zero). Due to these distributional problems, it is often recommended to ignore missing values rather than impute values by mean substitution (Little and Rubin, 1989). The following example shows how to apply mean substitution to a data set with missing values.

```
> library(ritso)
> data(mdata)
> tail(mdata)

   ageh agew edu   inc kid
20   38   NA   0 25000   3
21   65   NA   5 70000   4
22   34   36  12 85000   1
23   40   35  16    NA  NA
24   38   38  18 95000   2
25   41   37  12    NA   2

> mean.subst <- function(a) {
+     a[is.na(a)] <- mean(a, na.rm=TRUE)
+     a
+ }
```

```
> ximp <- apply(mdata, 2, mean.subst)
> tail(ximp)

   ageh  agew edu   inc  kid
20   38 35.25   0 25000 3.00
21   65 35.25   5 70000 4.00
22   34 36.00  12 85000 1.00
23   40 35.00  16 57738 2.25
24   38 38.00  18 95000 2.00
25   41 37.00  12 57738 2.00
```

5. **Regression substitution.** The technique of regression substitution replaces missing values in an input variable by first treating this input as a target and using the remaining input variables (and other rejected variables) as predictors in a regression model. Then the predicted value of the regression model is used to substitute the missing value. This technique might be more accurate than simply substituting a measure of central tendency, since the imputed value is based on other input variables. The problem with this technique is that it underestimates standard errors by underestimating the variance in x.

6. **Last value carried forward.** For longitudinal data (repeated measures are taken per subject) one could apply the *Last Value Carried Forward (LVCF or LOCF)* technique. The last observed value is used to fill in missing values in subsequent observations assuming that the most recent observation is the best guess for subsequent missing values (i.e. that the response remains constant for the last observed value). Unfortunately this assumption could be biased. This technique can be applied in R using the function `na.locf()` from the package **zoo** as shown in the next example.

   ```
   > library(zoo)
   > na.locf(mdata)
   ```

7. **Using information from related observations.** In official Statistics, the imputation of missing values with a donor from the underlying data (*hot-deck imputation*) or with a donor from external data (*cold-deck imputation*) is still popular.

   Usually the data set is split into domains, and within such a domain the data are ordered based on a set of pre-defined variables, which are called *sorting variables*. *Sequential* hot/cold-deck imputation then selects the observation which is at the top of the sorted variables as a donor, while *random* hot/cold-deck imputation chooses randomly an observation in the domain. Listing 1 shows an example of the hot-deck procedure using the function `hotdec()` from the R package **VIM**. A small subset of the Community Information Survey (CIS) 2008 is used (for a short discussion about this data set, see Section **??**). This data set includes some missing values for turnover (*turn08*) and number of employees (*emp08*). In this

function the variables which have to be imputed must be specified and within each (sorted) domain the donor has to be chosen (for more details, see the help option of this function).

```
library(VIM)
head(xmiss, 3)
   nuts nace emp08   turn08 x$weight
2  AT12 1110     3  2524483     1.250
1  AT13 1110    75       NA     1.250
3  AT31 1412    19  3181055     5.167

x <- hotdeck(xmiss, variable=c("emp08","turn08"), ord_var=c("nace","
    nuts"),domain_var="nuts")

head(x3, 3)
  nuts nace emp08   turn08 x$weight
2 AT12 1110     3  2524483     1.250
1 AT13 1110    75 10083030     1.250
3 AT31 1412    19  3181055     5.167
```

Listing 1: Hot-deck imputation.

8. **Dummy variable adjustment** or (indicator variable adjustment). This technique will set the missing values of a given variable to be equal to some arbitrary value (usually the mean for non-missing cases) and create a dummy variable indicating missingness which is then included in the regression model. This method is simply an ad-hoc means to keep observations in the analysis and has no sound theoretical justification. It leads to biases in the estimated regression parameters and the standard errors, even if the data are MCAR, thus making it unacceptable (Jones, 1996).

9. **Deterministic imputation.** The deterministic imputation identifies cases in which there is only one possible solution (based on logical rules) and thus allows the record to satisfy the rules.

2.4 Missing data imputation

Many different methods for imputation have been developed over the last few decades. The techniques for imputation may be divided into univariate methods such as column-wise (conditional) mean imputation, and multivariate imputation using the linear dependencies between variables. In the latter case there are generally three approaches: (i) data-ordering and distance-based imputation methods such as hot-deck methods and k-nearest neighbour imputation, (ii) covariance-based methods such as the approaches by Verboven et al (2007) or Serneels and Verdonck (2008), and (iii) model-based methods approaches such as regression imputation (Raghunathan et al, 2001; Templ et al, 2010) or depth-based imputation (Béguin and Hulliger, 2004). The assumption of elliptical distributions is necessary for all covariance-based methods, but not for depth-based ones.

Because of the unique structure of the data arising in in official statistics, "only" the already considered nearest neighbour (hot-deck, cold-deck, k-nearest neighbour) methods and model-based imputation provide good estimates for the missing values.

1. **k-nearest neighbour imputation.** More sophisticated methods for missing data imputation are based on a neighbour search. The k-nearest neighbour imputation searches for the k-nearest observations (respective to the observation which has to be imputed) and replaces the missing value with the mean of the found k observations (or with the most frequent value among the k nearest neighbours in case of a discrete variable). As an estimate of the mean it is recommended to use the (weighted) median instead of the arithmetic mean. Listing 2 provides an example of the k-nearest neighbour imputation method which chooses the distance metric for each variable automatically ((it can, of course, also be specified)). Similarly to the hot-deck imputation the variables of interest have to be determined, especially those variables that contribute to the calculation of the distance (see the `dist_var` argument of the function).

```
x <- kNN(xmiss, variable=c("emp08","turn08"), k=5, dist_var=c("nuts
    ","nace","emp08","turn08"), numFun = median)
```

Listing 2: kNN imputation.

2. **Maximum likelihood and multiple imputation.** *Maximum likelihood (ML)* and *multiple imputation (MI)* are currently considered the "state of the art" and are the recommended missing data techniques in the methodological literature. These methods have a strong theoretical framework and are supported by a large number of empirical studies in different domains. The popularity of these methods has risen recently due to the available implementations in a variety of both commercial and free statistical software programs. The *Expectation Maximisation (EM)* imputation method (Dempster et al, 1977) assumes that the underlying model for the observed data is Gaussian. This method is able to deal with MCAR and MAR missing values mechanism. For Gaussian data the EM algorithm starts with some initial values for the mean and the covariance matrix and iterates through imputing missing values (imputation step) and re-estimating the mean and the covariance matrix from the complete data set (estimation step). The iteration process ends when the maximum relative difference in all of the estimated means, variances or covariances between two iterations is less than or equal to a given value. The parameters thus estimated are used to draw the missing elements of the data matrix under the multivariate normal model (for further details about the Gaussian imputation see Schafer, 1997, Sections 5.3 and 5.4). Functions for treating missing values in normal data are implemented in the R package **norm**. Listing 3 presents an example using data from a household survey with missing values (Schafer, 1997)). The example data are available in the package **norm**.

```
> library(norm)
> data(mdata)
> tail(mdata, 4)
   ageh agew edu   inc kid
22   34   36  12 85000   1
23   40   35  16    NA  NA
24   38   38  18 95000   2
25   41   37  12    NA   2
> s <- prelim.norm(mdata)   #do preliminary manipulations
> thetahat <- em.norm(s, showits=FALSE)   #find the mle
> rngseed(1234567)   #set random number generator seed
> ximp <- imp.norm(s,thetahat,mdata)  #impute missing data under the
    MLE
> tail(ximp, 4)
   ageh     agew edu      inc      kid
22   34 36.00000  12 85000.00 1.000000
23   40 35.00000  16 79331.41 2.755392
24   38 38.00000  18 95000.00 2.000000
25   41 37.00000  12 54523.40 2.000000
```

Listing 3: Imputation of missing multivariate normal data with `imp.norm()` from the R package **norm**.

There are at least three popular implementations that include iterative model-based imputation: the software implementation by Raghunathan et al (2001), the R-package **mi** (Gelman et al, 2010) and the function `irmi()` of R-package **VIM** (Templ and Filzmoser, 2008). (Templ et al, 2010) have shown the advantages of the IRMI method (implemented in the function `irmi()`) which does not assume that the data follows a multivariate normal distribution and uses robust methods.

All the mentioned software implementations are able to deal with the randomness inherent to the data and can be used for multiple imputation generating more than one candidate for a missing cell (Rubin, 1987). Multiple imputation is one way to express uncertainty related to missing values and can be used as one way to estimate the variance of an estimator. However, a valuable inference can also be obtained by applying bootstrap methods (Little and Rubin, 1987; Alfons et al, 2009). Note that whenever the focus is not on variance estimation, there is no need for multiple imputation.

The basic procedure behind most model-based imputation methods is the already mentioned EM algorithm. For the estimation, regression methods are usually applied in an iterative manner, which is known as regression switching, chain equations, sequential regressions or variable-by-variable Gibbs sampling (see, e.g., van Buuren and Oudshoorn, 2005; Muennich and Rässler, 2004).

Listing 4 shows the application of the function `irmi()` from the R package **VIM**.

A short print output reveals that `irmi()` automatically recognizes (this can be specified anyway) the distribution of the variables.

```
x <- irmi(xmiss, robust=TRUE, robMethod="MM")

Imputation performed on the following data set:
          type       #missing
nuts      "nominal"  "0"
nace      "nominal"  "0"
emp08     "numeric"  "10"
turn08    "numeric"  "100"
x$weight  "numeric"  "0"

head(x, 3)
  nuts nace emp08   turn08 x$weight
2 AT12 1110     3  2524483    1.250
1 AT13 1110    75 12084880    1.250
3 AT31 1412    19  3181055    5.167
```

Listing 4: EM-based regression imputation using function `irmi()` from package **VIM**.

3. **Other imputation methods.** Numerous alternative models and computational methods exist for multiple imputation. For example, MCMC algorithm can be used, but applied to models other than the multivariate normal model. For example, for data in which all the variables are categorical the R package **cat** can be used. It uses the MCMC algorithm under a multinomial model or a restricted log-linear model.

   Another R package, **mix**, is suitable for data sets and models that include both categorical and quantitative variables. While the approach implemented in this package might seem to be ideal for many situations, the model is rather complex and its implementation requires considerable thought and care (Allison, 2001).

3 Editing and Outlier Detection

Data collected in surveys generally contain errors due to a variety of reasons. For example, a respondent may give a wrong answer or errors may arise while processing the data. *Statistical data editing*, i.e. checks and corrections, are necessary to increase the quality of the data. First, erroneous values in the data set have to be localized, preferably in an automated manner. The localized erroneous values have to be replaced by reasonable values (imputation). It is not necessary to remove all errors from a data set in order to obtain reliable publication figures (De Waal, 2008) since the sampling error might be higher than the error related to minor errors in the data. Conducting automated micro-editing for minor errors is often too ambitious and leads to over-editing. However, logical relationships in the data should be preserved. For example, the income components should amount to the income or a girl who is 14 years old cannot be married

and have three children. In addition, large non-representative outliers (measurement errors) have to be localized and replaced by reasonable estimates or by true values which may be obtained by contacting the corresponding person or enterprise. Even a bulk of errors could be prevented by including validation rules in the data collection phase, for example, by using an electronic questionnaire.

Selective editing methods are techniques for identifying the most influential errors in a data set, i.e. the errors that have a substantial impact on pre-defined aggregates or statistics. Generally speaking, it is an application of the theory of the influence function in robust statistics. The drawbacks of selective editing are that one observation may have considerable influence on one statistic but almost no influence on another statistic, and that the multivariate structure of the data are not considered adequately. This is solved by outlier detection rules.

Outlier detection: The detection of outliers, i.e. of atypical observations which deviate from the usual data variability, is very important in statistical analysis since classical statistical models applied to data including outliers can lead to misleading results. In addition to that, measurement errors may have high influence on aggregates typically published in statistical tables. Continuous scaled variables in official statistics, mostly related to business statistics, come with a high percentage of zeros, resulting in semi-continuous variables. This leads to a serious limitation of methods used for outlier detection and in combination with missing values in the data makes an application of well-known standard outlier detection methods impossible. One approach to deal with these extra challenges is to simply handle the zeros in the data as missing values, to impute these "missings" with an appropriate imputation method and, finally, to apply conventional outlier detection methods on the imputed data. A possible disadvantage of this approach is the strong dependence on the performance of the imputation method used. Another approach is to omit observations with zeros. However, this causes a problem for multivariate methods due to the fact that excluding observations with zeros might render a data matrix far too small for drawing significant conclusions. Hence, a pair-wise approach of certain multivariate methods seems rather sensible due to the fact that it is now possible to make use of a considerable amount of observations from the actual data without having to resort to imputation.

The multivariate aspect of the data collected in surveys makes the task of outlier identification particularly challenging. The outliers can be completely hidden in one or two dimensional views of the data. This underlines that univariate outlier detection methods are useless, although they are favoured in National Statistical Offices (NSO) because of their simplicity. For more details on multivariate outlier detection in business survey data, see Todorov et al (2011) and the references therein. The focus of the present study is on using robust methods to identify the outliers which can subsequently be treated in the traditional way and particulary in presenting R software tools to accomplish this

task.

3.1 General principles

A general approach to multivariate outlier identification in a p-dimensional data set $X = (x_1, \ldots, x_n)$ is to compute a given measure for the distance of a particular data point from the centre of the data and declare those points which are too far from the centre as outliers. Usually, as a measure of "outlyingness" for a data point $x_i, i = 1, \ldots, n$, a robust version of the (squared) Mahalanobis distance RD_i^2 is used, computed relative to high breakdown point robust estimates of location T and covariance C of the data set X:

$$RD_i^2 = (x_i - T)^t C^{-1} (x_i - T) \tag{5}$$

The most common estimators of the multivariate location and scatter are the sample mean $\bar{x}$ and the sample covariance matrix S, i.e. the corresponding ML estimates (when the data follow a normal distribution). These estimates are optimal if the data derive from a multivariate normal distribution but are extremely sensitive to the presence of even a few outliers in the data. The outlier identification procedure based on $\bar{x}$ and S faces the following two problems (Rousseeuw and Leroy, 1987):

1. *Masking:* multiple outliers can distort the classical estimates of mean $\bar{x}$ and covariance S in such a way (attracting $\bar{x}$ and inflating S) that they do not necessarily attain large values for the Mahalanobis distance, and

2. *Swamping:* multiple outliers can distort the classical estimates of mean $\bar{x}$ and covariance S in such a way that observations which are consistent with the majority of the data attain large values for the Mahalanobis distance.

In the last several decades much effort has been devoted to the development of affine equivariant estimators with a high breakdown point. The most widely used estimators of this type are the Minimum Covariance Determinant (MCD) estimator and the Minimum Volume Ellipsoid (MVE) estimator, S-estimators and the Stahel-Donoho estimator. These estimators can be configured in such a way as to achieve the theoretically maximal possible breakdown point of 50 percent which gives them the ability to detect outliers even if their number amounts to nearly half of the sample size. If we give up the requirement for affine equivariance, estimators like the orthogonalized Gnanadesikan-Kettenring (OGK) estimator are available and the reward is an extreme gain in speed. For definitions, algorithms and references to the original papers see Maronna et al (2006). Most of these methods are implemented in the R statistical environment (R Development Core Team, 2011) and are available in the object-oriented framework for robust multivariate analysis (Todorov and Filzmoser, 2009).

After having found reliable estimates for the location and covariance matrix of the data set, the second issue is determining how large the robust distances should be in order to declare a point an outlier. The usual cutoff value is a quantile of the χ^2 distribution,

like $D_0 = \chi^2_p(0.975)$. The reason is that if X follows a multivariate normal distribution, the squared Mahalanobis distances based on the sample mean $\bar{x}$ and sample covariance matrix S follow χ^2_p distribution (see, for example Johnson and Wichern, 2002, p. 189). This will no longer be valid if robust estimators are applied and/or if the data have other than multivariate normal distribution. Maronna and Zamar (2002) propose using a transformation of the cutoff value which should help the distribution of the squared robust distances RD_i^2 to resemble χ^2 for non-normal original data:

$$D_0 = \frac{\chi^2_p(0.975) med(RD_1^2, \ldots, RD_n^2)}{\chi^2_p(0.5)}. \tag{6}$$

For other alternatives which may lead to more accurate cutoff value, see Filzmoser et al (2005); Hardin and Rocke (2005); Cerioli et al (2009); Riani et al (2009).

A drawback of all methods considered so far is that they only work with complete data, which is not a usual case when dealing with sample surveys. In the next two subsections we will describe and introduce methods that are able to cope with missing values.

3.2 Algorithms based on imputation

Robustifying the EM algorithm. Little and Smith (1987) were the first to propose a robust estimator for incomplete data by replacing the MLE in the M-step of the EM algorithm (see Dempster et al, 1977) by an estimator belonging to the general class of M-estimates (Huber, 1981) and called this procedure ER-estimator. They suggested to use ML estimation as a starting point for the ER algorithm, where the missing values were replaced by the median of the corresponding observed data. Unfortunately, the breakdown point of this estimator, as of all general M-estimates, cannot be higher than $1/(p+1)$ (see for example Maronna et al, 2006, p. 186) which renders it unusable for the purpose of outlier detection. Copt and Victoria-Feser (2004) constructed a high breakdown point estimator of location and covariance for incomplete multivariate data by modifying the MCD estimator and using it as a starting point not for an ER algorithm, but for an S-estimator, adapted to work with incomplete data. They call this estimator `ERTBS`. An implementation of this procedure was available by the authors of this study in the form of a compiled shared library, but it did not perform as well as expected and was excluded from further investigations in the present work.

Normal imputation followed by high-BP estimation. A straightforward strategy for adapting estimators of location and covariance to work with missing data is to perform one preliminary step of imputation and then run any of the above described algorithms, such as, for example MCD, OGK, S and Stahel-Donoho (SDE) on the complete data. Many different methods for imputation have been developed over the last few decades and we will consider a likelihood-based approach such as the aforementioned expectation maximization (EM) imputation method (Dempster et al, 1977) here, assum-

ing the underlying model for the observed data is Gaussian. This method is able to deal with the MCAR and MAR missing values mechanism. For Gaussian data the EM algorithm starts with some initial values for the mean and the covariance matrix and iterates through imputing missing values (imputation step) and re-estimating the mean and the covariance matrix from the complete data set (estimation step). The iteration process ends when the maximum relative difference in all of the estimated means, variances or covariances between two iterations is less than or equal to a given value. The parameters estimated in this way are used to draw the missing elements of the data matrix under the multivariate normal model (for further details about the Gaussian imputation, see Schafer, 1997, Sections 5.3 and 5.4). We are tempted to call this class of methods PM-MCD (poor man's MCD), etc. but for the sake of simplicity we will simply call them MCD, S, SDE, etc., meaning the high breakdown method applied to complete data and at the same time applied to normally (non-robustly) imputed data. Whenever another method for imputation precedes the high breakdown estimation method, like the robust sequential imputation described in the next section, the corresponding notation will be used. Thereby we can also adapt projection based algorithms like SIGN1 (Filzmoser et al, 2008).

The next step after reliably estimating the location T and covariance matrix C is to compute the robust distances from the incomplete data. For this purpose we have to adapt Equation (5) to use only the observed values in each observation x_i and then scale up the obtained distance. We rearrange the variables if necessary and partition the observation x_i into $x_i = (x_{oi}, x_{mi})$ where x_{oi} denotes the observed part and x_{mi} - the missing part of the observation. Similarly, the location and covariance estimates are partitioned, so that we have T_{oi} and C_{oi} as those parts of T and C which correspond to the observed part of x_i. Then

$$RD_{oi}^2 = (x_{oi} - T_{oi})^t C_{oi}^{-1} (x_{oi} - T_{oi}) \tag{7}$$

is the squared robust distance computed only from the observed part of x_i. If x_i is uncontaminated, follow a multivariate normal distribution, and if the missing values are missing at random, then the squared robust distance given by Equation (7) is asymptotically distributed as $\chi^2_{p_i}$ where p_i is the number of observed variables in x_i (see Little and Smith, 1987).

