sdcMicro: a new flexible R-package for the generation of anonymised microdata: Design issues and new methods

Matthias Templ*

* Department of Methodology, Statistics Austria, Guglgasse 13, 1110 Vienna, Austria. (matthias.templ@statistik.gv.at) and

* Department of Statistics and Probability Theory, Vienna University of Technology, Wiedner Hauptstr. 8-10, 1040 Vienna, Austria. (templ@tuwien.ac.at)

Abstract. Data protection specialists need flexible software tools for the exploratory use of protection methods to generate high quality confidential data. Microdata protection is widely used and is often the only possible way to provide data to both researchers and users. In this paper we present a methodological and computational framework for the generation of anonymised microdata and give insights to the developed **R**-package `sdcMicro`. This package may become the standard software for microdata protection since it is very flexible, easy to use and contains all popular methods, plus some new ones. The package can also be used for comparison of methods and of original versus perturbed data not only by measuring information loss but also by various comparison plots.

1 Motivation

Nowadays there are various methods for dealing with confidential microdata developed to make such data accessible to researchers and users. Roughly speaking, these methods can be clustered into 3 categories:

(i) Remote Access based on the idea of hidden data and checked outputs.

(ii) Remote Access based on the idea of a free view of the data without the possibility to download the data.

(iii) perturbed microdata publicity as *public* or *scientific use files* or different variants between public and scientific use files.

In the following there is a short motivation of point (iii) since many researchers do not see a future in this point (iii) and therefore do not see the necessarity for a (new) software on microdata protection.

Regarding (i) researchers can apply models on data which can not be seen or where data and particularly outliers are perturbed [12, see e.g. in] but they can apply models. This approach is often called *confidential preserving model servers*. But it is the results of a models which are the objects of interest not the underlying data [22]. Considering a *Remote Execution* framework these results are checked by the census staff and, for example, in case of regression analysis synthetic residuals are often provided. But then, you see synthetic residuals based on residuals from really worst estimates because of using least squares regression on data which, naturally, includes outliers. Furthermore, you would never have the ability to find a good model without applying the whole range of diagnostic tools on robust estimates. This is in contradiction to [11] and others (e.g. [22]) because for such robust estimates it is improbably to provide synthetic residuals (because you always see large or heavy perturbed residuals/outliers), for example, and there is no method available for the detection of leverage points. It is also well known in robust statistics literature that you need robust methods for the detection of outliers which are essential and which cannot be provided. We can conclude that model servers are not compatible with a modern statistical world unless the underlying data is of a multivariate normal distribution or can be transformed into one which turns out to be unrealistic for real complex data. Concluding that, model servers are not compatible with a modern statistical world with the expection the underlying data is of a or can be transformed to a multivariate normal distribution which is unrealistic for real complex data.

In the second approach (ii) researchers may look at the data and can choose a suitable method for analysing the data, but it is not possible to download the data in any way (for applications in (ii) see e.g. [14] or [2]). Remote access can only partially be applied in some countries depending on the discipline from which the data originates. This is due to different legacy for data coming from different disciplines. In Austria, for example, only microdata protection is reasonable for official statistics because of the legal situation which prohibits a view on "original" data.

In (iii) users and researchers have access to perturbed data. In this paper we will show that such a perturbation can be easily performed with newly developed and extremely flexible software package by minimizing information loss and re-identification.

In addition to that, we will also concentrate on the design of this new package for microdata protection and illustrate the open source philosophy of this project.

2 Using R for SDC

R [20] is an open source high-level statistical computing environment subjected to the General Public License and therefore freely available and expendable. Furthermore, **R** has become the standard statistical software and thousands of people are

involved in the development of **R** both at universities and companies. More than 1000 add-on packages have been built in the last years.