The MCD estimator is not very efficient in normal models, especially if h is selected to achieve the maximal breakdown point (BP) (Croux and Haesbroeck, 1999), and the same is valid for the OGK estimator (Maronna et al, 2006, p. 193, 207). To overcome the low efficiency of these estimators, a reweighted version can be used (see Lopuhaä and Rousseeuw, 1991; Lopuhaä, 1999). For this purpose weight w_i is assigned to each observation x_i, defined as $w_i = 1$ if $RD_{oi}^2 \leq \chi^2_{p_i,0.975}$ and $w_i = 0$ otherwise, relative to the raw estimates (T, C) and using Equation (7). Then the reweighted estimates are computed

as

$$
\begin{aligned}
T_R &= \frac{1}{\nu}\sum_{i=1}^{n} w_i x_i, \\
C_R &= \frac{1}{\nu-1}\sum_{i=1}^{n} w_i (x_i - T_R)(x_i - T_R)^t,
\end{aligned} \tag{8}
$$

where ν is the sum of the weights, $\nu = \sum_{i=1}^{n} w_i$. Since the underlying data matrix is incomplete, the EM algorithm is used to compute T_R and C_R. These reweighted estimates (T_R, C_R), which have the same breakdown point as the initial (raw) estimates but better statistical efficiency, are computed and used by default for the methods MCD and OGK.

Robust sequential imputation followed by high-BP estimation. Since we assume that outliers are present in the data we could expect an improvement of the performance of the previously described methods if the non-robust Gaussian imputation is substituted by a robust imputation technique that can handle simultaneously missing and outlying values. One such method was proposed by Vanden Branden and Verboven (2009) (`RSEQ`), extending the sequential imputation technique (`SEQimpute`) of Verboven et al (2007) by robustifying some of its crucial steps. `SEQimpute` starts from a complete subset of the data set X_c and sequentially estimates the missing values in an incomplete observation, say x^*, by minimizing the determinant of the covariance of the augmented data matrix $X^* = [X_c; (x^*)^t]$. Since `SEQimpute` uses the sample mean and covariance matrix it will be vulnerable to the influence of outliers and it is improved by plugging in robust estimators of location and scatter. One possible solution is to use the outlyingness measure as proposed by Stahel (1981) and Donoho (1982), which was successfully used for outlier identification in Hubert et al (2005). We can compute the outlyingness measure for the complete observations only, but once an incomplete observation is imputed (sequentially) we can compute the outlyingness measure for it also and use it to decide whether this observation is an outlier or not. If the outlyingness measure does not exceed a predefined threshold the observation is included in the further steps of the algorithm. After obtaining a complete data set we proceed by applying a high breakdown point estimation method in the same way as described in the previous section.

3.3 Algorithms based on other strategies

Transformed Rank Correlation (TRC) This is one of the algorithms proposed by Béguin and Hulliger (2004) and is based, similarly to the OGK algorithm of Maronna and Zamar (2002), on the proposal of Gnanadesikan and Kettenring for pair-wise construction of the covariance matrix. After all missing items have been imputed (or some observations with too few observed items have been removed) the complete data matrix can be used to perform the transformation and compute the final robust location and covariance matrix as in the case of complete data.

Epidemic Algorithm (EA) The second algorithm proposed by Béguin and Hulliger (2004) is based on data depth and is distribution free. It simulates an epidemic which starts at a multivariate robust centre (sample spatial median) and propagates through the point cloud. The infection time is used to determine the outlyingness of the points. The latest infected points or those not infected at all are considered outliers. The adaption of the algorithm to missing values is straightforward by leaving out missing values from the calculations of the univariate statistics (medians and median absolute deviations) as well as from the distance calculations.

BACON-EEM (BEM) The third algorithm (Béguin and Hulliger, 2008) developed in the framework of the EUREDIT project is based on the algorithm proposed by Billor et al (2000), which in turn is an improvement over an earlier "forward search" based algorithm by one of the authors.

The last three algorithms - TRC, EA and BEM also take into account the sampling context in which the data are assumed to be a random sample s of size n from a finite population U with size N and the sample is drawn with the sample design $p(s)$. The sample design $p(s)$ defines the inclusion probabilities and the corresponding sampling weights. With these weights the classical mean and covariance are estimated using the Hàjek estimators, and the weighted median and MAD are estimated as described in Béguin and Hulliger (2004) where further details about the adaptation of the algorithms to sampling weights can be found.

Robust Principal Components for incomplete data Principal component analysis (PCA) is a widely used technique for dimension reduction achieved by finding a smaller number k of linear combinations of the originally observed p variables and retaining most of the variability of the data. These new variables, referred to as *principal components*, are uncorrelated with each other and account for a decreasing amount of the total variance, i.e. the first principal component explains the maximum variance in the data and the second principal component explains the maximum variance in the data that has not been explained by the first principal component and so on. Dimension reduction by PCA is mainly used for visualization of multivariate data by scatter plots (in a lower dimensional space) or transformation of highly correlated variables into a smaller set of uncorrelated variables which can be used by other methods (e.g. multiple or multivariate regression). The classical approach to PCA measures the variability through empirical variance and is essentially based on the computation of eigenvalues and eigenvectors of the sample covariance or correlation matrix. Therefore, the results may be extremely sensitive to the presence of even a few atypical observations in the data. The outliers could artificially increase the variance in an otherwise uninformative direction and this direction will be determined as a PC direction. PCA was probably the first multivariate technique subjected to robustification, either by simply computing the eigenvalues and eigenvectors of a robust estimate of the covariance matrix or directly by estimating each principal component in a robust manner. Different approaches to

robust PCA are presented in Todorov and Filzmoser (2009) and examples are given how these robust analyses can be carried out in R. Details on the methods and algorithms can be found in the corresponding references.

Projecting the data into a lower dimensional space, one could obtain an estimate of the location and covariance matrix and then use them for outlier detection as described in the beginning of this section or alternatively, one could directly compute the Mahalanobis distances of the projected observations to the projection of the centre of the data (see, for example Filzmoser et al, 2008).

If the data are incomplete as is usual in business surveys, standard classical or robust PCA cannot be applied. Walczak and Massart (2001) propose using the EM approach to deal with missing data in PCA, and Serneels and Verdonck (2008) extended it to robust PCA. Most of the known methods for robust PCA are implemented in the package **rrcov** (see Todorov and Filzmoser, 2009), and the corresponding versions for dealing with incomplete data can be found in the package **rrcovNA**. More details about the implementation and examples will be presented in Section 3.6.

Handling of semi-continuous variables. In establishment and other surveys variables are often used, which have valid values either in a given interval or are zero. These variables must be treated in the same way as regular variables, except that a value of zero is also accepted. There could, of course, be a minimum that is bigger than zero on the variable and the number of zero valued observations could be larger than half of the total number. Such variables are called semi-continuous variables and it is obvious that none of the methods discussed so far can handle such types of variables. Recently Meraner (2010) proposed a modification of the OGK algorithm which can handle semi-continuous variables. This approach takes advantage of the pair-wise character of the algorithm which allows "skipping" the zeros in the actual computation of robust location and covariance matrix estimates, and then use them for outlier detection.

3.4 Software availability

The algorithms discussed in this paper are available in the R package **rrcovNA** which in turn uses the packages **robustbase**, **rrcov** and **mvoutlier**. These packages are available from the Comprehensive R Archive Network (CRAN) at `http://CRAN.R-project.org` under the GNU General Public License. The three algorithms from the EUREDIT project (`TRC`, `EA` and `BEM`) were kindly provided by the authors and will be included in a later version of **rrcovNA**.

3.5 Example session

In this section we will introduce the base functionalities provided in the package **rrcovNA** to analyse incomplete data by an example session. First, we have to load the package

rrcovNA which will cause all other necessary packages to be loaded as well. The framework includes example data sets but here we will only load those which will be used throughout the following examples. For the rest of the paper it will be assumed that the package has already been loaded.

```
R> ##
R> ## Load the 'rrcovNA' package and the data sets to be
R> ## used throughout the examples
R> ##
R> library("rrcovNA")
R> data("bush10")
```

Most of the multivariate statistical methods are based on estimates of multivariate location and covariance, therefore these estimates play a central role in the framework. We will start by computing the robust *minimum covariance determinant* estimate for the data set `bush10` included in the package **rrcovNA**. After computing its robust location and covariance matrix using the MCD method implemented in the function `CovNAMcd()` we can print the results by calling the default `show()` method on the returned object `mcd`. Additional summary information can be displayed with the `summary()` method. The standard output contains the robust estimates of location and covariance. The summary output (not shown here) additionally contains the eigenvalues of the covariance matrix and the robust distances of the data items (Mahalanobis type distances computed with the robust location and covariance instead of the sample ones).

```
R> ##
R> ## Compute MCD estimates for the modified bushfire data set
R> ## - show() and summary() examples
R> ##
R> mcd <- CovNAMcd(bush10)
R> mcd

Call:
CovNAMcd(x = bush10)
-> Method:  Minimum Covariance Determinant Estimator for incomplete data.

Robust Estimate of Location:
   V1     V2     V3     V4     V5
109.5  149.5  257.9  215.0  276.9

Robust Estimate of Covariance:
    V1       V2       V3       V4       V5
V1   697.6    489.3  -3305.1   -671.4   -550.5
V2   489.3    424.5  -1889.0   -333.5   -289.5
V3 -3305.1  -1889.0  18930.9   4354.2   3456.4
```

```
V4   -671.4   -333.5   4354.2   1100.1    856.0
V5   -550.5   -289.5   3456.4    856.0    671.7

R> summary(mcd)

R> ##
R> ## Example plot of the robust against classical
R> ##  distances for the modified bushfire data set
R> ##
R> plot(mcd)
```

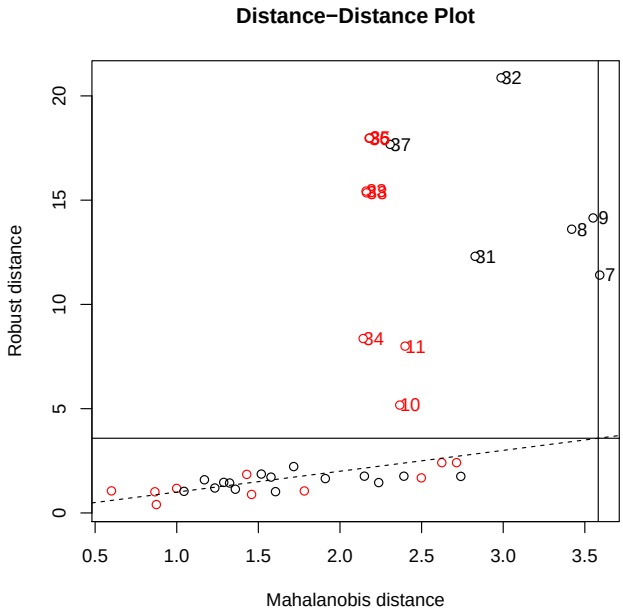


Figure 1: Example plot of the robust against classical distances for the modified `bushfire` data set (including missing values).

Now we will show one of the available plots by calling the `plot()` method—Figure 1 presents the Distance-Distance plot introduced by Rousseeuw and van Zomeren (1990) which plots the robust distances versus the classical Mahalanobis distances and allows to classify the observations and identify the potential outliers. The observations containing missing values are shown in a different colour. The description of this plot as well as examples of more graphical displays based on the covariance structure will be shown in Section 3.7. Apart from the demonstrated MCD method the package provides many other robust estimators of multivariate location and covariance for incomplete data. It is important to note that the format of the output and the graphs will be the same, regardless of which estimation method was used. For example, the following code lines will compute the S estimates for the same data set and provide the standard and extended output (not shown here).

```
R> ##
R> ## Compute the S-estimates for the modified bushfire data set
R> ##  bush10 and provide the standard and extended output
R> ##
R> est <- CovNASest(bush10, method="bisquare")
R> est
R> summary(est)
```

Nevertheless, the variety of methods could pose a serious hurdle for the novice and may even be quite tedious for the experienced user. Therefore, a shortcut is provided—the function `CovNARobust()` can be called with a parameter set specifying any of the available estimation methods, but if this parameter set is omitted the function will select a method on the basis of the data size. As shown in the example below, the function selected the Stahel-Donoho estimates in this case. For details and further examples see Section 3.6.

```
R> ##
R> ## Automatically select the appropriate estimator according
R> ##  to the problem size - in this example the Stahel-Donoho
R> ##  estimates will be selected.
R> ##
R> est <- CovNARobust(bush10)
R> est

Call:
CovSde(x = x, control = obj)
-> Method:  Stahel-Donoho estimator

Robust Estimate of Location:
   V1     V2     V3     V4     V5
102.4  145.6  298.4  224.1  284.0

Robust Estimate of Covariance:
    V1       V2       V3       V4       V5
V1   974.1    713.7  -5257.1  -1339.9  -1061.2
V2   713.7    587.8  -3487.3   -870.5   -693.9
V3 -5257.1  -3487.3  31954.6   8408.5   6645.5
V4 -1339.9   -870.5   8408.5   2275.7   1786.1
V5 -1061.2   -693.9   6645.5   1786.1   1412.3
```

3.6 Object model and implementation details

The object model for the S4 classes and methods implementing the different multivariate location and scatter estimators for incomplete data follows the proposed class hierarchy given in Todorov and Filzmoser (2009). The abstract class `CovNA` serves as a

base class for deriving all classes representing classical and robust location and scatter estimation methods. It defines the common slots and corresponding accessor methods, provides implementation for the general methods like `show()`, `plot()` and `summary()`. The slots of `CovNA` hold some input or default parameters as well as the results of the computations: the location, covariance matrix and the distances. The `show()` method presents brief results of the computations and the `summary()` method returns an object of class `SummaryCovNA` which has its own `show()` method. These slots and methods are defined and documented only once in this base class and can be used by all derived classes. Whenever new data (slots) or functionality (methods) are necessary, they can be defined or redefined in the particular class.

The classical location and scatter estimates for incomplete data are represented by the class `CovNAClassic` which inherits directly from `CovNA` (and uses all slots and methods defined there). The function `CovNAClassic()` serves as a constructor (generating function) of the class. It can be called by providing a data frame or matrix. As already demonstrated in Section 3.5 the methods `show()` and `summary()` present the results of the computations. The `plot()` method draws different diagnostic plots which are shown in one of the next sections. The accessor functions like `getCenter()`, `getCov()`, etc. are used to access the corresponding slots. Another abstract class, `CovNARobust` is derived from `CovNA`, which serves as a base class for all robust location and scatter estimators. The classes representing robust estimators like `CovNAMcd`, `CovNASest`, etc. are derived from `CovNARobust` and provide implementation for the corresponding methods. Each of the constructor functions `CovNAMcd()`, `CovNAOgk()` and `CovNASest()` performs the necessary computations and returns an object of the class containing the results. Similarly to the `CovNAClassic()` function, these functions can be called either with a data frame or a numeric matrix.

3.6.1 Generalized estimation function

The variety of estimation methods available for incomplete data, each of them with different parameters, as well as the object models described earlier in this section can be overwhelming for the user, especially for the novice who does not care much about the technical implementation of the framework. One function is therefore provided which gives quick access to the robust estimates of the location and covariance matrix for incomplete data. The class `CovNARobust` is abstract (defined as *VIRTUAL*) and no such objects can be created but any of the classes derived from `CovNARobust`, such as `CovNAMcd` or `CovNAOgk`, can act as an object of class `CovNARobust`. The function `CovNARobust()`, which technically is not a constructor function, can return an object of any of the classes derived from `CovNARobust` according to the user request. This request can be specified in one of three forms:

- If only a data frame or matrix is provided and the control parameter is omitted, the function decides which estimate to apply according to the size of the problem at hand. If there are less than 1000 observations and less than 10 variables or less

than 5000 observations and less than 5 variables, the Stahel-Donoho estimator will be used. Otherwise, if there are less than 50000 observations, either bisquare S estimates (in case of less than 10 variables) or Rocke type S estimates (for 10 to 20 variables) will be used. In both cases the S iteration starts at the initial MVE estimate. And finally, if there are more than 50000 observations and/or more than 20 variables the Orthogonalized Quadrant Correlation estimator (function `CovNAOgk()` with the corresponding parameters) is used. This is illustrated by the following example:

```
R> ##
R> ## Automatically select the appropriate estimator according
R> ##  to the problem size.
R> ##
R> genNAData <- function(n, ncol){
+     x<-rnorm(n); x[sample(1:n, size=0.1*n)] <- NA
+     matrix(x, ncol=ncol)
+ }
R> ## 20x2      - SDE
R> getMeth(CovNARobust(genNAData(n=40, ncol=2)))

[1] "Stahel-Donoho estimator"

R> ## 2000x8    - bisquare S
R> getMeth(CovNARobust(genNAData(n=1600, ncol=8)))

[1] "Stahel-Donoho estimator"

R> ## 2000x10   - Rocke S
R> getMeth(CovNARobust(genNAData(n=20000, ncol=10)))

[1] "S-estimates: Rocke type"

R> ## 100000x2  - OGK
R> getMeth(CovNARobust(genNAData(n=200000, ncol=2)))

[1] "Orthogonalized Gnanadesikan-Kettenring Estimator"
```

- The simplest way to choose an estimator is to provide a character string with the name of the estimator—one of `"mcd"`, `"ogk"`, `"s-fast"`, `"s-rocke"`, etc.

```
R> ##
R> ## Rocke-type S-estimates
```

```
R> ##
R>     getMeth(CovNARobust(matrix(rnorm(40), ncol=2),
+         control="rocke"))

[1] "S-estimates: Rocke type"
```

- If some of the estimation parameters need to be specified, the user can create a control object (derived from `CovControl`) and pass it to the function together with the data. For example, to compute the OGK estimator using the median absolute deviation (MAD) as a scale estimate and the quadrant correlation (QC) as a pairwise correlation estimate we create a control object `ctrl` passing the parameters `s_mad` and `s_qc` to the constructor function and then calling `CovNARobust` with this object.

```
R> ##
R> ## Specify some estimation parameters through a control
R> ##  object.
R> ##  The last command line illustrates the accessor method
R> ##  for getting the correlation matrix of the estimate
R> ##  as well as a nice formatting method for covariance
R> ##  matrices.
R> ##
R>     data("toxicity")
R>     ctrl <- CovControlOgk(smrob = "s_mad", svrob = "qc")
R>     est <- CovNARobust(toxicity, ctrl)
R> ##    round(getCenter(est),2)
R> ##    as.dist(round(getCorr(est), 2))
```

For more details, see the description of the function `CovRobust()` for complete data in Todorov and Filzmoser (2009).

3.6.2 Robust PCA for incomplete data

The object model for the S4 classes and methods implementing the principal component analysis methods follows the proposed class hierarchy given in Todorov and Filzmoser (2009), but for simplicity the number of classes is reduced and the different estimation methods are specified by a parameter. The abstract class `PcaNA` (derived from `Pca` in package **rrcov**) serves as a base class for deriving all classes representing classical and robust principal components analysis methods. It defines the common slots and the corresponding accessor methods and provides implementation for the general methods like `show()`, `plot()`, `summary()` and `predict()`. The slots of `PcaNA` hold some input or default parameters like the requested number of components as well as the results

of the computations: the eigenvalues, the loadings and the scores. The `show()` method presents brief results of the computations, and the `predict()` method projects the original or new data to the space spanned by the principal components. It can be used either with new observations or with the scores (if no new data are provided). The `summary()` method returns an object of class `SummaryPca` which has its own `show()` method. As in the other sections of the package these slots and methods are defined and documented only once in this base class and can be used by all derived classes. Whenever new information (slots) or functionality (methods) are necessary, they can be defined or redefined in the particular class.

Classical principal component analysis for incomplete data is represented by the class `PcaNA` withe `method="class"` which inherits directly from `Pca` (and uses all slots and methods defined there). The function `PcaNA()` serves as a constructor (generating function) of the class. It can be called either by providing a data frame or matrix or a formula with no response variable, referring only to numeric variables. Let us consider the following simple example with the data set `bush10` containing missing values. The code line

```
R> PcaNA(bush10, method="class")
```

can be rewritten as (and is equivalent to) the following code line using the formula interface

```
R> PcaNA(~ ., data = bush10, method="class")
```

The function `PcaNA()` with `method="class"` performs the standard principal components analysis and returns an object of the class `PcaNA`.

```
R> ##
R> ## Classical PCA
R> ##
R> pca <- PcaNA(~., data=bush10, method="class")
R> pca

Call:
PcaNA(formula = ~., data = bush10, method = "class")

Standard deviations:
[1] 163.679611  27.335832  16.573119   8.417495   1.502502
```

```
Loadings:
           PC1         PC2        PC3        PC4          PC5
V1 -0.02659665  0.40918484  0.4023615  0.8184958 -0.005503999
V2 -0.01525114  0.90453802 -0.2930261 -0.3087768 -0.019260616
V3  0.90576986 -0.02651191 -0.3610926  0.2202021  0.001101088
V4  0.32660506  0.07626469  0.6095852 -0.3314624 -0.637221583
V5  0.26827246  0.08865405  0.5002589 -0.2763426  0.770419481
```

```
R> summary(pca)
```

```
Call:
PcaNA(formula = ~., data = bush10, method = "class")
Importance of components:
                            PC1     PC2      PC3     PC4     PC5
Standard deviation     163.6796 27.3358 16.57312 8.41749 1.50250
Proportion of Variance   0.9607  0.0268  0.00985 0.00254 0.00008
Cumulative Proportion    0.9607  0.9875  0.99738 0.99992 1.00000
```

```
R> plot(pca)
```

The `show()` method displays the standard deviations of the resulting principal components, the loadings and the original call. The `summary()` method presents the importance of the calculated components. The `plot()` draws a PCA diagnostic plot which is presented and described later. The accessor functions like `getLoadings()`, `getEigenvalues()`, etc. are used to access the corresponding slots, and `predict()` is used to rotate the original or new data to the space of the principle components.