In the following the usefulness of **R** related to SDC is presented. The most important and most essential feature of a software for SDC is the reproducibility of results. The most effective way of anonymising microdata is by doing the anonymisation steps in an explorative and in some sense iterative way, since we can apply various methods on various variables with different parameters producing different effects on the data and while looking for sufficient anonymisation of the data with respect to low information loss. Therefore, we must have the ability to reproduce any step of the anonymisation process easily. This is fulfilled by running a script with all the commands included. It is then easy to change and adapt this script and get all the new results "in real time". Of course, this is fulfilled by many software tools, but the real advantage of using **R** is that we can interactively "play" with the data, i.e. we have access to all the objects in the workspace of **R** any time and can change, display or apply operations on it on the fly. This is very useful during the anonymisation of data and a quite different concept than to write a "batch file" which can then be executed.

In addition to that, we recommend the use of a powerful easy-to-grasp visualization tools to see the effect of the anonymisation on the data instead of computing various measures of information loss (see e.g. in [23]).

But dynamical reports can also be used for documenting the anonymisation when using **R** in combination with `Sweave` ([17]). Reports can then be generated very effectively and quickly for particular steps in the production process.

Since many different possible data formats are used by users it is very comfortable to have the opportunity to import and export data formats from data base software and statistical software like `SPSS`, `SAS`, `Stata`, `DBF`, `Excel`, `ASCII`, and many more. Sometimes it is also useful to run the anonymisation tools via batch mode, which can be easily derived with package `sdcMicro` [24].

It is also very important to provide a software for SDC which is platform independent and works on all common operating systems.

All these things are supported when using **R**.

An important and essential feature of any software dealing with data is the reproducibility of results. An advantage of a command line interface (CLI) like in **R** is that any result can be reproduced very easily. Furthermore, all the code is open-source and one can have a detailed look at the code in each of the steps of the microdata protection procedure for any method. Also very important is the reproducibility of results when applying different algorithms for SDC on data and doing a validation and comparison of these different results.

R has very powerful graphical tools implemented and one can generate simple and complex user-defined graphics easily. This is very important since mostly it is better to show perturbation effects or other effects resulting from the protection

process with suitable plots as with some measures of information loss (see e.g. also in [23]).

In combination with LaTeX someone can produce dynamical reports, where LaTeX-code and **R**-code can be written and executed in/from one document with the help of `Sweave` [17]. Of course, one can also standardise such dynamical reports to get automatically all the results from any data in a standardised format. Such reports can then generated very effective and could save time for particular steps in the production process.

Nearly all common data formats from database software and other statistical software like SPSS, SAS, STATA, MINITAB, DBF, ... can be imported. Furthermore, there are also procedures available to export the data in most of these data formats.

3 Design Goals of Package `sdcMicro`

When perturbing numerical data the optimal perturbation method depends strongly on the multivariate structure of the data. Therefore, a flexible tool with which one can try out various methods and easily compare the methods is necessary. Even when trying out different global recordings on variables or when playing on some parameters from certain methods a flexible tool and easy to grasp comparison plots and additional evaluation tools are necessary.

For **R** users it is not natural to use a GUI and this might be the main reason why a GUI is only partially implemented for *sdcMicro* till now. But, of course, it is possible to create a GUI in top of the existing code for *sdcMicro*, if there will be a demand for this, since with **R** one can produce powerful GUI's, e.g. with the *Rtcltk* package. Nevertheless, the design of the package allows an easy use and not only one of the numerous users ask for a GUI until now.

One can also run the package via batch mode. Furthermore, one can also run directly from **R** any other software which might be also sometimes useful.

The advantages of an object-oriented programming language become apparent in the `scdMicro` package. However, the methods and class approach from **R** is not a class-oriented programming language (like `C` or `Java`) but a richer one as it is a function- and class-oriented programming language.