The robust PCA methods are performed by supplying the corresponding parameter to the function `PcaNA()` and correspond to the complete data methods `PcaHubert`, `PcaLocantore`, etc. derived from `PcaRobust` in **rrcov**. The constructor function `PcaNA()` with the corresponding parameter `method=c("locantore", "hubert", "grid", "proj", "class", "cov")` performs the necessary computations and returns an object of the class containing the results. In the following example the same data are analysed using a projection pursuit method.

```
R> ##
R> ## Robust PCA
R> ##
R> rpca <- PcaNA(~., data=bush10, method="grid", k=3)
R> rpca
```

```
Call:
PcaNA(formula = ~., data = bush10, method = "grid", k = 3)
```

```
Standard deviations:
[1] 134.50391  24.94751   4.79467

Loadings:
          PC1        PC2        PC3
V1 0.01248127  0.5058076 -0.1470630
V2 0.12643303  0.7822247 -0.3153327
V3 0.87902227 -0.2551691 -0.3964164
V4 0.35099686  0.1952972  0.6683505
V5 0.29661415  0.1703844  0.5244993
```

```
R> ##summary(rpca)
```

3.7 Visualization of the results

The default plot accessed through the method `plot()` of class `CovNARobust` is the Distance-Distance plot introduced by Rousseeuw and van Zomeren (1990). An example of this graph which plots the robust distances versus the classical Mahalanobis distances is illustrated in Figure 1. The dashed line represents the points for which the robust and classical distances are equal. The horizontal and vertical lines are drawn at values $x = y = \sqrt{\chi^2_{p,0.975}}$. Points beyond these lines can be considered outliers and are identified by their labels. All observations which have at least one missing value are shown in red.

The other available plots are accessible either interactively or through the `which` parameter of the `plot()` method. Figure 2 shows the pair-wise correlations (`which="pairs"`) computed classically as the sample correlation coefficients (excluding the pair-wise missing values) and computed robustly by applying the Minimum Covariance Determinant (MCD) method for incomplete data. In the upper triangle the corresponding ellipses represent bivariate normal density contours with zero mean and unit variance together with a bivariate scatter plot of the data. The observations which have a missing value in any of the coordinates are projected on the axis and are shown in red. The lower triangle presents classical and robust correlation coefficients. A large positive or negative correlation is represented by an elongated ellipse with a major axis oriented along the ± 45 degree direction while near to zero correlation is represented by an almost circular ellipse. The differences between the classical and robust estimates are easily distinguishable.

```
R> mcd <- CovNAMcd(bush10)
R> plot(mcd, which="pairs")
```

The left panel of Figure 3 exemplifies the distance plot in which robust and classical Mahalanobis distances are shown in parallel panels. The outliers have large robust

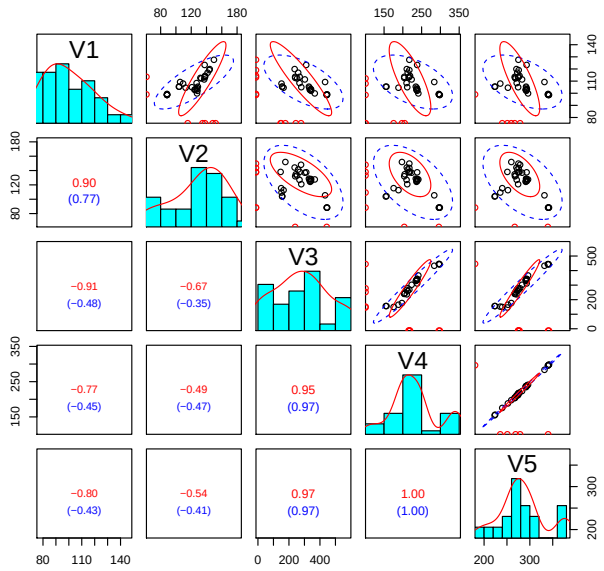


Figure 2: Classical and robust correlations and scatter plot matrix with tolerance ellipses.

distances and are identified by their labels. The right panel of Figure 3 shows a Quantile-Quantile comparison plot of the robust and the classical Mahalanobis distances versus the square root of the quantiles of the chi-squared distribution.

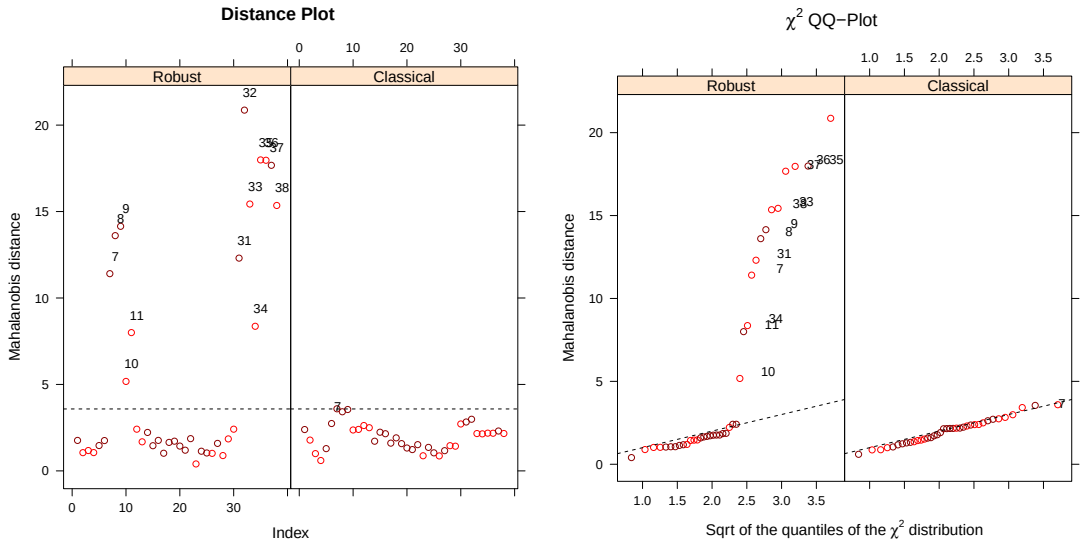


Figure 3: Distance plot and chi-square Q-Q plot of the robust and classical distances.

The next plot presented in Figure 4 shows a scatter plot of the data on which the 97.5% robust and classical confidence ellipses are superimposed. The observations with distances larger than $\sqrt{\chi^2_{p,0.975}}$ are identified by their subscript. In the right panel of

Figure 4 a `screeplot` of the `ces` data set is shown, presenting the robust and classical eigenvalues.

```
R> ##
R> ## a) scatter plot of the data with robust and classical
R> ##  confidence ellipses.
R> ## b) screeplot presenting the robust and classical eigenvalues
R> ##
R> data("bush10")
R> data("ces")
R> X <- bush10[,c(2,3)]
R> usr <- par(mfrow=c(1,2))
R> plot(CovNAMcd(X), which="tolEllipsePlot", classic=TRUE)
R> plot(CovNAMcd(ces), which="screeplot", classic=TRUE)
R> par(usr)
```

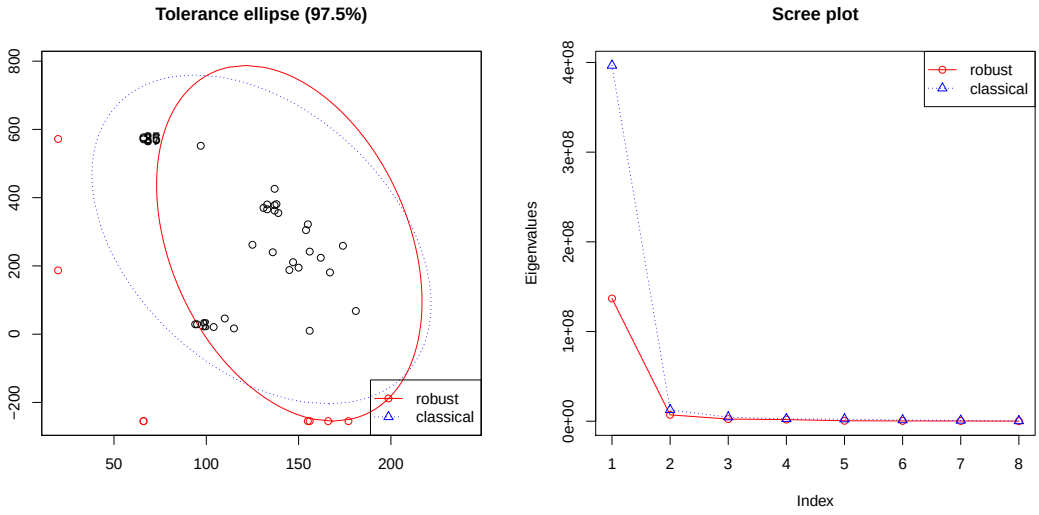


Figure 4: Robust and classical tolerance ellipse for two selected variables (V2 and V3) of the modified `bushfire` data and robust and classical scree plot for the Consumer Expenditure Survey data (`ces` data set).

In the context of PCA Hubert et al (2005) defined a *diagnostic plot* or *outlier map* which helps to distinguish between regular observations and different types of outliers. The diagnostic plot is based on the *score distances* and *orthogonal distances* computed for each observation. The *score distance* is defined by

$$SD_i = \sqrt{\sum_{j=1}^{k} \frac{t_{ij}^2}{l_j}}, \qquad i = 1, \ldots, n, \tag{9}$$

where t_{ij} is the element of the score matrix T. It measures the distance of each observation to the subspace spanned by the first k principal components. The *orthogonal*

distance is defined by

$$OD_i = ||x_i - m - Pt_i||, \qquad i = 1, \ldots, n \tag{10}$$

where t_i is the ith row of the score matrix T. This measure corresponds to the distance of the projection of each observation into the space spanned by the first k principal components. The *diagnostic plot* shows the score versus the orthogonal distance, and indicates the cut-off values with a horizontal and vertical line, which makes it possible to distinguish regular observations from the two types of outliers (for details, see Hubert et al, 2005). An example of the classical and robust diagnostic plot for the `bush10` data set is shown in Figure 5.

```
R> ##
R> ## An example of the classical and robust diagnostic
R> ##  plot for the bush10 data set
R> ##
R> usr<-par(mfrow=c(1,2))
R> plot(PcaNA(bush10, k=3, method="class"), main="Classical PCA")
R> plot(PcaNA(bush10, k=3, method="locantore"))
R> par(usr)
```

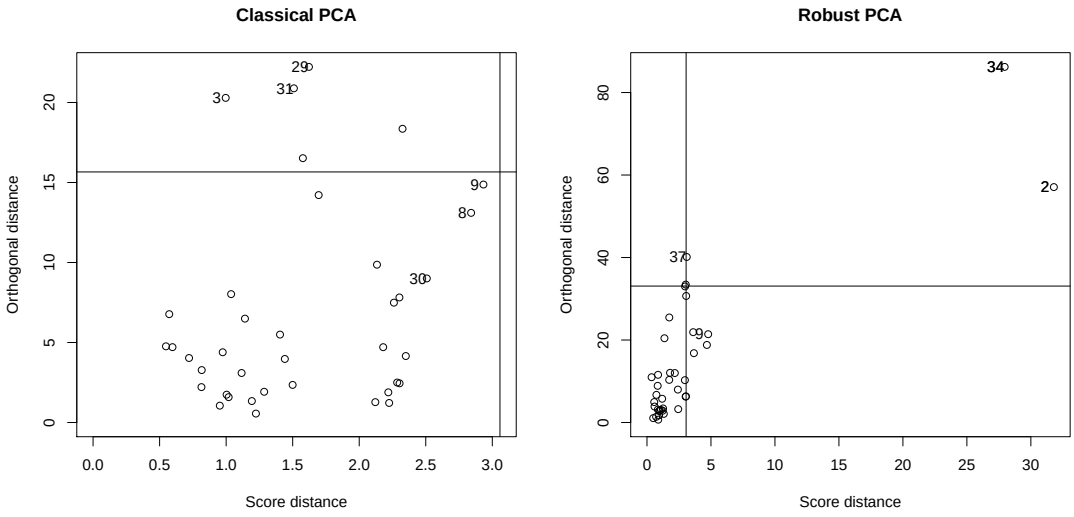


Figure 5: Classical and robust diagnostic plot for the `bush10` data with $k = 3$.

4 Visualization of missing values

Statistical graphics are an important component of modern data analysis and as pointed out by (Tukey, 1962, 1977) data has to be explored using graphics to identify possible problems even before applying models to the data or producing any statistics. When data include missing values most visualization tools fail because they are designed to analyse complete data sets. However, virtually all data sets in official statistics contain missing values due to measurement errors or non-response.

Useful tools for visualizing missing values and exploring the data as well as the structure of the missing values are available in the R package **VIM** (Templ and Alfons, 2011). Depending on the structure of the missing values, these tools may help to identify the mechanism generating the missing values. **VIM** stands for "Visualisation and Imputation of Missing Values". The package was written by Matthias Templ, Andreas Alfons and Alexander Kowarik and is maintained by Matthias Templ. Package **VIM** allows to explore and analyse data which contains missing values. It is possible to explore the structure of missing values, as well as to produce high-quality graphics for publications.

In the following examples the R package **VIM** is applied to INDSTAT 2 edition 2010, the *UNIDO Industrial Statistics Database*. The INDSTAT 2 database contains time series data from 1963 onwards. Data are available by country, year and ISIC (International Standard Industrial Classification of All Economic Activities) Revision 3 at the 2-digit level. All variables containing monetary value data are originally stored in national currency at current prices. The system allows for data conversion from national currency into current US dollars using the average period exchange rates as given in the IMF/IFS (International Monetary Fund/International Financial Statistics). The database contains seven principle indicators of industrial statistics:

1. Number of establishments (EST). An "establishment" is ideally a unit that engages in one or in predominantly one kind of activity at a single location under single ownership or control, for example, a workshop or factory.

2. Number of employees (EMP). The number of employees includes all persons engaged other than working proprietors, active business partners and unpaid family workers.

3. Wages and salaries (WS) (at current prices). Wages and salaries include all payments in cash or in kind paid to "employees" during the reference year in relation to work performed for the establishment.

4. Gross output (GO) (at current prices). The measure of gross output normally reported is the census concept, which covers only activities of an industrial nature. The value of census output in the case of estimates compiled on a production basis comprises: (a) the value of sale of all products of the establishment; (b) the net change between the beginning and the end of the reference period in the value

of work in progress and stocks of goods to be shipped in the same condition as received; (c) the value of industrial work done or industrial services rendered to others; (d) the value of goods shipped in the same condition as received less the amount paid for these goods; and (e) the value of fixed assets produced during the period by the unit for its own use. In the case of estimates compiled on a shipment basis, the net change in the value of stocks of finished goods between the beginning and the end of the reference period is also included.

5. Value added (VA) (at current prices). The measure of value added normally reported is the census concept, which is defined as the value of census output less the value of census input, which covers: (a) value of materials and supplies for production (including cost of all fuel and purchased electricity); and (b) cost of industrial services received (mainly payments for contract and commission work and repair and maintenance work).

6. Gross fixed capital formation (GFCF). Gross fixed capital formation refers to the value of purchases and own account construction of fixed assets during the reference year less the value of corresponding sales.

7. Number of female employees (EMPF).

8. Index of Industrial Production (IIP). The index of industrial production is designed to show the real (i.e. free of price fluctuation) change of production. Currently, its base year is 2000.

For the purpose of our examples two auxiliary variables were included in the data set:

1. Consumer Price Index (CPI) (with base year 2000)

2. Index of Manufacturing Value Added (total manufacturing). The manufacturing value added (MVA) is a measure of the total value of goods and services produced by the manufacturing sector as defined in ISIC Revision 3. MVA is calculated as the difference of gross output and intermediate consumption according to the concepts recommended in the system of national accounts (SNA). MVA represents the part of GDP originating from manufacturing activities The MVA data are at constant 2000 prices at the country level and are taken from from the national accounts data base. It has already been treated for missing values at the end of the time series using nowcasting methods developed by Boudt et al (2009). The MVA data from national accounts data can differ from the MVA data in the INDSTAT database. Missing data in one database can therefore not be replaced with the corresponding available observation in the other database.

Table 1 presents a brief description and numerical summary of the selected variables including the arithmetic mean and median of all values of the corresponding variables.

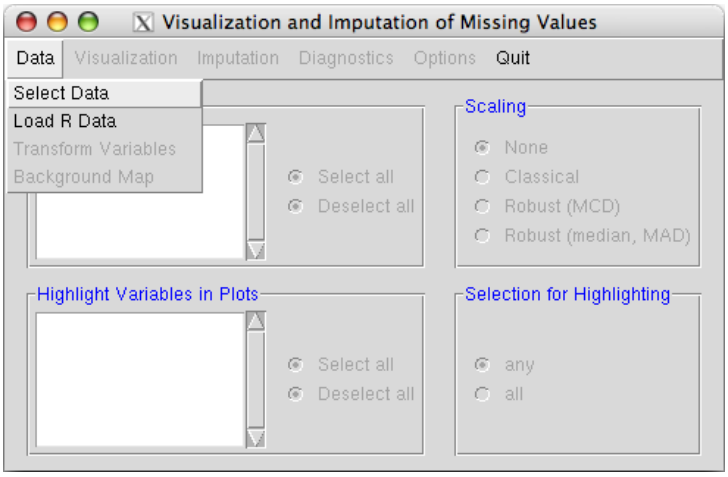


Figure 6: The **VIM** GUI and the *Data* menu.

4.1 The graphical user interface of VIM

The graphical user interface (GUI) of the package **VIM** has been developed using the R package **tcltk** (R Development Core Team, 2011) and allows easy handling of the functions included in the package **VIM**. Figure 6 shows a screenshot of the main window, which pops up automatically after loading the package. The most important menus for visualization are *Data*, *Visualization* and *Options*.

```
library(VIM)
vmGUImenu()
```

Listing 5: Loading the GUI of VIM.

4.2 Importing data and selection of variables

The menu *Data* allows selecting a data frame from the R workspace. In addition, a data set in `.RData` format can be imported from the file system into the R workspace, which is then loaded into the GUI directly. After a data set has been chosen, variables can be selected in the main dialog (see Figure 7). One important feature is that the variables are used for plotting in the same order as they were selected, which is especially useful for parallel coordinates plots (Wegman, 1990; Venables and Ripley, 2003).

Variables are selected based on two criteria:

1. In the *Select variables* menu (see Figure 7). Variables chosen here are plotted on a graph.

2. In the *Highlight variables* menu (see Figure 7). Variables chosen here determine the colour of points (or lines or bars) in the plot, whereby one colour is assigned to the observations with no missing values in the selected variables and another to those observations which have one missing value in those variables (or, optionally,

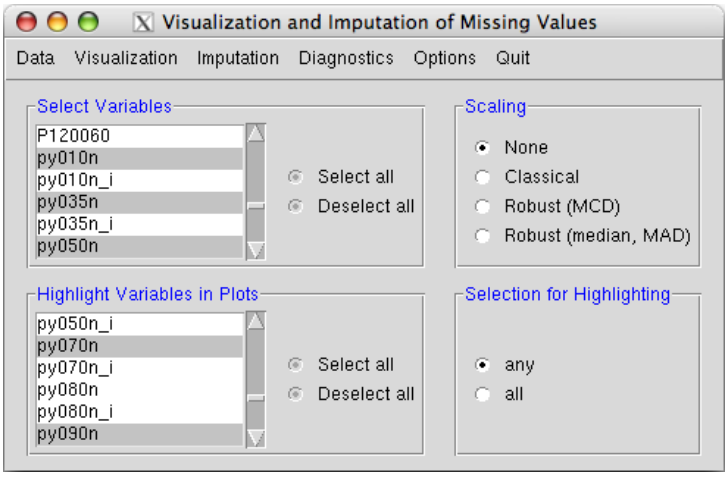


Figure 7: Variable selection with the **VIM** GUI.

if all values are missing in those variables). If more than one variable is chosen for highlighting, it is possible to select whether observations with missing values in any or in all of these variables should be highlighted (see the lower right frame in Figure 7).

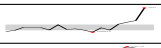
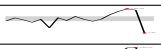
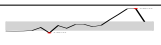
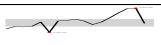
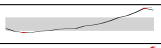
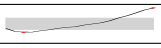
Transformations of variables can be applied via the menu `Data` → `Transform Variables`. The transformed variables are thereby appended to the data set in use. Most of the commonly used transformations in official statistics are available, such as the Box-Cox transformation (Box and Cox, 1964) and the log-transformation as an important special case of the Box-Cox transformation. In addition, several other transformations that are frequently used for compositional data (Aitchison, 1986) are implemented. Variables can also be scaled. While transformed variables are added to the data set and become selectable in the variable selection menu, scaling is performed on-the-fly, i.e. the scaled variables are simply passed to the underlying plot functions (they are not permanently stored). Background maps and coordinates for spatial data can be selected in the *Data* menu as well.

Depending on how many variables are selected for plotting, plotting methods can be selected from the *Visualization* menu. Plots that are not applicable with the selected variables are disabled, e.g. if only one plot variable is selected, multivariate plots are not available.

4.3 An example session

In order to demonstrate some of the features of the R package **VIM** we will use data from INDSTAT 2, the *Industrial Statistics Database of UNIDO*. Table 1 provides a brief description and a numerical summary of those variables including the arithmetic mean and median of all values of the corresponding variables. A sparkline (Tufte, 2010) in the last column shows the trend of the arithmetic mean of each variable. When

Table 1: Overview of INDSTAT2 2010 ISIC Revision 3 data set and the auxiliary variables selected for the example session.

Label	Brief Description	Mean	Median	Sparkline & IQR
CountryCode	Country Code			
Year	1991 – – – 2008			
Employment	Number of Employees	1775863	293321.5	
WagesSalaries	Wages and Salaries	21149277971	1594202976	
Output	Gross output: measured by the census concept, which covers only activities of an industrial nature	167768482087	16635259487	
ValueAdded	Value added: value of census output less the value of census input	64463768091	6768179752	
IIP	Index of Industrial Production	105.20	100	
CPI	Consumer Price Index	99.29	99.02	
IMVA	Index of MVA	103.71	99	

only one variable is selected, only univariate plots can be applied. Standard univariate plots, such as bar plots and spine plots (a special case of mosaic plots which can be considered a generalization of stacked or highlighted bar plots) for categorical variables and histograms, spinograms (an extension of histograms) and different types of box plots for continuous variables have been adapted to display information for incomplete data.

Univariate missingness year by year: We can study the missingness of the variables year by year using the *spine plot* (see Hummel, 1996) taking the year as a categorical variable and displaying the proportion of missing and observed values in the variables. Stacked bars are drawn with vertical extent showing the proportion of both missing values (red) and observed ones (blue). This plot can be created by the menu `Visualization` → `Spine Plot with Missings` or through the function `spineMiss()`. The *spine plot* in Figure 8 illustrates that the missing rates are almost constant over time from 1991 until 2003. After 2003, the missing rates increase. The bar at the right-hand side provides the mean proportion of missing values over all years. Figure 9 shows the missing rates for the auxiliary variables. We observe that variable IIP behaves differently in year 2000, but this is not surprising since 2000 is the base year and IIP has a value of 100 percent whenever there was production. The missing rate for `CPI` is almost constant over time with the exception of a structural break from 2001 onwards. The missing rate is almost negligible for variable `IMVA` (data are only missing for a few countries).

Study of dependence in missingness across variables: How many missing values are contained in the single variables is often of interest. What is even more interesting is that there can be certain combinations of variables with a high number of missing values. We could perform this analysis in cross-sections of the data, for example, for a particular year. Let us use the year 2006. We select the required variables (see Figure 7) and create the graph using the menu `Visualization` → `Aggregate Missings`. Figure 10 shows the proportions and combinations. 55 observations do not contain any missing values while 21 observations include missing values in the first four variables, i.e. the

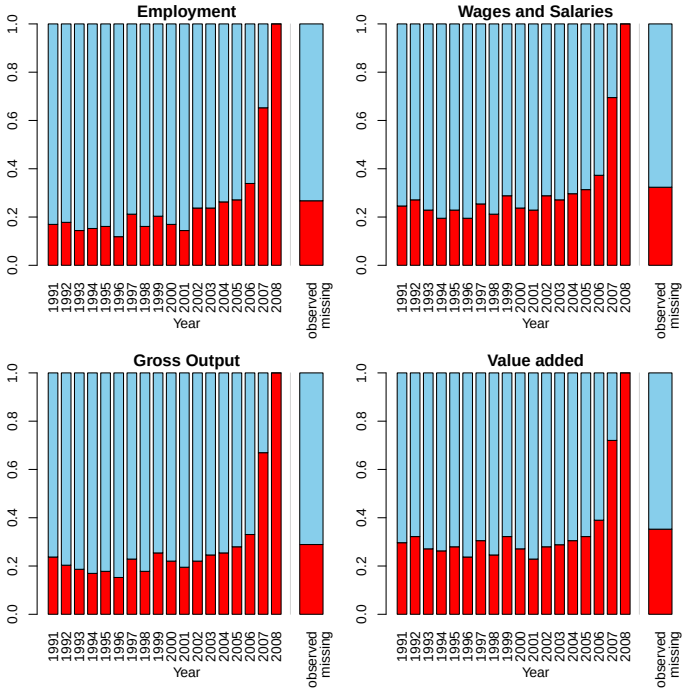


Figure 8: Spine plots for the main variables of the INDSTAT2 data.

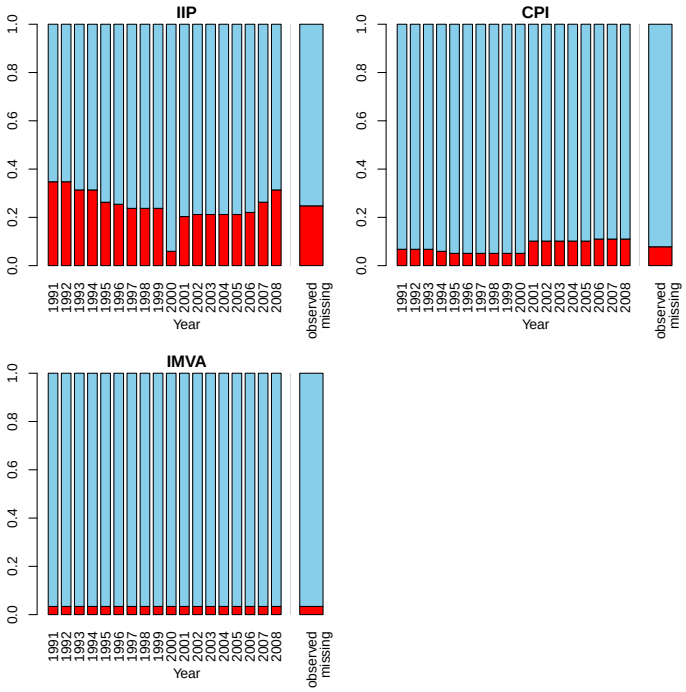


Figure 9: Spine plots for the auxiliary variables.

corresponding countries did not provide any value for the first four (21 observations) or first five (10 observations) variables. Only few missing values can be considered as item non-responses where only one (or few) values are missing while the others are available.

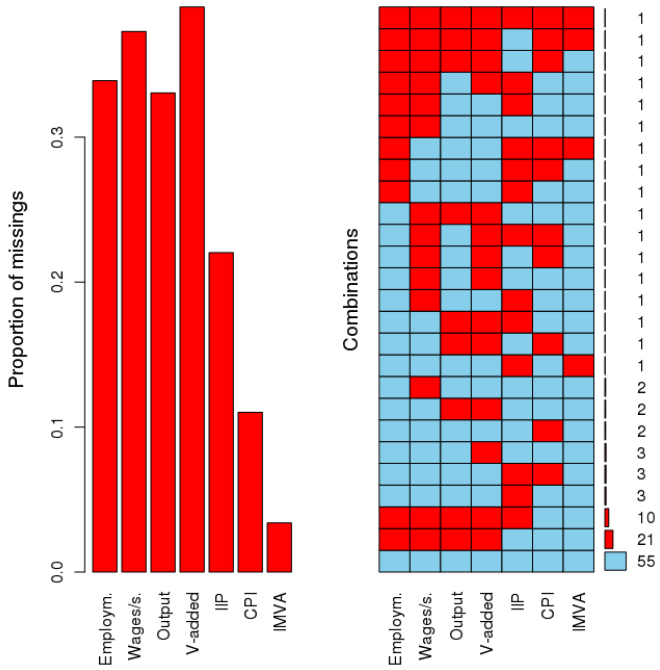


Figure 10: Simple aggregations of INDSTAT2 data for the year 2006.

Using the matrix plot: The above result is even better visible in the *matrix plot* shown in Figure 11. The *matrix plot* is a powerful multivariate plot. All cells of the data matrix are represented by (small) rectangles. In the example in Figure 11, red rectangles correspond to missing values and a grey scale is used for the available data. To determine the grey level, the variables are first scaled to the interval $[0, 1]$. Small values are assigned a light grey and high values a dark grey (0 corresponds to white, 1 to black). In addition, the observations can be sorted by the magnitude of a selected variable, which can also be done interactively by clicking in the corresponding column of the plot. Using the GUI, a matrix plot can be produced by selecting `Visualization` → `Matrix Plot`. Figure 11 shows a matrix plot of all variables, sorted by the variable *Employee* (left panel) and by variable *IIP* (right panel). It demonstrates again that the missing values occur on the same rows in the first four variables (monotone missingness). This reflects the structure of the missing values: the countries either report all their values or all values are missing. It is also interesting that *IIP* values contain nearly no missing values for establishments with a mid-size number of employees.

Parallel box plots: Figure 12 presents an example of *parallel box plots* for the variable *Wages and Salaries* which was log-transformed (base 10). In addition to a standard boxplot (left), boxplots grouped by observed (blue) and missing (red) values in other variables are drawn. Furthermore, the frequencies of the missing values are represented by numbers in the bottom margin. The first line corresponds to the observed values in *Wages and Salaries* and their distribution among the different groups, and the second line represents the missing values. It can be easily seen that the missing values in the

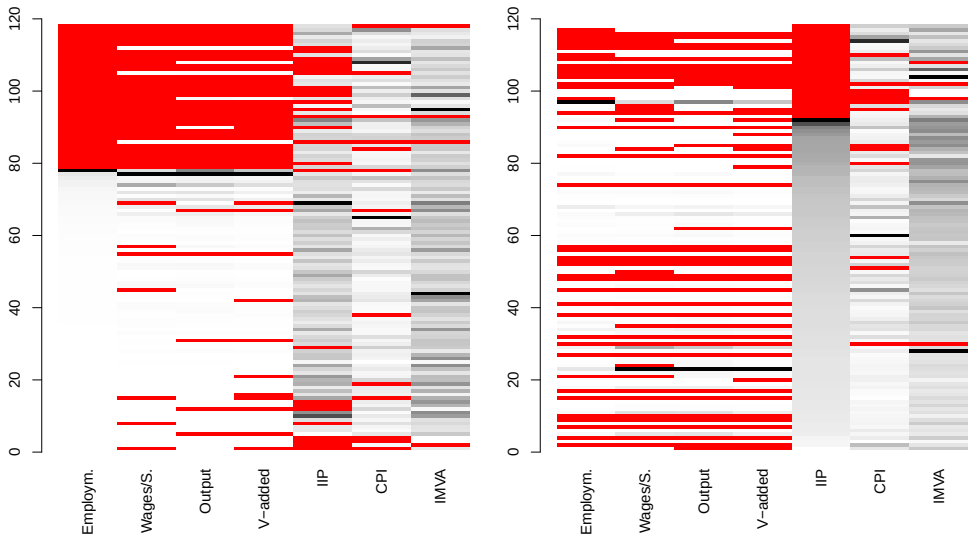


Figure 11: Matrix plot of the INDSTAT data (2006) ordered by the variable Employment (left panel) and the value of IIP (right panel).

variable *Employment* predominantly occur for low values of *Wages and Salaries*. This indicates a situation in which the missing values in *Employment* are dependent on the values of the variable *Wages and Salaries*. This situation is referred to as *Missing At Random (MAR)*. This is also true for the other variables. Special care needs to be taken

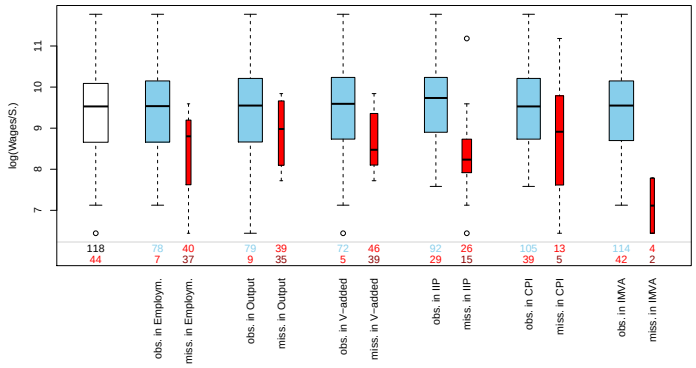


Figure 12: Parallel box plot of *Wages and Salaries* split by the occurrences of missing values in other variables.

when imputation of such data is performed.

4.4 Other graphical functions

Various other plots are available in the package which can also be run from within the GUI. We will illustrate some of these using the examples given in Templ and Filzmoser (2008), which are based on a subset of the European Survey of Income and Living Conditions (EU-SILC) from Statistics Austria of 2004. This well known, complex data set is used to measure poverty and monitor the Lisbon 2010 strategy of the European

Union - see http://en.wikipedia.org/wiki/Lisbon_Strategy. The data set contains a large number of missing values which have been imputed with model-based imputation methods. Since we cannot expect that the generation mechanism of the missing values is *Missing Completely at Random (MCAR)*, we have to decide which variables to include for imputation. The considered visualization tools are helpful for this type of decision.

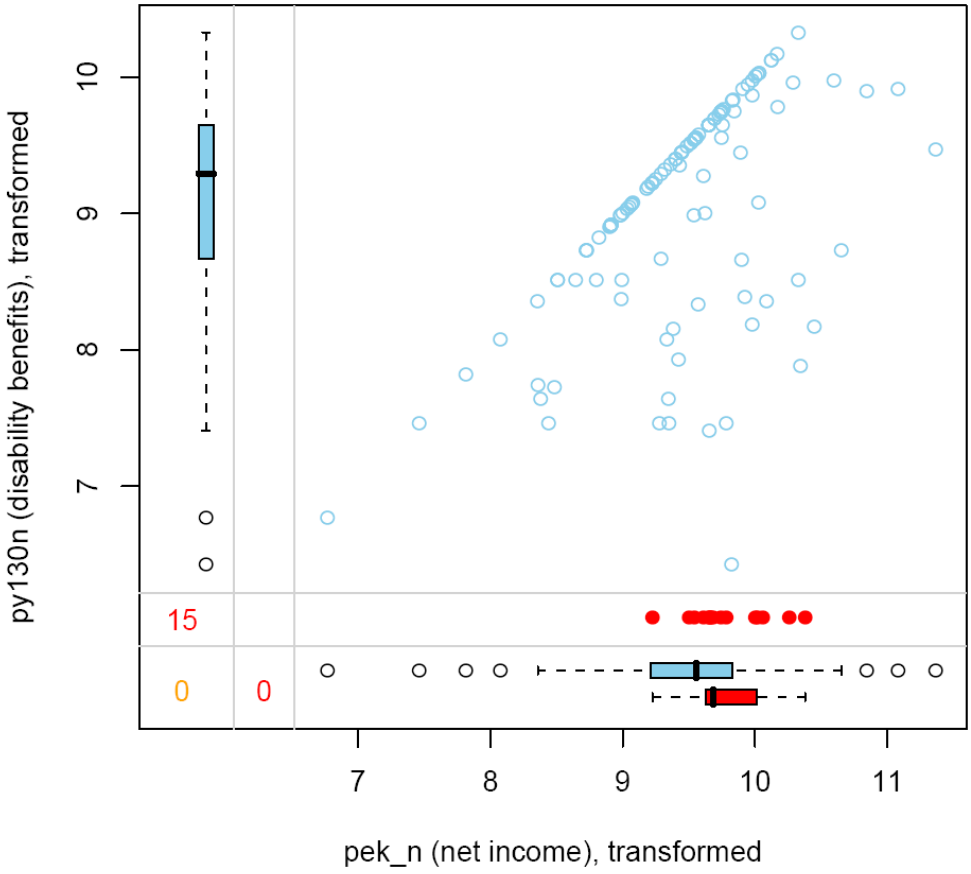


Figure 13: Scatter plot of the variable `pek_n` (net income) and py130n (unemployability) with missing data information in the plot margins.

Scatter plots for incomplete data. Information about missing values can be added to a standard *scatter plot* which shows the data values for a pair of variables. Figure 13 provides an example from Templ and Filzmoser (2008) using the variables `pek_n` (net income) and `py130n` (unemployability) of the EU-SILC data presented in a scatter plot. In addition, box plots for missing (red) and available (blue) data are shown in the outer plot margins. For example, along the horizontal axis the two parallel box plots represent the variable net income, while the red box plot denotes those values of the net income where no values for unemployability are available, and the blue box plot represents the net income values where information for unemployability is available as well. A comparison of the two box plots can indicate the missing data mechanism. In this example the net income for persons who provided no information for unemployability seems to be higher than that where information is available. The plot also shows a

univariate scatter plot of the values that are missing in the second variable (red points) and the frequency of these values by a number (lower left). The number in the lower left corner (here 0) is the number of observations missing in both variables. This kind of bivariate scatter plot can easily be extended to a *scatter plot matrix* representing all combinations of a set of variables.

Parallel coordinates plot. A parallel coordinates plot is a visualization technique for multivariate data used to plot individual data elements across many dimensions (Wegman, 1990; Venables and Ripley, 2003). Similarly to the previously described plots, the information of missingness of another variable can be colour-coded. Figure 14 shows a parallel coordinates plot of a subset of the EU-SILC data where the colour of the lines refers to observations which are missing (red) or available (blue) for the variable `py050n` (employees' income). The amount of missing values in `py050n` is related to several of the presented variables. In Figure 14 we can see that a high proportion of small values in

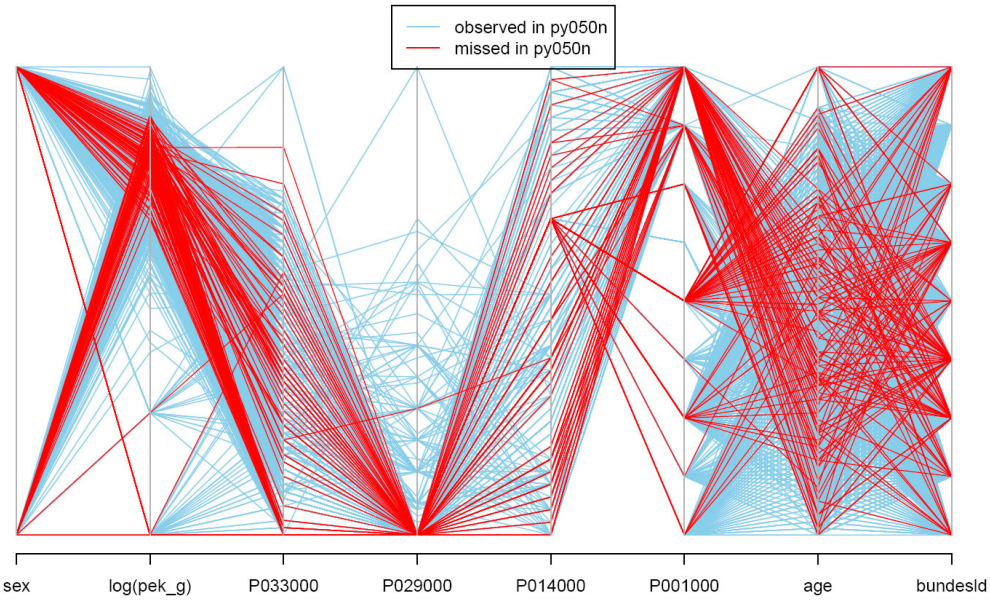


Figure 14: Parallel coordinates plot for a subset of the EU-SILC data (source: Templ and Filzmoser (2008)). The colour indicates missing values in variable `py050n` (employees' income).