In **R** everything is an object and every object is related to a specific class. The class of an object determines how it will be treated and generic functions perform either a task or an action on its arguments specific to the class of the argument itself. The class mechanism offers the programmer the facility of designing and writing generic functions for special purposes which is extensively used in *sdcMicro*. Nearly all functions, e.g. the one for the individual risk methodology or the frequency calculation, produce objects from a certain class. Different *print*, *summary* and *plot* methods are provided for each of these objects depending on their class. `plot(ir1)`

produces a completely different result than `plot(fc1)` assuming that the objects `ir1` and `fc1` are objects from different classes, i.e. resulting from different functions in package *sdcMicro*.
This object-oriented approach allows a simple usage of the package for any user, independently of the proficiency in **R**. Furthermore, users can try out different methods with different parameters and they can easily compare the methods with the implemented summary and plot methods.

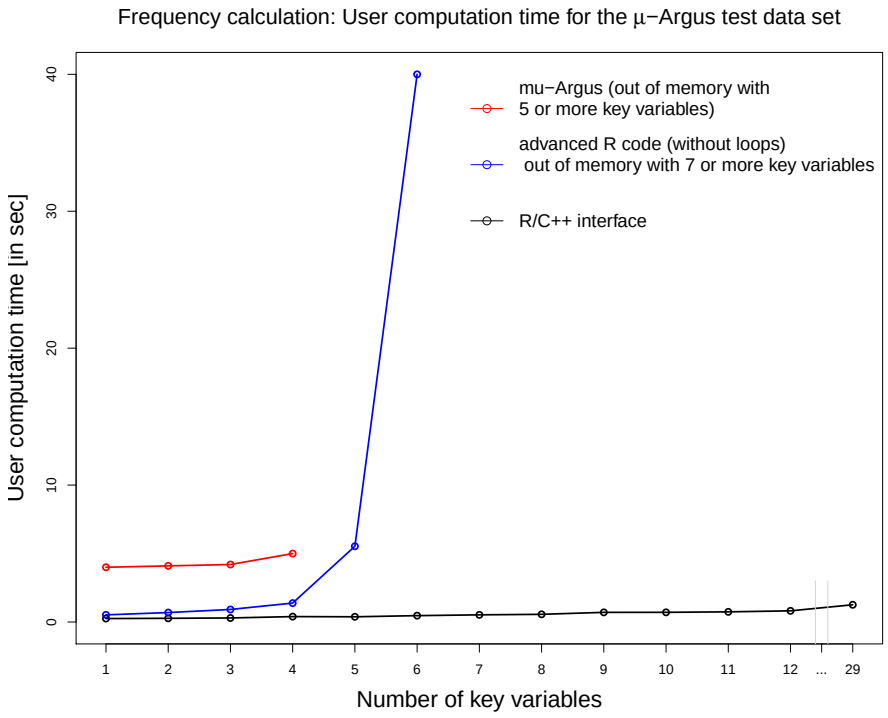
Note that no metadata management needs to be done by the user, even not after importing the data into **R**. You can apply the methods directly on your data sets or on objects for certain classes. At first you must only determine which of the variables are the key variables, the weight vector and the numerical ones.
An online documentation is included in the package containing all explanations on all input and output parameters. Furthermore, various examples are included for each of the functions. These examples can be easily executed by the users.

To be able to deal with large data sets extensive computational calculation steps are implemented in `C++` and included in **R** via the `R/C++` interface. The calculation of the frequency counts is one of the most critical calculation steps regarding to the computation time. Figure (**??**) shows the calculation time carried out by an one year old personal computer[1] with Windows XP operating system. The computation time can not be shown in this figure because it turns out to be too large when using loops in **R** for the calculation of the frequency counts because it is too large. Only when using functions in **R**, which are said to be implemented in `C` or `Fortran`, it is possible to calculate the frequency counts with up to 6 key variables by using this relatively small data set with only 4000 observations. This is not possible for the μ-Argus system ([15], version 4.1.0) which runs out of memory with 5 or more key variables. The developed `R/C++` can also handle a very large number of key variables[2] in reasonable time even for much larger data sets than the μ-Argus test data set. In Figure (**??**) you can easily see that the computation time is always less than 1 second for this small data set and it is also low for larger data sets.