`P033000` (years of employment) implies missing values in `py050n`. Additionally, missing values in `py050n` only occur for employees who have more than one employment (in this case the values were set to 0 in variable `P029000`). Furthermore, Figure 14 shows that the amount of missing values depends on the actual values for the variables `P001000` (different employment situation) and `bundesld` (province). Finally, for variable `pek_g` (gross income) missing values only occur in a certain range.

Presenting missing values on geographical maps. If geographical coordinates are available for a data set, it can be interesting to check whether missingness of a variable corresponds to spatial patterns on a map. For example, the observations of

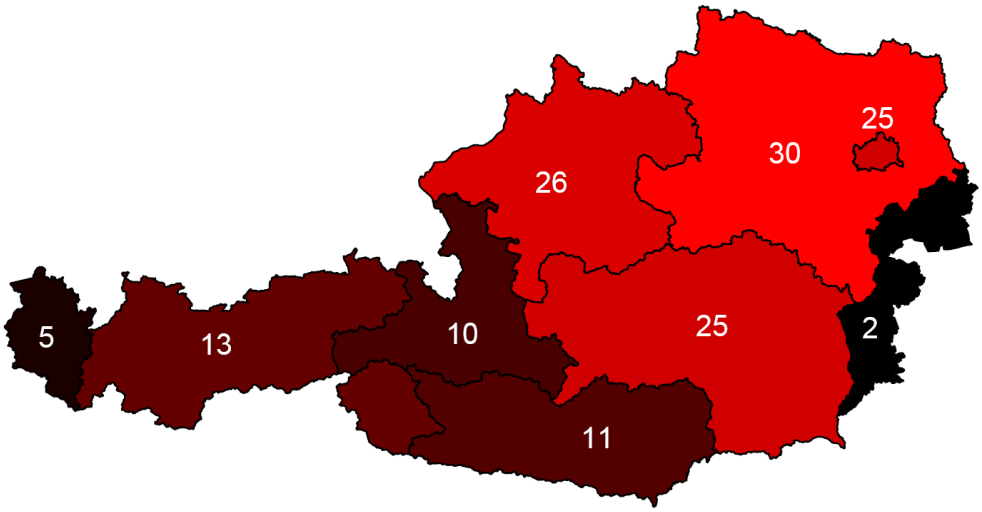


Figure 15: Presenting missing values on a map (source: Templ and Filzmoser (2008)). The number of missing values in the variable `py050n` (employees' income) for the nine provinces of Austria are presented. Regions with a higher amount of missing values (see the numbers in each province) are presented by a brighter shade red.

a variable could be drawn by growing dots on the map, reflecting the magnitude of the data values, and missingness in a second variable could be colour-coded. This allows conclusions about the relation of missingness to the values of the variable and to the spatial location. Figure 15 provides an example from Templ and Filzmoser (2008) presenting the EU-SILC data for which only the assignment of each observation to the nine provinces of Austria is available, and not the spatial coordinates. Therefore, we can only visualize the amount of missing values of a variable in the map. Figure 15 shows the map of Austria with its nine provinces. The number of missing values of the variable `py050n` (employees' income) is coded according to an *rgb* colour scheme, resulting in a brighter shade of red for provinces with a higher number of missing values for the given variable. Additionally, the numbers of missing values in the provinces are indicated.

4.5 Summary

Most of the graphical tools available in the package **VIM** described in this section, can be accessed from the main menu but can at the same time be invoked by the corresponding functions from the command line or an R script. In addition, many interactive features are implemented in the plot functions in order to allow easy modification of the generated plots. The following list presents the main functions of the package:

- `aggr()` - Aggregations for missing values
- `barMiss()` - Bar plot with information about missing values
- `bubbleMiss()`, `growdotMiss()` - Growing dot map with information about missing values

- `histMiss()` - Histogram with information about missing values
- `marginplot()` - Scatter plot with additional information in the margins
- `matrixplot()` - Matrix plot
- `mosaicMiss()` - Mosaic plot with information about missing values
- `pairsVIM()` - Scatter plot matrix
- `parcoordMiss()` - Parallel coordinates plot with information about missing values
- `pbox()` - Parallel box plots with information about missing values
- `rugBA()` - Rug representation of missing values
- `scattJitt()` - Bivariate jitter plot
- `scattmatrixMiss()` - Scatter plot matrix with information about missing values
- `scattMiss()` - Scatter plot with information about missing values
- `spineNiss()` - Spine plot with information about missing values
- Imputation functions
  - `hotdec()` - Hot-Deck Imputation
  - `kNN()` - k-Nearest Neighbour Imputation
  - `irmi()` - Iterative robust model-based imputation (IRMI)

Further examples of the application of the package **VIM** to official statistics data can be found in the *vignette* of the package (see Listing 6) which presents an application of **VIM** for analysing a highly complex data set, the European Statistics on Income and Living Conditions (EU-SILC).

```
library(VIM)
vignette("VIM-EU-SILC")
```

Listing 6: Calling the documentation of the package VIM.

Another example of the application of the package used for determining administrative data quality can be found in (Daas et al, 2010). For a detailed description of the functions see Templ and Filzmoser (2008).

5 Statistical disclosure control

One underlying principle of data providers is that values of statistical units such as enterprises or persons are strictly confidential. The United Nations Economic Commission for Europe (UNECE) adopted the *Fundamental Principles for Official Statistics* in the UNECE region. The United Nations Statistical Commission adopted these principles in 1994 at the global level. With regard to the confidentiality of the collected data the *Fundamental Principles for Official Statistics* state that

> *"Individual data collected by statistical agencies for statistical compilation, whether they refer to natural or legal persons, are to be strictly confidential and used exclusively for statistical purposes."*

This is not the only reason why organizations such as national statistical offices or international organizations like UNIDO should meet these requirements. National laws on data privacy must also be respected.

Statistical Disclosure Control (SDC) techniques can be defined as the set of methods to reduce the risk of disclosing information on individuals, businesses or other organizations. Such methods are only related to the dissemination step and are usually based on restricting the amount of or modifying the data released (see EUROSTAT: http://epp.eurostat.ec.europa.eu/portal/page/portal/research_methodology/methodology/statistical_disclosure_control SDC consists of methods which can be categorized as two different concepts: (i) the anonymization of microdata and (ii) the anonymization of cells in tables. However, tables generated from source perturbed microdata can also fulfil confidentiality requirements. Therefore, microdata perturbation can also be deemed a technique for publishing confidential tables.

Microdata release: The demand for high quality microdata for analytical purposes has increased rapidly among researchers and the public over the last few years. In order to respect existing laws on data privacy and to be able to provide microdata to researchers and the public, statistical institutes, agencies and other institutions may provide masked data. The aim and in many cases the legal obligation of data holders who want to disseminate microdata is to provide data for which it may only be possible to identify statistical units by disproportional costs and time resources. The aim of SDC is to reduce the risk of disclosing information on statistical units (individuals, enterprises, organizations) and, on the other hand, to provide as much information as possible by minimizing the amount of data modification.

```
# x . . .   a data set with a possible large number of variables
x[12876, c(3,21,6)]
         region  gender occupation
12867     4321    male   Univ.-Prof.
```

Listing 7: Example: Re-identifying a statistical unit.

In Example 7 a data set is used with 14365 observations including 300 variables. The observation with ID 12867 reports the following values on the three variables *region*, *gender* and *occupation*: a male person living in a region with an ID equal to *4321*, who is employed as a university professor. The region is in fact a small town in the rural area of Austria and many people might have the knowledge that only one professor who is male lives in that small town - this person is **unique** in the sample and can be **re-identified** easily. Re-identifying a person means that all information in the data set about this person is known to the intruder. In Example 7 this means that a data intruder knows the other 297 values of that person which might include very sensitive information like information about taxes or cancer.

This example shows that it does not suffice to delete only **direct identifiers**, i.e. variables like social security number, IDs, addresses, etc., which allow direct identification of any statistical unit in the sample. In addition to the deletion of direct identifiers, the combination of values of some **indirect identifiers** leads to disclosure problems. Those indirect identifiers which data intruders might have information to are called **key variables** for statistical disclosure control. The anonymization of these key variables is discussed in Section 5.3.

Publication of tables: Statistical agencies generally do not publish microdata, but disseminate information in the form of aggregated data. Aggregated data are usually represented as statistical tables with totals in the margins. Even though statistical tables present aggregated information of individuals contributing to the table cells, the risk of identifying single statistical units using tables is present and can, in fact, be high. Section 5.5 describes methods how to avoid the disclosure of statistical units in tables.

5.1 Package sdcMicro

In this section we will show how statistical disclosure control methods can be applied to data using the R packages **sdcMicro** (Templ, 2010, 2008, 2009a; Templ and Meindl, 2010) and **sdcTable** (Meindl, 2010; Templ and Meindl, 2010). Another popular software which has implemented methods for masking **microdata** is μ-Argus (Hundepool et al, 2010). μ-Argus has been developed over the last 15 years in several large European research projects. This software provides a graphical user interface (GUI) which can help users who have little experience with statistical software produce safe microdata sets. However, the user has to fully determine specific metadata information which makes the application of μ-Argus cumbersome. In addition to that, the software is error prone and the GUI is based on old technology using Visual Basic without including facilities for reproducibility. Thus, it is not possible to write code via command line interface and the actions performed in the GUI are not saved.

However, data masking can be easily performed with the recently developed flexible software package **sdcMicro** by minimizing information loss and the risk of re-identification

in a very flexible manner. This software can be downloaded from the comprehensive R archive network (CRAN) at http://cran.r-project.org. The package **sdcMicro** includes the most popular techniques for microdata protection. It is designed to protect survey data including sampling weights, but it can also be applied to data without survey weights (e.g. census data). The package **sdcMicro** comes with a manual including help files for all functions. Possible interactions of the functions are described in a package vignette (see Listing 8).

```
require(sdcMicro)
help(package=sdcMicro)
vignette('sdcMicroPaper')
```

Listing 8: Loading `sdcMicro`, displaying the help index and the package vignette.

The package is designed to be flexible and to provide reproducibility, which is also true for the graphical user interface (GUI) of **sdcMicro** (for a detailed description of the GUI, see Templ, 2009a). The package **gWidgetsRGtk2** (Verzani and Lawrence, 2009) is used to generate the GUI and must be installed when using it. This package allows the **gWidgets** API to use the package **RGtk2**(Lawrence and Lang, 2008), based on the powerful Gtk2 libraries within R. The Gtk2 libraries usually come with a standard Linux installation. When installing the R package **gWidgetsRGtk2**, a very easy-to-use built-in install routine pops up (in case that `Gtk2` is not yet installed). Following installation, R has to be restarted once.

The graphical user interface can be loaded by typing the following R command:

```
sdcGUI()
```

Listing 9: Starting the GUI.

Using the GUI, data can easily be loaded, variables selected and different methods easily applied to these selected variables. The GUI provides for interaction between all possible objects and updates all views after a user operation was carried out. Each effect of a mouse click and each operation performed is saved in a script (see Figure 17). This script can later be easily modified and re-applied to the data and/or run only up to a specific line. The examples shown in this section could be easily carried out with the GUI. However, it seems more appropriate to only show the corresponding lines of code in the following examples instead of presenting a large number of screenshots.

A popular software for *tabular data protection* is τ-Argus (Hundepool et al, 2008) which was developed in parallel with μ-Argus and includes most of the well-known methods. However, it is a proprietary solution and the open-source R package **sdcTable** can therefore be recommended. The R package **sdcTable** is a newly developed package to protect tabular data. The package was developed as an open-source project and can be downloaded from the comprehensive R archive network (CRAN, see `http://cran.r-project.org`). The package itself depends on the R package **lpSolve** (Berkelaar and others, 2010) which provides an interface for calling the open source linear

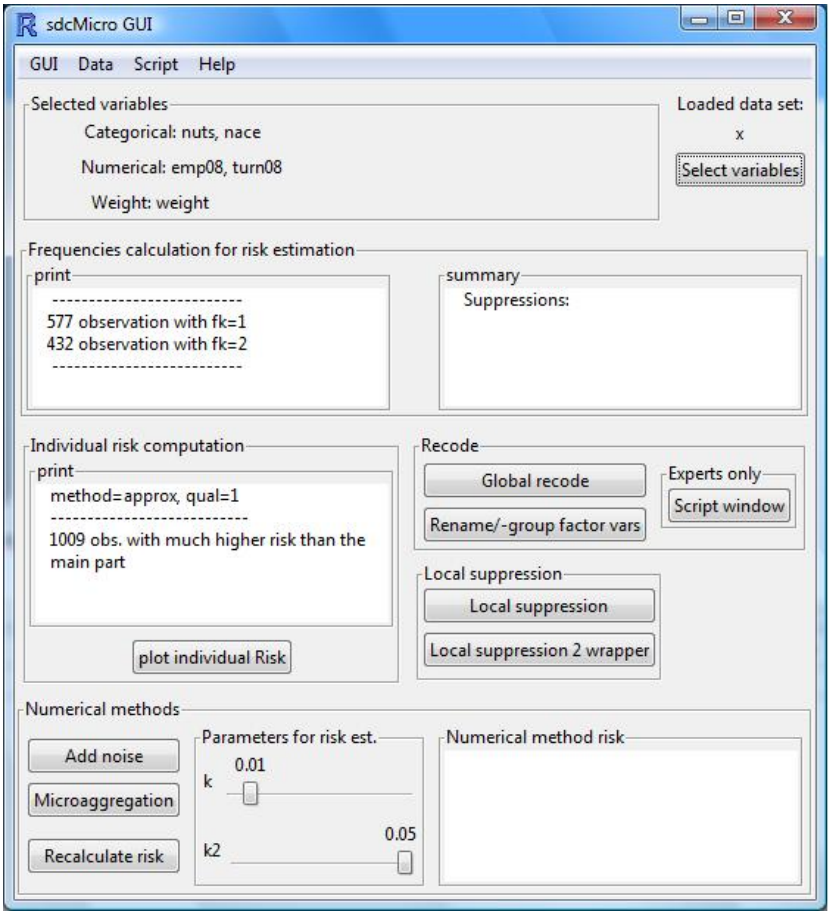


Figure 16: The graphical user interface of **sdcMicro**. The window shows the relevant summaries for anonymization of the CIS data set (see below for the description of this example).

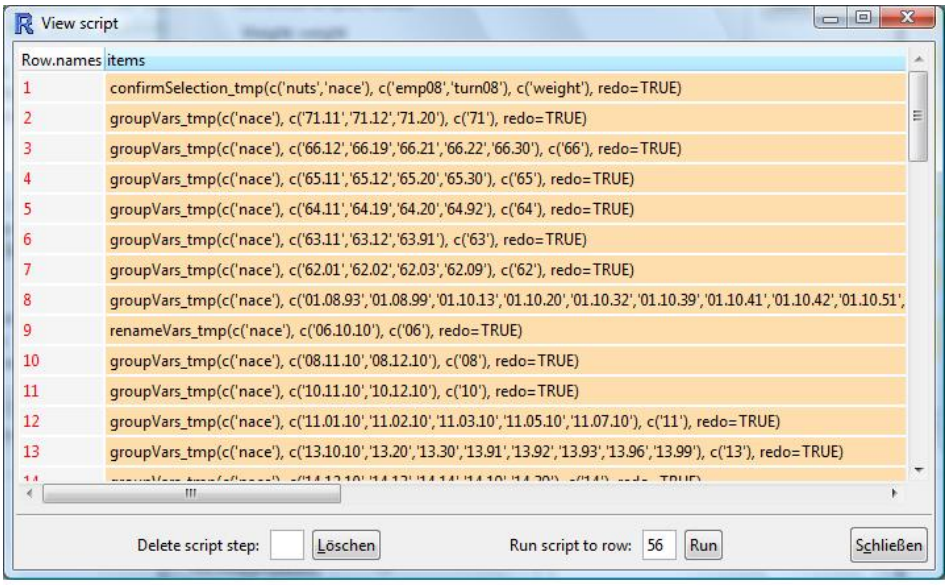


Figure 17: The script generated automatically from the users' interactions with the GUI.

programming solver *lpSolve* within the R environment. The **sdcTable** provides online help for all included functions as well as test data sets and examples which can be copied by the user and directly pasted and run in the R scripting window. To view the available help, the user needs to type the following command:

```
help(package=sdcTable)
```

Listing 10: Displays the help index of package **sdcTable**.

5.2 Statistical disclosure control methods for microdata

Today a number of different concepts exist to make confidential data accessible to researchers and users. A brief outline of these concepts is given below, followed by a detailed discussion on available software for statistical disclosure control in R.

5.2.1 Remote access

The most flexible way to provide microdata is by installing remote access facilities where researchers can have a look at the data using a secure connection. They can analyse the data and choose a suitable model, but it is not possible for them to download it. Unfortunately, in some countries remote access is only be partially available, depending on the discipline from which the data originates. This is attributable to different legislations on data for data deriving from different disciplines. In Austrian law, for example, only microdata masking or remote execution is permissible in official statistics because of the legal situation which prohibits viewing "original" data.

5.2.2 Remote execution and model servers

Whenever remote access facilities are not available, remote execution may be considered. The user can send queries to the remote execution system. After the code is applied to the corresponding data, the output is checked by the data holders' SDC experts to detect and to avoid confidentiality disclosure. Since this is a very resource-intensive task, data holders often send data users structural synthetic data sets so they test their methodology. Such data sets have the same variables and the same scale of variables as the original data. However, remote execution has two drawbacks. First, output checking is a highly sophisticated process and data holders can never be sure whether certain results will lead to disclosure or not. Furthermore, data holders might never be able to understand all methods (for example, available in more than 2500 R packages) that can be applied to the data. Therefore, output checking may not be feasible at all. The second drawback is that the user could fit models to the data based on the synthetic values, but the original data might have quite a different multivariate structure. Since it is not possible to apply exploratory methods on the original data, the existence of data abnormalities could remain undetected. In particular, outliers which could influence the models may be very difficult to detect, since outliers are perturbed or suppressed in any diagnostic of the model due to confidentiality reasons. The same problems are present

in the so-called confidentiality preserving model servers (Steel and Reznek, 2005) where users can apply pre-defined methods on a data server by clicking on a user interface. Moreover, these model servers are limited to only few methods which can be implemented within a reasonable time because every method must be adapted to avoid the disclosure of results.

5.2.3 Secure computer lab

In various statistical offices, researchers have access to data in a secure environment which is placed in a secure room at the corresponding institution. Hence, the researcher can analyse data and the final results are usually checked for confidentiality by the staff of the data holder.

5.2.4 Data perturbation

Since remote access is not available in many cases due to legal restrictions and remote execution is too cost-intensive and time consuming, on the other hand, data source perturbation is one methodology to make data confidential at a minimum loss of information.

5.3 Microdata protection using sdcMicro

Microdata protection/perturbation has proved to be extremely popular and has spread extensively in the last few years because of the significant rise in the demand for *scientific use files* among researchers and institutions.

5.3.1 Example data: The community innovation survey

To demonstrate the available methods for microdata protection we will use a data set from a business survey - the *Community Innovation Survey (CIS)*. The survey collects information on the innovative tendency at enterprise level. The 2008 data include 3534 enterprises responding on 121 variables. The most important variables are *economic activity* (NACE) in accordance with NACE Revision 1.1, *geographic location* (NUTS), *number of employees* (emp08) and *expenditures on innovation and research* (rtot). The survey is based on a stratified simple random sample design with the following stratification variables: *economic activities*, *enterprise size* and *geographic location*. A full description of the survey can be found in (Eurostat, 2006).

This data set is anonymities step by step using different source perturbation methods. The tables generated from such source perturbed data can be considered to be non-disclosive, depending on the method used.

5.3.2 Anonymization of categorical variables

A data intruder may have information of some key variables. *Direct identifying variables*, such as IDs, names, addresses, social security numbers, are usually not disseminated.

Combinations of *indirect identifers*, such as NUTS classification and NACE, may be used to link data sources and to identify statistical units. If an observation (statistical unit) is re-identified, the data intruder will be able to identify all entries of this statistical unit in the data set whereas some entries may contain sensitive information. The subject matter specialist responsible for the specific data used can make subjective yet realistic assumptions about what an intruder might be able to ascertain about respondents and what information is available in the released data set which might be used by a data intruder to disclose information on statistical units (see Templ and Alfons, 2010). In the case of CIS survey data which we consider in our example, Ichim (2008) proposes selecting the variables *economic activity* and *geographical location* as categorical key variables and *number of employees* and *turnover* as continuous scaled key variables.