4 Implemented Methods and Data

Methods like global recoding, local suppression and the individual risk methodology (see e.g. in [10]), top- and bottom-coding, PRAM ([16]), more than 10 microaggregation methods (mdav, pca-methods, robustified pca-methods with fast algorithms and clustering, individual ranking, methods based on robust Mahalanobis distances, ...) (more information on these methods and on microaggregation can be found in [23], [1], [9], [7], [6], [8]), rank swapping [5], 5 well known adding noise methods

[1] Intel x86 based system with 3Ghz and 1 GB memory
[2] to choose such a large number of key variables is, of course, not always meaningful.



(see e.g. in [3]) including ROMM [25] and various other methods are implemented. There is also a method for fast generation of synthetic data included [18] with which multivariate normal distributed data can be generated regarding to the covariance of the original data, but, of course, this does not reflect the distribution of real complex data. Further methods will be implemented in near future (see below in section *Future Developments*).

Several test data sets are implemented in the package. There are some very small test data sets which used by other authors for demonstration in the past. The test data set from μ-ARGUS [15] can also be used as well as the test data sets from the CASC project [13].

4.1 New Methods

Some new algorithms for microaggregation are proposed and implemented in package `sdcMicro`. A simple approach is to cluster the data at first and then sort the data in each group by the most influential variables in the groups or by sorting the observations in each group by the first robust principal component using e.g. the MCD-estimator [21].

We proposed a new algorithm for microaggregation called RMDM (**R**obust **M**ahalanobis **D**istance based **M**icroaggregation where MDAV [8] is adapted in the following way:

1. Compute the robust center of the data. This can be the L_1-median or the coordinate-wise median.

2. Consider the most distant observation x_r to the robust center using robust Mahalanobis distances. The MCD-Estimator can be used to calculate the robust covariance matrix that is needed for the calculation of the robust Mahalanobis distances.

3. Find the most distant observation x_s by calculating robust Mahalanobis distances with the center in x_r.

4. Choose $k-1$-nearest neighbors from x_r and also from x_s using robust Mahalanobis distances computed with center x_r and center x_s. Aggregate x_r and its $k-1$ nearest neighbors with an average as well as x_s with their $k-1$ nearest neighbors. The average can be the arithmetic mean but also robust measures of location.

5. Take the previous dataset minus the aggregated data from the last step as a new dataset and continue with (1.) until all observations are microaggregated.

Note, that there are special rules at the end of the algorithm which are described in [8]. This proposed algorithm is more natural than the original MDAV algorithm since we deal with multivariate data taking the covariance structure of the data into account. For larger data sets usual distances can be chosen to find the nearest neighbors in item (4.) of the algorithm (we name this adaption of the algorithm RMDM2). These distances must be calculated only once at the beginning of the algorithm.

Robust versions of principal component methods for microaggregation are implemented, too.

The *clustpppca* and the *RMDM2* algorithms for microaggregation worked best for most of the data sets (like the Tarragona data set from the CASC project (see section 5)). The *clustppca* algorithm is described in [23].

Top and bottom coding can be replaced by (multivariate) outlier detection via robust statistics. Only those observations are recoded/perturbed which are outliers and therefore of high re-identification risk. Such a method is included as an adding noise procedure.

5 A Small Tour in sdcMicro

Within the limitation of pages only a small tour in *sdcMicro* can be done excluding most of the graphical results and some steps of recoding variables. Comments are marked with #, the output from **R** with **R**. For further details, please have a look at the examples and documentation which are included in package *sdcMicro*.