Frequency counts and disclosure rRisk: Consider a random sample of size n drawn from a population of size N. Let $\pi_i, i = 1,\ldots,N$ be the (first order) inclusion probabilities, i.e. the probability that the element u_i of a population of the size N is chosen in a sample of the size n. All possible combinations of categories in the key variables $X_1,\ldots,X_m$ can be calculated by cross tabulation of these categorical variables. Let $f_k, k = 1,\ldots,n$ be the frequency counts obtained by cross tabulation and let F_k be the frequency counts of the population which belong to the same category. If $f_k = 1$ applies the corresponding observation is unique in the sample. If $F_k = 1$ applies then the observation is **unique** in the population. However, F_k is usually unknown since in statistics information on samples is collected and therefore little information on the population is known (e.g. from registers and/or external sources).

Estimation of F_k: F_k may be estimated by using the sample weights. Whenever an observation has a sampling weight equal to 100 it can be assumed that 100 observations have the same characteristics in the population related to the stratification variables of a (complex) sampling design. The estimates of the frequency counts in the population are given by the sum of the weights associated with the observations which belong to the corresponding category. F_k may then be estimated by

$$\hat{F}_k = \sum_{i \in \{j | x_{j\cdot} = x_{k\cdot}\}} w_i \tag{11}$$

where $x_{i\cdot}$ denotes the i-th row of the matrix with the key variables. The frequency calculation can be done using the R function `freqCalc()`, see Listing 11. The print method reports that 577 enterprises are unique in the sample.

```
x <- read.csv2('cis.csv')
x <- x[,c("nuts","nace","emp08","turn08","weight")]
ff <- freqCalc(x, keyVars=1:2, w=5)
ff

 ---------------------------
577 observation with fk=1
432 observation with fk=2
 ---------------------------

summary(ff$Fk)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   8.313  22.670  51.450  59.400 351.200
```

Listing 11: Calculating the frequency counts and displaying the results.

In function `freqCalc()`, `x` is an object of class `data.frame` or `matrix`, `keyVars` is a vector specifying the column index of the key variables and `w` defines the column index of the weight variable. The resulting output of the function are the frequency counts of the sample and the estimated frequency counts of the population. The function is implemented in C and is integrated in R using the R/C interface. Therefore, the computation is very fast and the function can handle large data sets with many key variables (for runtime issues, see Templ, 2008). In Figure 16 the frequency counts for the CIS data are shown.

Global risk measures: A global measure of the re-identification risk is given by the number of uniqueness which occurs in both the sample and the population. It can be expressed in the following way:

$$\tau_1 = \sum_{k=1}^{n} \mathbb{I}(F_k = 1, f_k = 1) \tag{12}$$

where $\mathbb{I}$ denotes the indicator function.

Another well-known global risk measure is given by:

$$\tau_2 = \sum_{k=1}^{n} \mathbb{I}(f_k = 1)\frac{1}{F_k} \tag{13}$$

in which the indicator is weighed with the reciprocal value of the population frequency counts. The higher the population frequency count, the lower the risk of re-identification. If F_k is particularly high, the data intruder cannot be certain wheather he is able correctly to assign the observation for which he holds information. F_k must be estimated when estimating τ_1 and/or τ_2. τ_1 and τ_2 are estimated by

$$\hat{\tau}_1 = \sum_{k=1}^{n} \mathbb{I}(f_k = 1)E(P(F_k = 1|f_k = 1)) \quad , \quad \hat{\tau}_2 = \sum_{k=1}^{n} \mathbb{I}(f_k = 1)E(\frac{1}{F_k}|f_k = 1) \tag{14}$$

Unfortunately, it is not reasonable to use Equation 11 to estimate $\hat{F}_k$ to estimate the

global risk. A certain distribution of F_k must be assumed in order to formulate a realistic measure of global risk (see, e.g., Templ, 2009b). The most popular method for estimating global risk is the Benedetti-Franconi Model (Benedetti and Franconi, 1998; Franconi and Polettini, 2004) which is implemented in the R package **sdcMicro** in function `indivRisk()`.

Superpopulation models: To estimate the frequencies of population F_k, it is assumed that the population is drawn from a superpopulation. This in fact means that the frequencies in the population will either be generated synthetically by drawing from a specific distribution of the frequency counts, or quantiles of the assumed distribution of F_k are used. Using quantiles of the *a priori* assumed distribution of F_k makes it possible to estimate the risk of each statistical unit. However, this estimation depends on the way the frequency counts of the population are modeled and on how well the model assumption are fulfiled. Many suggestions exist in the literature: the use of a Poisson-Gamma superpopulation model (Bethlehem et al, 1990), the Dirichlet-multinomial model (Takemura, 1999), the negative binomial model (Benedetti and Franconi, 1998; Franconi and Polettini, 2004), a log-linear model (Skinner and Shlomo, 1998, 2006), a multinominal model (Forster and Webb, 2007), the Poisson-inverse Gaussian model (Carlson, 2002).

Application to the example data set CIS: We use the function `indivRisk()` from the R package **sdcMicro** to estimate the individual risk $\hat{r}_k$ (see Listing 13).

```
rk <- indivRisk(ff)
rk
method=approx, qual=1
 ---------------------------
820 obs. with much higher risk than the main part
plot(rk)
```

Listing 12: Estimation and display of individual risks.

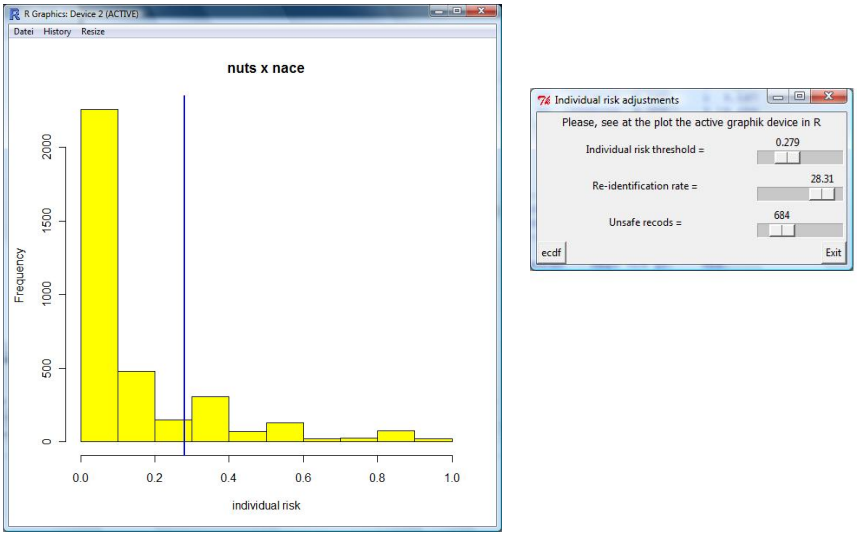
Figure 18(a) shows the enterprises' individual risk. Approximately 700 enterprises have a rather high risk for re-identification (see Figure 18(b)).

Alternatively, the global risk can be easily determined by using a log-linear model assuming that the population frequencies are Poisson distributed. The probability table of the categorical key variables is considered, taking the sampling weights into account. The cell probabilities can be described by a *logit* link function using the following regression model:

$$\text{logit} p = \log o = \log \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k \tag{15}$$

where k is the amount of cells in the table and o is the short expression for *odds*. Using the exponents, Equation 15 can be transformed to

$$e^{logit\, p} = o = \frac{p}{1-p} = e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\ldots+\beta_k x_k} = e^{\beta_0} e^{\beta_1 x_1} e^{\beta_2 x_2} \ldots e^{\beta_k x_k}$$



(a) Individual risk. (b) Corresponding slider.

Figure 18: Individual risk of enterprises in the CIS data before anonymization.

The inverse of the *logit* function is called logistic function. If $logit\ (\pi) = z$, then $\pi = \frac{e^z}{1+e^z}$. Now let us determine the cell probabilities and the global risk for our CIS data set in Listing 13. The global risk measure reports that 83 enterprises are unique in the sample and unique in the population.

```
Fk <- aggregate(x[,'weight'], list(x$nuts,x$nace), sum) # Fk
x1 <- Fk[,1]
x2 <- Fk[,2]
y <- Fk[,3]/sum(Fk[,3]) # cell probabilities
form = y~x1:x2  # model
mod = glm(form,family="poisson") # estimation
pred <- (predict(mod))  # prediction
pred <- exp(pred)/(1+exp(pred))  # transf. with logistic fkt.
res <- pred*sum(Fk[,3]) # estimated frequencies in population
sum(res==1 & aggregate(ff$fk, list(x$nuts,x$nace),
                unique)[,3] ==1) # global risk
[1] 83
```

Listing 13: Log-linear model for the CIS survey data.

All concepts for risk estimation require a lot of assumptions about the superpopulation, which may not hold in practice. The best way to model F_k depends on the underlying data, i.e. each data set requires different assumptions. The main drawback of all these methods is that they are not data driven, but rely on *a priori* assumptions about distributions.

***k*-anonymity:** The concept of k-anonymity simplifies the risk estimation because no estimation of cell frequencies at population level is needed in this case. We have k-anonymity for a statistical unit whenever at least k statistical units exist in the sample which have the same combination of values in the categorical key variables. The goal

is fo every statistical unit to at least have k-anonymity with $k \geq 2$. Listing 11 already revealed that 557 enterprises have a frequency 1 in the sample and are thus unique. The aim is to achieve k-anonymity by applying *global recoding* and *local suppression*.

Global recoding and local suppression: Categories including only few entries lead to uniqueness and high risk. Thus, recoding into broader categories (e.g. recoding from NACE 4-digit level to NACE 2-digit level) or combining categories (e.g. combining two NACE 2-digit levels) may reduce the risk. The original CIS data do not have k-anonymity and the disclosure risk might therefore be high. The risk is reduced through the global recoding of a variable. Here the categories of NACE are assigned to broader categories. As mentioned before, it is very likely that after the recoding, more observations are equal in the key variables and so $\hat{F}_k$ increases, which decreases the individual risk $\hat{r}_k$. Recoding NACE from the 4-digits level to the 1-digit level reduces the risk dramatically. This recording can also be done from the GUI of **sdcMicro**. The result of the recoding which is saved in the object `x2` is shown in Listing 14.

```
ff2 <- freqCalc(x2, keyVars=1:2, w=5)
ff2


 ---------------------------
0 observation with fk=1
2 observation with fk=2
 ---------------------------
```

Listing 14: Global recoding of one key variable.

However, some statistical units do not reach 2-anonymity. Suppose that we do not want to apply further recoding. Another option is to use local suppression. With this method, certain values in the key variables are suppressed to reduce the risk. This should be done in an optimal way, i.e. to suppress as few values as possible, on the one hand, and to guarantee low risk of re-identification, on the other hand. An iterative algorithm for finding an optimal solution is implemented in the package **sdcMicro** (Templ, 2010). The user can also assign weights to the key variables, because some variables often seem to be less important (such as *NUTS* in our case) than others (such as NACE, the economic activity classification). The probability of local suppressions in variables with high weights is then higher than for variables with minor importance/weights, see Listing 15. Finally, the function `localSupp2Wrapper()` suppressed one value in NACE and one value in NUTS to reach 2-anonymity.

```
localSupp2Wrapper(x, keyVars=1:2, w=5, kAnon=2, importance=c(0.5,1))

[1]  2-anonymity has been reached.
```

Listing 15: Obtaining k-anonymity (here: $k = 2$).

The data now has a low risk of re-identification for each observation, see Figure 19.

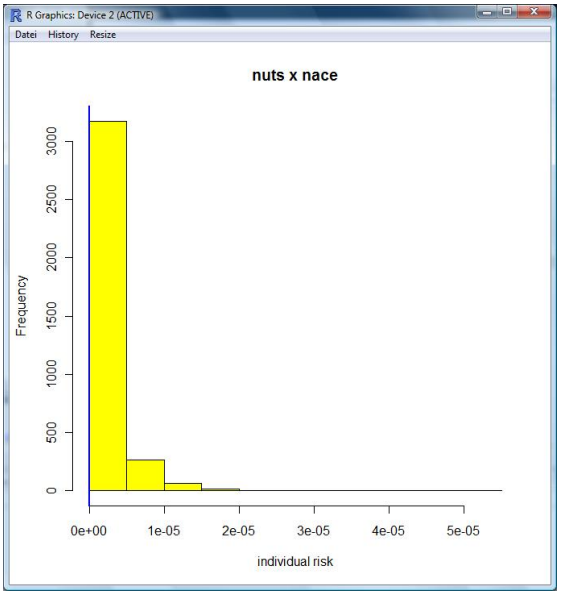


Figure 19: Individual risk of enterprises after recoding and local suppression.

Post randomization Method (PRAM): The post randomization Method (PRAM) is a perturbation disclosure control technique applied to categorical variables in microdata (Kooiman et al, 2002). In PRAM the values of categorical variables may change to different categories according to a pre-defined probability mechanism. The transition probabilities for any category to any other category are usually saved in a transition matrix. The key aspect of the PRAM method is that it conserves the original frequency distributions while minimizing the loss of information. It is easy to see that this concept is quite different from the perturbation concepts discussed earlier. It is possible that a statistical unit is not changed even if the disclosure risk is high for that unit. However, the data intruder will never know whether some values are altered and whether the match is the correct one. It is important for the transition probabilities to have been chosen appropriately before PRAM can be applied. The specific choice of the matrix will influence the disclosure risk which leads to an iterative process of choosing the Markov matrix: the effects on the disclosure risk can be computed with a certain matrix, and if these effects are not satisfactory, the choice of the matrix should be performed again, and so on. Also, the effect of PRAM on statistical analyses performed on the perturbed microdata file should be considered (for further analysis see Kooiman et al, 2002).

The function `pram()` provides the invariant PRAM methodology and produces objects from class `pram`. The arguments of the function are shown in Listing 16. The argument `pd` is the minimum diagonal entries for the generated transition matrix and `alpha` is the amount of perturbation used for the invariant pram method.

```
args(pram)
R> function (x, pd = 0.8, alpha = 0.5)
```

Listing 16: Parameters for `pram()`.

A print method and a summary method are provided for objects of the `pram` class. In the following, the variable `NUTS` from our example data set from the CIS business survey is perturbed. A lot of information is stored in the resulting object `NUTSpram` (see Listing 17) including the invariant transition matrix. Summary and print methods are provided as well.

```
nuts <- x[,"nuts"]
NUTSpram <- pram(nuts)
summary(NUTSpram)

 ----------------------
 original frequencies:

AT11 AT12 AT13 AT21 AT22 AT31 AT32 AT33 AT34
 100  646  645  193  488  714  273  284  190

 ----------------------
 frequencies after perturbation:

AT11 AT12 AT13 AT21 AT22 AT31 AT32 AT33 AT34
 109  637  651  189  486  733  265  276  187

 ----------------------
 transitions:
      transition Frequency
1   AT11 --> AT11        83
2   AT11 --> AT12         3
3   AT11 --> AT13         3
4   AT11 --> AT21         2
5   AT11 --> AT22         3
6   AT11 --> AT31         3
7   AT11 --> AT32         1
8   AT11 --> AT33         1
9   AT11 --> AT34         1
10  AT12 --> AT11         7
11  AT12 --> AT12       582
12  AT12 --> AT13         8
13  AT12 --> AT21         7
14  AT12 --> AT22        14
15  AT12 --> AT31        14
16  AT12 --> AT32         7
17  AT12 --> AT33         1
..                       ..
```

Listing 17: An application of PRAM and selected output of the summary method.

5.3.3 Anonymization of continuous scaled variables

The concept of uniqueness used in the previous section no longer works when continuous scaled variables are observed, because nearly every combination of continuous scaled variables is unique in the sample. Even the *pram* methodology cannot be applied since

the transition matrix becomes almost $n \times n$ with n being the number of observations in the data set. A data intruder may have information about a value of a statistical unit. If this value matches with the anonymized data then he can be quite certain that the re-identification was successful. The intruder then knows all the information about this unit in the data set, i.e. the values of each variable of this unit. Note that this information can be very sensitive, for example, information about competitors in an economic branch. Moreover, a data intruder may also use *record linkage* techniques to identify already perturbed values. A successful re-identification of a statistical unit like an enterprise in business data is possible if a value is not sufficiently perturbed.

Adding noise methods. Adding noise to a data matrix X is simply done by

$$Y = X + \varepsilon$$

where $X \sim (\mu, \Sigma), \varepsilon \sim N(0, \Sigma_\varepsilon)$, $\Sigma_\varepsilon = \alpha \cdot diag(\sigma_1^2, \sigma_2^2, \ldots, \sigma_p^2), \alpha > 0$, $Cov(\varepsilon_i \neq \varepsilon_j)$ for all $i \neq j$ and p is equal to the dimension of the matrix containing the continuous scaled variables which are to be perturbed (see e.g. also in Brand, 2004). However, multivariate characteristics will not be preserved when adding (uncorrelated) noise independently to each variable. Correlation coefficients could be preserved if correlated noise is added. In this case, the covariance matrix of the masked data is $\Sigma_Y = (1+\alpha)\Sigma_X$ (for example, see Brand, 2004). In Listing 18 correlated noise is added to the two continuous scaled key variables of the CIS survey.

```
x_corNoise <- addNoise(x[,3:4], method='correlated2', noise=50)
```

Listing 18: Adding correlated noise.

Note that many other methods can be applied using the function `addNoise()`. The selected method is determined by the function argument `method` (see Templ, 2010, 2008).

Templ and Meindl (2008a) showed that all these methods are either influenced by outliers or they do not preserve outliers sufficiently. It is therefore reasonable to apply robust methods for any method to apply noise (see Templ and Meindl, 2008a). In addition, Templ and Meindl (2008b) showed that outliers have to be much more protected than the rest of the observations because outlying observations have a higher risk for re-identification than non-outliers. This is especially true for business data sets since the inclusion probabilities of large companies are 1 or close to 1, and in addition, enterprises with large turnover are not part of the main body of the data. Also, minor modifications of the data do not prevent re-identification with statistical matching methods.

Rank swapping. The term *rank swapping* was introduced by Dalenius and Reiss (1982) but has evolved and assumed new meanings relating to new statistical methodologies over the past 25 years. In this method the entries of one variable are sorted by their numerical values (ranking). Each ranked value is then swapped with another ranked value that has been chosen randomly within a restricted range. The rank of two

swapped values cannot differ by more than p percent of the total number of observations. Rank swapping must be applied to each variable separately and the multivariate data structure is therefore not preserved very well (see also Templ and Meindl, 2010). In **sdcMicro** rank swapping is applied column-wise as is shown in Listing 19.

```
x_swapp <- swappNum(x[,3:4, p=15)
```

Listing 19: Rank swapping using a 15 percent swapping range.

Microaggregation. Microaggregation is defined as follows (see the glossary of statistical disclosure control at `http://neon.vb.cbs.nl/casc/Glossary.htm`):

> Records are grouped based on a proximity measure of variables of interest, and the same small groups of records are used in calculating aggregates for those variables. The aggregates are released instead of the individual record values.

The most challenging part with regard to this technique is the choice of the "proximity" measure. Note that the multivariate structure of the data is only preserved if similar observations are aggregated.

Sorting data based on one single variable in ascending or descending order (method `single` in **sdcMicro**), sorting the observations in each cluster (after clustering the data) by the most influential variable in each cluster (method `influence`, see Templ (2006)) as well as sorting (and re-ordering the data after aggregation) in each variable (for the individual ranking method, see Defays and Nanopoulos, 1993) and projection methods where typically the data are sorted based on the first principal component are not considered optimal for multivariate data (see Templ, 2007). Projection methods typically sort the data according to the first principal component (method `pca`) or its robust counterpart (method `pppca`, whereas the methods used to obtain the principal components can differ. Usually, all principal components must be estimated when using standard approaches for PCA, but the method `pppca` prevents this and estimates the first several (robust) principal components by projection pursuit without the need to estimate the covariance matrix.

The Maximum Distance to Average Vector (MDAV) method often provides better results. This method is an evolution of the multivariate fixed-size microaggregation (see Domingo-Ferrer and Mateo-Sanz, 2002, for example). However, this method (`mdav` in **sdcMicro**) is based on Euclidean distances in a multivariate space and replaces similar observations based on these distances with their aggregates. The algorithm has been improved by replacing Euclidean distances with robust Mahalanobis distances. In Templ (2006), a new algorithm called RMDM (**R**obust **M**ahalanobis **D**istance based **M**icroaggregation) was proposed for microaggregation where MDAV (Domingo-Ferrer and Mateo-Sanz, 2002) is adapted in several ways.