Supposing you have already downloaded and installed **R** and have also installed package `sdcMicro`, which can be installed directly from **R** or downloaded directly from *http://cran.r-project.org/src/contrib/Descriptions/sdcMicro.html*, you can load both the package and the data and print[3] the first seven columns and the first four rows of the data with the following command in **R**:

```
library(sdcMicro); data(free1); xtable(free1[1:4, 1:8])
```

	REGION	SEX	AGE	MARSTAT	KINDPERS	NUMYOUNG	NUMOLD
1	36.00	1.00	43.00	4.00	3.00	0.00	0.00
2	36.00	1.00	27.00	4.00	3.00	0.00	0.00
3	36.00	1.00	46.00	4.00	1.00	0.00	0.00
4	36.00	1.00	27.00	4.00	1.00	0.00	0.00

For this demonstration the μ-Argus test data was chosen but you can simply use another data set from the package or your own data.

In the following, the frequency counts are calculated as described in [4] and allocated to object `fr1` which is now automatically of class `freqCalc`. Several methods are available for this class. Object `fr1` can then be used as an input for the individual risk computation. A new object `ir1` will be produced which is of class `indivRisk`. Several methods are again available for this class and, for example, function `plot` will automatically know which plot method must be used. Figure (1) is generated by this plot method. The implementation of this plot method for individual risk methods is quite similar as in μ-Argus.

```
fr1 <- freqCalc(free1, keyVars=1:3, w=30)
rk1 <- indivRisk(fr1)
class(rk1)
R> [1] "indivRisk"
methods(class = indivRisk)
R> [1] plot.indivRisk  print.indivRisk
plot(rk1)
```

The script shown above can easily be adapted (exclude the **R** results), e.g. by adding the function `globalRecode()`. `globalRecode()` recodes several categories of a variable into less categories or discretize a numerical variable. So, this function checks the class of the variable and does the recoding based on the class of the variable. But you still have the ability to manipulate your data in a explorative way. You might try out recoding a variable to observe the influence of your recoding on the frequency count calculation and the individual risk computation. Note that you can easily reproduce all your steps easily either by running a part of your script or the whole script.

[3] `xtable()` produces a LaTeX-styled print output.

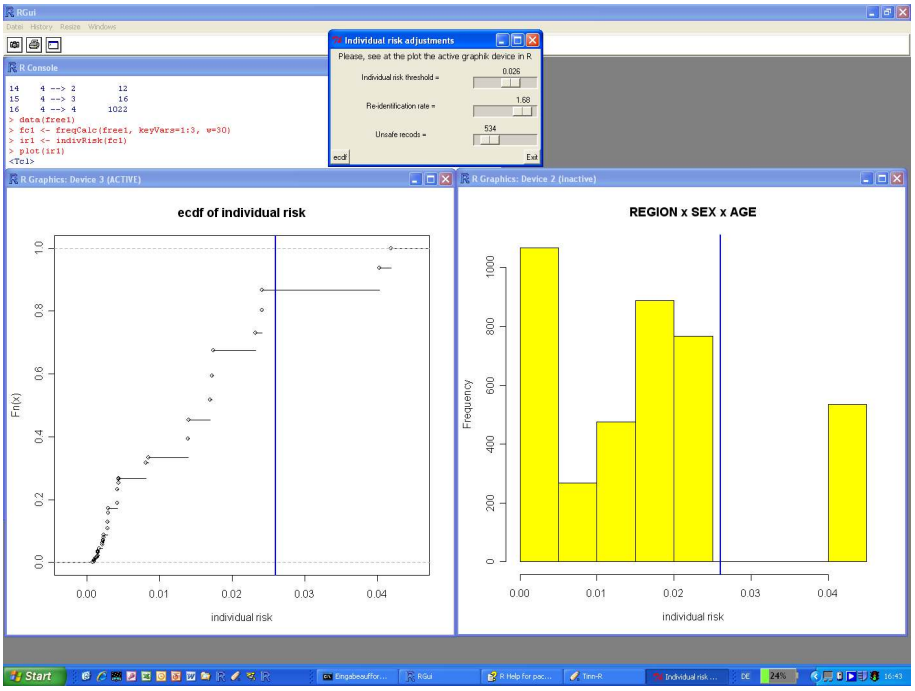


Figure 1: Individual risk for the first 3 variables of the μ-Argus test data set. In the upper region of the figure you will see a helpful slider which is directly linked with the graphics.