Microaggregation can easily be applied with the package **sdcMicro**:

```
microaggregation(x, method='METHOD', k=3)
```

Listing 20: General procedure to apply microaggregation.

More than ten options are available for the argument `method` of this function Templ (2008). The argument `k` determines the aggregation level used, i.e. how many statistical units are joined together.

5.3.4 Information loss

After perturbing the continuous scaled variables, the data utility and the disclosure risk of the perturbed data have to be evaluated. For data utility, the original values are compared to the perturbed ones, for example, by aggregated distances from original data points to corresponding values from the perturbed data divided by the standard deviation for each variable (Mateo-Sanz et al, 2005, method IL1). However, this measure is large even if only one outlier is highly perturbed and the rest values are exactly the same as in the original data set. The data utility can be checked with the function `dUtility()` (see Listing 21) whereby different methods can be chosen. Note that *x* is the original data set and *xm* the masked data set in Listing 21.

```
dUtility(x, xm, method="IL1")
```

Listing 21: Calculation of the IL1 data utility measure.

In addition, a `summary()` method for objects created by `addNoise()` and `microaggregation()` is available, which can calculate many (more than ten) different measures of data utility.

5.3.5 Disclosure risk

It is assumed that an intruder can link the masked record to its original value (see, e.g., Mateo-Sanz et al, 2005). Given the value of a masked variable we must check whether the corresponding original value falls within an interval centred on the masked value. The width of the intervals are chosen based on the outlyingness of the observation (Templ and Meindl, 2008a).

```
dRisk(x, xm)
```

Listing 22: Estimation of disclosure risk for perturbed continuous scaled variables.

5.4 Generation of synthetic confidential data

Instead of perturbing microdata, the underlying sample can be simulated using the information of the sample. This is typically done using regression methods. Generating completely realistic synthetic data seems an impossible task. Nevertheless, being as close to reality as possible is sufficient for most user needs.

One approach for generating synthetic data sets is discussed by Rubin (1993). He addresses the confidentiality problem connected with the release of publicly available microdata and proposes the generation of fully synthetic microdata sets using multiple imputation. Raghunathan et al (2003), Drechsler et al (2008) and Reiter (2009) discuss this approach in further detail. The R package **simPopulation** provides modeling facilities to generate synthetic data. When generating synthetic data, the following conditions need to be considered (Münnich et al, 2003; Münnich and Schürle, 2003; Kraft, 2009):

- The actual size of regions and strata.
- Appropriate representation of the marginal distributions and interactions between variables.
- Allowing heterogeneities between subgroups, especially regional aspects.
- Avoidance of pure replication of units from the underlying sample, as this generally leads to extremely small variability of units within smaller subgroups.
- Ensuring data confidentiality.

In addition, data from official statistics often contains information about components. The data simulation method must ensure that a breakdown of a variable into components is achieved in a realistic manner. In general, the procedure consists of four steps:

1. Set up of the structure
2. Simulation of categorical variables
3. Simulation of continuous variables
4. Splitting continuous variables into components.

The procedure has to consider the stratification of the data to account for heterogeneities such as regional differences. Furthermore, sample weights have to be considered in each step to ensure a high similarity of expected and realized values. Concerning data confidentiality, a detailed analysis of the framework using different worst case scenarios was carried out in (Templ and Alfons, 2010). The conclusion of this analysis is that the synthetic population data are confidential and can therefore be distributed to the public. The key variables of the CIS data set can be easily simulated using the package **simPopulation** (Alfons and Kraft, 2010).

5.5 Tabular data protection

Statistical agencies generally do not publish microdata but disseminate information in the form of statistical tables consisting of aggregated data. If too few statistical units contribute to a cell in a table, a data intruder would be able to re-identify these statistical units. For example, if the enterprise *voestalpine* is the only one which contributes to

Table 2: Two-dimensional frequency table with margins.

	size 1	size 2	size 3	Total
5151	0	6	7	13
5152	5	13	7	25
5153	19	24	8	51
5154	9	17	11	37
5155	4	7	0	11
5156	4	2	5	11
5157	7	6	1	14
515	48	75	39	162

a certain cell of a, say, two-dimensional table by NACE and NUTS, the cell value then reflects the turnover of this enterprise. Note that the cell value might contain information about more confidential information which a competitor of *voestalpine* should not have access to.

When generating tables from source perturbed data (discussed in the previous section) confidential tables might also be obtained.

5.5.1 Frequency and magnitude tables

A statistical table is constructed from microdata. According to the characteristics of one or more dimensional variables, all statistical units (e.g. persons, enterprises or legal entities) that possess the same set of characteristics for each of the dimensional variables are grouped. If the number or units for each combination of the dimensional variables is listed, the resulting table is referred to as a frequency table. If some aggregated measure (usually the sum of a continuous scaled variable) is listed, the table is referred t as magnitude table. However, it should be noted that each statistical table is defined by a set of linear constraints connecting inner and marginal cells. In order to illustrate this relation, the following simple, two-dimensional frequency table (2) is considered (see also in Castro and Baena, 2008). In Table 2 the number of enterprises featuring a given characteristic correspond to certain NACE codes and employee size ranges. It is easy to see that in this simple example the linear constraints are just the sum of the rows and columns. This means that the number of enterprises in NACE 5151 and the number of enterprises producing products in the other NACE 4-digits level sum up to the total number of enterprises in the NACE classification code 515. Additionally, summing up over the employee size classes for any NACE code results in the linear restrictions which form the totals in each NACE. Table 3 in which the total turnover in each category is listed for each combination of NACE code and employee size class (the original values are not reported for the CIS data because of confidentiality reasons). From the frequency Table 2 it is clear that only one statistical unit contributes to the cell defined by NACE code 5157 and employee size range 3. Thus, a data intruder learns that the income of this individual is 194535 Euro. Disclosure would definitely occur if the

Table 3: Magnitude two-dimensional table (in thousand).

	size 1	size 2	size 3	
5151	0	332693	2360685	3593378
5152	25946	381443	2049021	1556410
5153	85602	834582	1245262	2165446
5154	53225	509625	2218779	2781629
5155	26314	94256	0	120570
5156	14127	46672	2804788	2955587
5157	44869	118456	194535	357860
515	250083	2407727	10873070	13530880

data intruder manages to identify this enterprise. Since laws on data privacy are strict in almost all countries, national statistical offices do not only protect microdata but also aggregated data such as statistical tables in order to avoid the disclosure of sensitive information. Popular methods for protecting aggregated output to avoid the possible re-identifications of statistical units or attribute disclosure are cell-suppression, rounding or table-reorganization by collapsing categories. Generally speaking, this means that some cells have to be suppressed or modified (see Templ and Meindl, 2010).

5.5.2 Primary sensitive cells

The first step when protecting tabular output is to determine table cells that need to be protected. These cells are referred to as primary sensitive cells. One popular rule for identifying primary unsafe table cells is the *minimum frequency rule*. Using this rule, any cell of a table for which n or less statistical units contribute is considered to be primary unsafe. The parameter n is often set to 3 or 4. Other more sophisticated rules are based on the concept that a cell should also be primary suppressed if one statistical unit dominates the cell (for details, see Cox, 1981; Willenborg and De Waal, 2000; Merola, 2003).

5.5.3 Secondary cell suppression

It is easy to see that the possible primary cell in Table 2 with a frequency 1 can be easily re-calculated by subtracting the row values from the row total. Due to such linear relationships which are typical for statistical tables it does not suffice to protect tables by identifying primary sensitive cells and suppressing their values. In order to avoid the re-calculation or the possibility to gain good estimates for sensitive cells, additional cells need to be suppressed. The problem related to the finding additional cells for suppression is termed *secondary cell suppression problem*. To find an optimal solution to suppress as few cells as possible in a hierarchical table, where margins can again be cell values of another table, is known to be a NP-hard problem. Optimal solutions exist which are based on minimizing a pre-defined cost function taking the linear relationships of the hierarchical tables into account (see, e.g., Fischetti and Salazar-Gonzalez, 1999). The objective function is often chosen to minimize the total number of additional suppressions

or similar criteria. The computational costs to solve the resulting complex combinatorial optimization problem are tremendous for large hierarchical tables. Therefore, different heuristic solutions for 2 and 3-dimensional tables have been proposed (Kelly et al, 1990; Cox, 1995; Castro, 2002; Fischetti and Salazar-González, 2000; Meindl, 2010).

5.6 Package sdcTable

The most challenging part from the user's point of view is the preliminary data preparation which is necessary before tabular protection can be applied. In this case, meta information about the hierarchical variables defining the table must be provided by the user. First the user has to determine the hierarchical structure of the table excluding any subtotals (Meindl, 2010). Subsequently, several different cell suppression methods can be easily applied using the function `protectTable()`. The manual of `sdcTable` includes a practical example of how to use the package to protect tables.

5.7 Summary

Statistical disclosure control has become increasingly important and statistical data collectors should devout a lot of effort to this. Otherwise, the trust of respondents will decrease and response rate will drop. In addition, laws on data privacy must be respected. The SDC methods can be categorized as two different concepts: (i) the anonymization of microdata and (ii) the anonymization of cells in tables. The packages **sdcMicro** and **sdcTable** provide practitioners with tools to fulfil the requirements of statistical disclosure control in these two categories. A brief description of the most popular methods for the protection of categorical and continues key variables was presented and illustrated with the corresponding software implementation in the R package **sdcMicro**. The following list presents the most important functions:

- `freqCalc()` for frequency calculation
- `indivRisk()` for individual risk estimation
- `globalRecode()` for recoding
- `localSupp()` and `localSupp2Wrapper()` for local suppression
- `pram()` for post randomization
- `microaggregation()`, `rankSwapp()`, `addNoise()` for perturbing continuous scaled key variables
- Various print, summary and plot methods.

All these functions can either be applied by using the command line interface or the **sdcMicro** graphical user interface, simply by clicking the corresponding buttons.

6 Working with large data sets

Generally, R keeps the data in memory and all calculations are performed in the memory. This means that it is not usual (but possible) to swap data from the hard disk during calculations. An integer value requires 4 bytes of memory, a value in double precision has 8 bytes. This means that R needs approximately 38 MB of memory when working with a data set of size 100000 rows and 50 columns (see Listing 23).

```
y <- matrix(rnorm(100000*50), ncol=50)  ## generate artificial data
dim(y)                                  ## dimension of the data set.
[1] 100000     50
x <- object.size(y)
print(x, units="Mb")
38.1 Mb                                 ## size of the data set.
```

Listing 23: Object size of a data set.

Either loading many such data sets in the R workspace or dealing with much larger data sets in combination with insufficient memory size could be problematic. However, solutions to deal with this problem exist which include:

- Storing the data in a database (see Todorov, 2010) and connecting to this database with one of the available packages
- Using specialized packages as, for example, the package **filehash** which stores the data on the hard disk and allows R to access these data seamlessly.
- Buying more RAM for the computer. Be aware that at 32 bit Windows one can have up to 3GB of memory since 32 Bit systems impose a theoretical maximum of 4GB and this number is determined as the maximum number of addressable values a 32 bit number can have, and not all of that RAM will be made available to the OS.

In addition, computer-intensive calculations may also cause a running out of memory. For example, calculating a distance matrix from the data used in Listing 23 is not possible by applying standard methods on a regular personal computer, since the object which has to be generated would have a size of 100000×100000 and this requires approximately 76200 Mb of memory.

Fortunately, the regular applications in official statistics never run out of memory when the computer has at least 2 Gb RAM which is standard nowadays.

The handling of large data sets in R is an actively developing research area and many packages are already available which cover one or other aspects of this issue. It is not possible to discuss here all of them, thus only a brief summary is provided. A more extensive review of methods for dealing with large data sets, particularly for analysing of official statistics data, is under preparation.

The package **filehash** allows to swapping data to the hard disk as shown in the example in Listing 24. It implements a simple key-value style database using character string keys, associated with data values that are stored on the disk. A simple interface is provided for the basic data manipulation functions on the database: insert, retrieve and delete. The **filehash** databases are treated much like environments and lists which are already used in R. By using the utilities provided it is possible to perform interactive and exploratory analyses on large data sets. Three different file formats for representing the database are currently available and new formats can easily be incorporated by third parties (Peng, 2006).

```
library(filehash)
x <- data.frame(matrix(rnorm(100000*50), ncol=50))
head(colnames(x))
dumpDF(x, dbName="df1.dump")
rm(x)
x1 <- db2env(db="df1.dump")
with(x1, mean(X1))
[1] -0.002216159
with(x1, lm(X1~X2+X3))

Call:
lm(formula = X1 ~ X2 + X3)

Coefficients:
(Intercept)           X2           X3
  -0.002234    -0.005787     0.003399
```

Listing 24: Using the package **filehash** to deal with large data sets.

Function `dumpDF()` (see Listing 24) swaps the data onto the hard disk, so R memory does not keep a copy of it, which saves memory. The data set can be accessed through the environment `x` (see Listing 24). To access it, the function `with()` can be used by naming the variables of the hashed data set.

Bigmemory is designed to analyse large data sets in R while the internal data management is done in C++. It is possible to simultaneously apply different procedures on one data matrix (shared memory).

The **biglm** package uses incremental computations to offer `lm()` and `glm()` functionality to data sets stored outside of R's main memory. The **ff** package offers file-based access to data sets that are too large to be loaded into the memory, along with a number of high-level functions (see http://CRAN.R-project.org/view=HighPerformanceComputing).

7 Summary, conclusions and outlook

There is an increasing demand for statistical tools which combine the ease of use of traditional software packages with the availability of the latest analytical methods, which is only possible on account of the flexibility provided by a statistical programming language such as R. In this study, continuing and extending the lines started with a previous UNIDO working paper (Todorov, 2010), we discussed the versatility of the R programming environment and how it is possible to apply this environment in national and international statistical offices. This was illustrated by examples from the statistical production process of UNIDO where certain steps were either migrated or newly developed in R.

There are many more interesting topics which might be worth considering when thinking about using R for working in the area of official statistics. A brief overview of such topics follows, which will be included in a future paper advocating the introduction of R in national or international statistical offices.

Survey analysis with R Complex survey samples are usually analysed using specialized software packages (SUDAAN, WesVarPC, etc.). From the most well known general purpose statistical packages Stata provides much more comprehensive support for analysing survey data than SAS and SPSS, and could successfully compete with the specialized packages. In R functionality for survey analysis is offered by several add-on packages, the most popular being the **survey** package. Detailed information can be found in the manuals of the package as well as from its home page, maintained by the author, Thomas Lumley, at http://faculty.washington.edu/tlumley/survey/. A brief overview follows:

1. Designs incorporating stratification, clustering and possibly multistage sampling, allowing unequal sampling probabilities or weights; multistage stratified random sampling with or without replacements;

2. Summary statistics: means, totals, ratios, quantiles, contingency tables, regression models for the whole sample and for domains;

3. Variances by Taylor linearization or by replicate weights (BRR, jack-knife, bootstrap, or user-supplied);

4. Post-stratification and raking;

5. Graphics: histograms, hexbin scatterplots, smoothers.

Other relevant R packages are **pps**, **sampling**, **sampfling**, all of which focus on design, in particular, *PPS* sampling without replacement.

Graphical user interface for R R is provided with a command line interface (CLI) which is the preferred user interface for power users because it allows direct control on calculations and it is very flexible. However, solid knowledge of the language is required and the learning curve is typically longer than with a graphical user interface (GUI). The user interface remains the biggest difference between R and S-PLUS or EViews, since those programmes implement a very sophisticated GUI. Many R users (or potential R users) are asking for and would probably benefit from a R GUI. Several projects exist developing or offering the opportunity to develop alternate user interfaces. A Special Interest Group mailing list (R-SIG-GUI) also exists to freely discuss issues of concern. The R (GUI) Wiki is also available to exchange information and ideas related to the use of R GUIs and to start using R. A comprehensive list of R GUI projects can be found at http://sciviews.org/_rgui/ with the most prominent being RCommander which is available as a CRAN package **Rcmdr** written in Tcl/Tk (see Fox, 2005).

Working with large data sets One issue R is often criticized for is the fact that it loads the complete data to be processed in the memory, which might be a serious limitation in some cases. In Section 6 we briefly discussed ways to resolve this problem by storing the data set in a database and using one of the suitable R packages for database access or by using some of the specialized CRAN packages for working with large data sets like the package **filehash**, **ff**, **bigmemory** and **biglm**. A more detailed analysis of this problem from the viewpoint of the analysis of official data with concrete examples from this area is still pending.

Programming with R As already mentioned above, if we want to combine the ease of use of the traditional software packages with the availability of the latest analytical methods, we need the flexibility provided by a statistical programming language. One of the main attractions of using the R environment is the ease with which users can extend the environment by writing their own programmes and functions. The R programming syntax is easy to learn (although many will not agree with this), even for users with no previous programming experience. On the other hand, for users with previous experience with other programming languages like C or Java, R is the statistical package of choice compared to SAS or SPSS. In the examples used throughout this paper we had to of course write code but there are specific programming issues which have to be considered: the object orientation of R, building of R packages, etc.

References

Acock AC (2005) Working with missing values. Journal of Marriage and Family 67(4):1012–1028

Aitchison J (1986) The Statistical Analysis of Compositional Data. Chapman & Hall

Alfons A, Kraft S (2010) **simPopulation**: Simulation of synthetic populations for surveys based on sample data. R package version 0.2

Alfons A, Templ M, Filzmoser P (2009) On the influence of imputation methods on laeken indicators: Simulations and recommendations. In: UNECE Work Session on Statistical Data Editing; Neuchatel, Switzerland, p 10, to appear

Allison DP (2001) Missing Data Thousand Oaks. Sage Publications

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 B (Statistical Methodology) 127(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

Benedetti R, Franconi L (1998) Statistical and technological solutions for controlled data dissemination. In: Pre-Proceedings of New Techniques and Technologies for Statistics, pp 225–232

Berkelaar M, others (2010) lpSolve: Interface to Lp_solve v. 5.5 to solve linear/integer programs. URL http://CRAN.R-project.org/package=lpSolve, r package version 5.6.5

Bethlehem J, Keller W, Pannekoek J (1990) Disclosure control of microdata. Journal of the American Statistical Association 85(409):38–45

Billor N, Hadi AS, Vellemann PF (2000) Bacon: Blocked adaptative computationally-efficient outlier nominators. Computational Statistics & Data Analysis 34(3):279–298

Boudt K, Todorov V, Upadhyaya S (2009) Nowcasting manufacturing value added for cross-country comparison. Statistical Journal of the IAOS: Journal of the International Association of Official Statistics 26:15–20

Box GEP, Cox DR (1964) An analysis of transformations. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 26:211–252

Brand R (2004) Microdata protection through noise addition. In: Privacy in Statistical Databases. Lecture Notes in Computer Science. Springer, pp 347–359

van Buuren S, Oudshoorn C (2005) Flexible multivariate imputation by MICE. Tno/vgz/pg 99.054, Netherlands Organization for Applied Scientific Research (TNO), URL http://web.inter.nl.net/users/S.van.Buuren/mi/docs/rapport99054.pdf

Carlson M (2002) Assessing microdata disclosure risk using the poisson-inverse gaussian distribution. Statistics in Transition 5:901–925

Castro J (2002) Network flows heutistics for complementary cell suppression. In: Lecture Notes in Computer Sciences. Inference Control in Statistical Databases., vol 2316, pp 59–73

Castro J, Baena D (2008) Using a mathematical programming modelling language for optimal CTA. In: Domingo-Ferrer J, Saygin Y (eds) Privacy in Statistical Databases, Lecture Notes in Computer Science, vol 5262, Springer, Heidelberg, pp 1–12

Cerioli A, Riani M, Atkinson AC (2009) Controlling the size of multivariate outlier tests with the MCD estimator of scatter. Statistics and Computing 19(3):341–353

Cochran WG (1977) Sampling Techniques, 3rd Edition. John Wiley

Copt S, Victoria-Feser MP (2004) Fast algorithms for computing high breakdown covariance matrices with missing data. In: Hubert M, Pison G, Struyf A, Van Aelst S (eds) Theory and Applications of Recent Robust Methods, Statistics for Industry and Technology Series, Birkhauser Verlag, Basel

Cox L (1981) Linear sensitivity measures in statistical disclosure control. Journal of Statistical Planning and Inference 75:153–164

Cox L (1995) Network models for complementary cell suppression. Journal of the American Statistical Association 90:1453–1462

Croux C, Haesbroeck G (1999) Influence function and efficiency of the minimum covariance determinant scatter matrix estimator. Journal of Multivariate Analysis 71:161–190

Daas PJ, Ossen SJ, Tennekes M (2010) Determination of administrative data quality: Recent results and new developments. URL http://q2010.stat.fi/papersbig/daas_ossen_tennekes_q2010_paper_session34_piet_daas_paper.pdf, Q2010 European Conference on Quality in Statistics, 4-6 May 2010, Helsinki

Dalenius T, Reiss S (1982) Data-swapping: A technique for disclosure control. In: Proceedings of the Section on Survey Research Methods, American Statistical Association, vol 6, pp 73–85

De Waal T (2008) An overview of statistical data editing. Discussion paper (08018), Statistics Netherlands, URL http://www.cbs.nl/NR/rdonlyres/693E4B18-9322-4AC2-99FD-DB61F03637B2/0/200818x10pub.pdf

Defays D, Nanopoulos P (1993) Panels of enterprises and confidentiality: the small aggregates method. In: Proceedings of the 1992 Symposium on Design and Analysis of Longitudinal Surveys, Statistics Canada, Ottawa, pp 195–204

Dempster AP, Laird MN, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 39:1–22

Domingo-Ferrer J, Mateo-Sanz J (2002) Practical data-oriented microaggregation for statistical disclosure control. IEEE Trans on Knowledge and Data Engineering 14(1):189–201

Donoho DL (1982) Breakdown properties of multivariate location estimators. Tech. rep., Harvard University, Boston, URL http://www-stat.stanford.edu/~donoho/Reports/Oldies/BPMLE.pdf

Drechsler J, Bender S, Rässler S (2008) Comparing fully and partially synthetic datasets for statistical disclosure control in the German IAB Establishment Panel. Trans Data Priv 1(3):105–130

Eurostat (2006) The forth community innovation survey (cis4). methodological recommendations. Tech. rep., Doc. Eurostat/F4/STI/CIS/2b

Filzmoser P, Garrett RG, Reimann C (2005) Multivariate outlier detection in exploration geochemistry. Computers & Geosciences 31:579–587

Filzmoser P, Maronna R, Werner M (2008) Outlier identification in high dimensions. Computational Statistics & Data Analysis 52(3):1694–1711

Fischetti M, Salazar-Gonzalez J (1999) Models and algorithms for optimizing cell suppression in tabular data with linear constraints. Journal of the American Statistical Association 95:916–928

Fischetti M, Salazar-González J (2000) Complementary cell suppression for statistical disclosure control in tabular data with linear constraints. Journal of the American Statistical Association 95:916–928

Forster J, Webb E (2007) Bayesian disclosure risk assessment: predicting small frequencies in contingency tables. Journal of the Royal Statistical Society C 56:551–570

Fox J (2005) The R Commander: A basic-statistics graphical user interface to R. Journal of Statistical Software 14(9):1–42, URL http://www.jstatsoft.org/v14/i09/paper

Franconi L, Polettini S (2004) Individual risk estimation in μ-ARGUS: a review. In: Privacy in Statistical Databases, Lecture Notes in Computer Science, Springer, Heidelberg, pp 262–272

Gelman A, Hill J, Su YS, Yajima M, Pittau M (2010) mi: Missing Data Imputation and Model Checking. URL http://CRAN.R-project.org/package=mi, r package version 0.09-11.03

Hardin J, Rocke DM (2005) The distribution of robust distances. Journal of Computational and Graphical Statistics 14:910–927

Hron K, Templ M, Filzmoser P (2010) Imputation of missing values for compositional data using classical and robust methods. Computational Statistics & Data Analysis 54(12):3095–3107

Huber PJ (1981) Robust Statistics. John Wiley & Sons

Hubert M, Rousseeuw P, Vanden Branden K (2005) ROBPCA: A new approach to robust principal component analysis. Technometrics 47:64–79

Hummel J (1996) Linked bar charts: Analysing categorical data graphically. Computational Statistics 11:233–33

Hundepool A, Van deWetering A, R R, Franconi L, Capobianchi A, DeWolf PP, Domingo-Ferrer J, Torra V, Brand R, Giessing S (2008) τ-ARGUS version 3.3 software and users manual. <http://neon.vb.cbs.nl/casc>

Hundepool A, Van deWetering A, R R, DeWolf PP, Giessing S, Fischetti M, Salazar J, Castro J, Lowthian P (2010) τ-ARGUS version 3.3 software and users manual. <http://neon.vb.cbs.nl/casc>

Ichim D (2008) Community innovation survey. a flexible approach to the dissemination of microdata files for research. Tech. rep., ESSNet on Statistical Disclosure Control

Johnson RA, Wichern DW (2002) Applied Multivariate Statistical Analysis. Prentice Hall, International, fifth edition

Jones MP (1996) Indicator and stratification methods for missing explanatory variables in multiple linear regression. Journal of the American Statistical Association 91(433):pp. 222–230

Kelly J, Golden B, Assad A (1990) Using simulated annealing to solve controlled rounding problems. Annals of Operations Research 2(2):174–190

Kooiman P, Willenbourg L, Gouweleeuw J (2002) A method for disclosure limitation of microdata. Tech. rep., Research paper 9705, Statistics Netherlands, Voorburg

Kraft S (2009) Simulation of a Population for the European Living and Income Conditions Survey. Master's thesis, Vienna University of Technology

Lawrence M, Lang DT (2008) RGtk2: R bindings for Gtk 2.8.0 and above. URL http://www.ggobi.org/rgtk2,http://www.omegahat.org, r package version 2.12.7

Little RJA, Rubin DB (1987) Statistical Analysis with Missing Data. John Wiley & Sons, New York

Little RJA, Rubin DB (1989) The analysis of social science data with missing values. Sociological Methods & Research 18:292–326

Little RJA, Smith PJ (1987) Editing and imputation for quantitative data. Journal of the American Statistical Association 82:58–69

Lopuhaä HP (1999) Asymptotics of reweighted estimators of multivariate location and scatter. The Annals of Statistics 27:1638–1665

Lopuhaä HP, Rousseeuw PJ (1991) Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. The Annals of Statistics 19:229–248

Lowman M (2009) Government statistics near and far. Journal of Advanced Analytics

Maronna RA, Zamar RH (2002) Robust estimation of location and dispersion for high-dimensional datasets. Technometrics 44:307–317

Maronna RA, Martin D, Yohai V (2006) Robust Statistics: Theory and Methods. John Wiley & Sons, New York

Mateo-Sanz J, Domingo-Ferrer J, Sebé F (2005) Probabilistic information loss measures in confidentiality protection of continuous microdata. Data Mining and Knowledge Discovery 11:181–193

Meindl B (2010) sdcTable: statistical disclosure control for tabular data. URL http://cran.r-project.org/package=sdcTable, r package version 0.0.12

Meraner A (2010) Outlier detection for semi-continuous variables. Masters thesis, Vienna University of Technology, Vienna

Merola G (2003) Generalized risk measures for tabular data

Muennich R, Rässler S (2004) Variance estimation under multiple imputation. In: Proceedings of Q2004 European Conference on Quality in Survey Statistics, Mainz, p 19

Münnich R, Schürle J (2003) On the simulation of complex universes in the case of applying the German Microcensus. DACSEIS research paper series No. 4, University of Tübingen

Münnich R, Schürle J, Bihler W, Boonstra HJ, Knotterus P, Nieuwenbroek N, Haslinger A, Laaksonen S, Eckmair D, Quatember A, Wagner H, Renfer JP, Oetliker U, Wiegert R (2003) Monte Carlo simulation study of European surveys. DACSEIS Deliverables D3.1 and D3.2, University of Tübingen

Peng R (2006) Interacting with data using the filehash package for R. Working paper 108, Johns Hopkins University, Dept. of Biostatistics, URL http://www.bepress.com/jhubiostat/paper108

R Development Core Team (2011) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, URL http://www.R-project.org/, ISBN 3-900051-07-0

Raghunathan T, Lepkowski J, Hoewyk J (2001) A multivariate technique for multiply imputing missing values using a sequence of regression models. Survey Methodology 27(1):85–95

Raghunathan T, Reiter J, Rubin D (2003) Multiple imputation for statistical disclosure limitation. J Off Stat 19(1):1–16

Reiter J (2009) Using multiple imputation to integrate and disseminate confidential microdata. Int Stat Rev 77(2):179–195

Riani M, Atkinson AC, Cerioli A (2009) Finding an unknown number of multivariate outliers. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 71(2):447–466

Rousseeuw PJ, Leroy AM (1987) Robust Regression and Outlier Detection. John Wiley & Sons, New York

Rousseeuw PJ, van Zomeren BC (1990) Unmasking multivariate outliers and leverage points. Journal of the American Statistical Association 85:633–651

Rubin D (1987) Multiple Imputation for Nonresponse in Surveys. Wiley, New York

Rubin DB (1993) Discussion: Statistical disclosure limitation. Journal of Official Statistics 9(2):462–468

Schafer J (1997) Analysis of Incomplete Multivariate Data. Chapman and Hall, London

Serneels S, Verdonck T (2008) Principal component analysis for data containing outliers and missing elements. Computational Statistics & Data Analysis 52(3):1712–1727, URL http://ideas.repec.org/a/eee/csdana/v52y2008i3p1712-1727.html

Skinner C, Shlomo N (1998) Estimating the re-identification risk per record in microdata. Journal of Official Statistics 14:361–372

Skinner C, Shlomo N (2006) Assessing identification risk in survey microdata using log-linear models. S3RI Methodology Working Papers M06/14, University of Southampton, Southampton Statistical Sciences Research Institute

Slater G (2011) International yearbook of industrial statistics 2010. Industrial Relations Journal 42(4):404–405

Stahel WA (1981) Robuste schätzungen: Infinitesimale optimalität und schätzungen von kovarianzmatrizen. Ph.d. thesis no. 6881, Swiss Federal Institute of Technology (ETH), Zürich, URL http://e-collection.ethbib.ethz.ch/view/eth:21890

Steel P, Reznek A (2005) Issues in designing a confidential preserving model server. In: Monographs in Official Statistics. Worksession on Statistical Data Confidentiality, Eurostat, Luxembourg

Takemura A (1999) In: Statistical Data Protection, Eurostat, Luxembourg

Templ M (2006) In: Privacy in Statistical Databases, Lecture Notes in Computer Science, vol 4302, Springer, Heidelberg, chap Software Development for SDC in R, pp 347 – 359

Templ M (2007) sdcMicro: A new flexible R-package for the generation of anonymised microdata - design issues and new methods. In: to appear in: Joint UNECE/Eurostat Work Session on Statistical Data Confidentiality. Monographs of Official Statistics.

Templ M (2008) Statistical disclosure control for microdata using the R-package sdcMicro. Transactions on Data Privacy 1(2):67–85, URL http://www.tdp.cat/issues/abs.a004a08.php

Templ M (2009a) A graphical user interface for microdata protection which provides reproducibility and interactions: the sdcMicro GUI. Transactions on Data Privacy 2(3):207–224, URL http://www.tdp.cat/issues/tdp.a030a09.pdf

Templ M (2009b) New Developments in Statistical Disclosure Control. Robust Statistics Applied to Official Statistics. Suedwestdeutscher Verlag fuer Hochschulschriften, Saarbruecken, Germany

Templ M (2010) sdcMicro: Statistical Disclosure Control methods for the generation of public- and scientific-use files. URL http://CRAN.R-project.org/package=sdcMicro, r package version 2.6.5

Templ M, Alfons A (2010) Disclosure risk of synthetic population data with application in the case of EU-SILC. In: Domingo-Ferrer J, Magkos E (eds) Privacy in Statistical Databases, Lecture Notes in Computer Science, vol 6344, Springer, Heidelberg, pp 174–186

Templ M, Alfons A (2011) VIM: Visualization and Imputation of Missing Values. URL http://CRAN.R-project.org/package=VIM, r package version 2.0.4

Templ M, Filzmoser P (2008) Visualization of missing values using the R-package VIM. Reserach report cs-2008-1, Department of Statistics and Probability Therory, Vienna University of Technology

Templ M, Meindl B (2008a) Robust statistics meets SDC: New disclosure risk measures for continuous microdata masking. In: Domingo-Ferrer J, Saygin Y (eds) Privacy in Statistical Databases, Lecture Notes in Computer Science, vol 5262, Springer, Heidelberg, pp 113–126

Templ M, Meindl B (2008b) Robustification of microdata masking methods and the comparison with existing methods. In: Domingo-Ferrer J, Saygin Y (eds) Privacy in Statistical Databases, Lecture Notes in Computer Science, vol 5262, Springer, Heidelberg, pp 177–189

Templ M, Meindl B (2010) Privacy and Anonymity in Information Management Systems. New Techniques for New Practical Problems, Springer, Berlin Heidelberg, chap Practical Applications in Statistical Disclosure Control Using R, pp 31 – 62. ISBN: 978-1-84996-237-7

Templ M, Kowarik A, Filzmoser P (2010) EM-based stepwise regression imputation using standard and robust methods. Reserach report cs-2010-3, Department of Statistics and Probability Therory, Vienna University of Technology

Todorov V (2010) R in the statistical office: The UNIDO experience. UNIDO Staff Working Paper, Vienna

Todorov V, Filzmoser P (2009) An object oriented framework for robust multivariate analysis. Journal of Statistical Software 32(3):1–47, URL http://www.jstatsoft.org/v32/i03/

Todorov V, Templ M, Filzmoser P (2011) Detection of multivariate outliers in business survey data with incomplete information. Advances in Data Analisys and Classification 5:37–56

Tufte ER (2010) Beautiful evidence. Graphics Press, Cheshire , Connecticut

Tukey JW (1962) The Future of Data Analysis. The Annals of Mathematical Statistics 33(1):1–67

Tukey JW (1977) Exploratory Data Analysis. Addison-Wesley

UNIDO (1996) Industrial statistics database: Methodological notes. Tech. rep.

UNIDO (2012) International Yearbook of Industrial Statistics. Edward Elgar Publishing Ltd, Glensanda House, Montpellier Parade, Cheltenham Glos GL50 1UA, UK

Upadhyaya S (2010) Compilation of energy statistics for economic analysis. UNIDO Staff Working Paper, Vienna

Upadhyaya S, Todorov V (2008) UNIDO data quality. UNIDO Staff Working Paper, Vienna

Vanden Branden K, Verboven S (2009) Robust data imputation. Computational Biology and Chemistry 33(1):7–13

Venables WN, Ripley BD (2003) Modern Applied Statistics with S, 4th edn. Springer

Verboven S, Vanden Branden K, Goos P (2007) Sequential imputation for missing values. Computational Biology and Chemistry 31(5-6):320–327

Verzani J, Lawrence M (2009) gWidgetsRGtk2: Toolkit implementation of gWidgets for RGtk2. published online, <http://cran.r-project.org/web/packages/gWidgetsRGtk2/index.html>

Walczak B, Massart D (2001) Dealing with missing data. Part I. Chemometrics and Intellegent Laboratory Systems 58:15–27

Wegman E (1990) Hyperdimensional data analysis using parallel coordinates. Journal of the American Statistical Association 85:664–675

Willenborg L, De Waal T (2000) Elements of statistical disclosure control. ISBN: 0387951210

Index

Glossary

`Inf`
: In R `Inf` and `-Inf` are positive and negative *infinity*. These apply to numeric values and real and imaginary parts of complex values but not to values of integer vectors. See also `NA`, `NaN` and `NULL` , 10

`NA`
: R representation of missing value (stands for *Not Available*). See also `NaN`, 7

`NULL`
: R representation of the null object (reserved word). `NULL` is often returned by expressions and functions whose value is undefined. Assigning `NULL` to an object deletes it, 8

`NaN`
: R representation of *impossible values*, e.g. division by 0 (stands for *Not a Number*). Computations involving `NaN` will return `NaN` or perhaps `NA`: which of those two is not guaranteed and may depend on the R platform (since compilers may re-order computations). See also `NA`, 10

CLI
: Command Line Interface, 74

Consumer Price Index (CPI)
: The Consumer Price Index (CPI) measures changes in the prices of goods and services that households consume (ILO: http://www.ilo.org/public/english/bureau/stat/guides/cpi/index.htm), 41

Gross fixed capital formation (GFCF)
: Gross fixed capital formation refers to the value of purchases and own account construction of fixed assets during the reference year less the value of corresponding sales, 41

Gross Output (GO)
The measure of gross output normally reported is the census concept, which covers only activities of an industrial nature. The value of census output in the case of estimates compiled on a production basis comprises: (a) the value of sale of all products of the establishment; (b) the net change between the beginning and the end of the reference period in the value of work in progress and stocks of goods to be shipped in the same condition as received; (c) the value of industrial work done or industrial services rendered to others; (d) the value of goods shipped in the same condition as received less the amount paid for these goods; and (e) the value of fixed assets produced during the period by the unit for its own use. In the case of estimates compiled on a shipment basis, the net change in the value of stocks of finished goods between the beginning and the end of the reference period is also included, 40

GUI
Graphical User Interface, 41, 74

IMF/IFS
International Monetary Fund/International Financial Statistics, 40

Index of Industrial Production (IIP)
The index of industrial production is designed to show the real (i.e. free of price fluctuation) change of production. Currently, its base year is 2000, 41

INDSTAT 2
The UNIDO Industrial Statistics Database comprises data for more than 175 countries and territories spanning the period 1963 to the latest available year. INDSTAT 2 contains the data reported according to ISIC Revision 3 at the 2-digit level, 40

ISIC
International Standard Industrial Classification of All Economic Activities, 40

Manufacturing Value Added (MVA)
The manufacturing value added (MVA) is a measure of the total value of goods and services produced by the manufacturing sector as defined in ISIC Revision 3. MVA is calculated as the difference of gross output and intermediate consumption according to the concepts recommended in the system of national accounts (SNA). MVA represents the part of GDP originating from manufacturing activities, 41

MAR
Missing at Random - the missing values are at random, given all the information available in the data set, 6

MCAR Missing Completely at Random - implies that the pattern of missing values is totally random, and does not depend on any variable, which may or may not be included in the analysis, 6

MNAR Missing Not at Random - means that there is an unknown process in the data that generates the missings., 6

NACE Statistical Classification of Economic Activities in the European Community, commonly referred to as NACE, is a European industry standard classification system consisting of a 6-digit code, 57

Number of employees The number of employees includes all persons engaged other than working proprietors, active business partners and unpaid family workers, 40

Number of establishments An "establishment" is ideally a unit that engages in one or in predominantly one kind of activity at a single location under single ownership or control, for example, a workshop or factory., 40

Record Linkage Record linkage refers to a merging that brings together information from two or more sources of data with the objective of consolidating facts concerning an individual or an event that are not available in any separate record (OECD Glossary of Statistical Terms, http://stats.oecd.org/glossary/detail.asp?ID=3103), 64

Selective editing Selective editing is a procedure which targets only some of the micro data items or records for review by prioritizing the manual work and establishing appropriate and efficient process and edit boundaries (OECD Glossary of Statistical Terms), 20

Statistical data editing Statistical data editing is the activity aimed at detecting and correcting errors (logical inconsistencies) in data. Graphical editing uses graphs to identify anomalies in data (OECD Glossary of Statistical Terms, http://stats.oecd.org/glossary/detail.asp?ID=3103), 20

Statistical Disclosure Control (SDC)

Statistical Disclosure Control techniques can be defined as the set of methods for reducing the risk of disclosing information on individuals, businesses or other organizations. Such methods are only related to the dissemination step and are usually based on restricting the amount of or modifying the data released (EUROSTAT: http://epp.eurostat.ec.europa.eu/portal/page/portal/research_methodology/methodology/statistical_disclosure_control, 52

UNECE

United Nations Economic Commission for Europe, http://www.unece.org/, 52

Value Added (VA)

The measure of value added normally reported is the census concept, which is defined as the value of census output less the value of census input, which comprises: (a) value of materials and supplies for production (including cost of all fuel and purchased electricity); and (b) cost of industrial services received (mainly payments for contract and commission work and repair and maintenance work), 41

Wages and salaries (WS)

Wages and salaries include all payments in cash or in kind paid to "employees" during the reference year in relation to work done for the establishment, 40

Printed in Austria
V.12-53185—May 2012—200

UNITED NATIONS INDUSTRIAL DEVELOPMENT ORGANIZATION
Vienna International Centre, P.O. Box 300, 1400 Vienna, Austria
Telephone: (+43-1) 26026-0, Fax: (+43-1) 26926-69
E-mail: unido@unido.org, Internet: www.unido.org