After minimising the re-identification risk you can apply local suppression (function `localSupp()`) on object `indivf` and `fr1` to delete the last unsureness about risky observations and make then use of the implemented print and summary methods. But, also global recoding and local suppression can also be used in an alternated manner, of course.

In addition to that, you can simply microaggregate numerical variables with more than 10 different methods. Furthermore, you can also use rank swapping and adding noise methods. There are a lot of comparison plot methods available to compare the perturbed data and the original data. You can also easily compare the different methods themselves. We will show this on another data set, the *Tarragona* data set[4], because the μ-Argus test data does include faked numerical variables which follow a multivariate uniform structure without correlations between the variables.

Microaggregation and a comparison of different methods with various measures of information loss can be done in the following way:

```
m1 <- microaggregation(Tarragona, method="onedims", aggr=3) # individual ranking method
```

[4]http://neon.vb.cbs.nl/casc/testsets.html

```
# now you can use some comparison plots... (see examples of package sdcMicro)
summary(m1) # or the implemented print and summary methods (suppressed)
xtable(valTable(Tarragona, method=c("addNoise: correlated2", "swappNum", "simple",
"onedims", "pca", "mdav", "clustpppca"))[,c(1,2,3,5,6,7,12)]) # measures of inf. loss
```

	method	amean	amedian	devvar	amad	acov	apcaload
1	addNoise: correlated2	0.09	3.20	0.18	5.89	0.09	8.97
2	swappNum	0.20	0.13	9.10	0.23	4.55	28.75
3	simple	0.00	3.50	3.62	1.11	1.81	20.69
4	individual ranking	0.00	0.02	13.71	0.03	6.85	19.45
5	pca	0.00	2.62	2.77	1.10	1.39	12.99
6	clustpca	0.00	2.34	2.69	1.11	1.35	11.72
7	mdav	0.00	4.18	3.44	1.84	1.72	8.76
8	rmdm2	0.00	1.51	1.76	0.58	0.88	12.00
9	clustpppca	0.00	3.64	2.79	1.55	1.40	10.69

The generated table from function `valTable()` shows that the proposed method RMDM2 (which is explained in section 4.1) performs best (see a few description on the information loss measures used in [23]) on this data set which is not surprisingly because the multivariate structure of the data is taken into account when using Mahalanobis distances.

In addition to that, various measures of risk and data utility can be applied on these results, e.g. with function `dRisk()` and `dUtility()`.

Another method for categorical variables is PRAM which can be easily applied with function `pram()`. In the following, variable `MARSTAT` form the μ-Argus test data set will be perturbed with the invariant PRAM methodology. A lot of information is stored in object `MARSTAT`, e.g. the invariant transition matrix. Summary and print methods provided as well.

```
MARSTAT <- pram(free1[,"MARSTAT"], p=0.8, alpha=0.5)
summary(MARSTAT)
 ----------------------          ----------------------       ----------------------
 original frequencies:          transitions:                  invariant transition matrix
                                    transition Frequency              [,1]    [,2]    [,3]   [,4]
   1    2    3    4             1     1 --> 1      2448       [1,] 0.9584 0.00903 0.00951 0.0230
2547  162  171 1120             2     1 --> 2        27       [2,] 0.1421 0.73943 0.02836 0.0900
                                 3     1 --> 3        28       [3,] 0.1416 0.02687 0.74079 0.0906
 ----------------------          4     1 --> 4        44       [4,] 0.0523 0.01302 0.01383 0.9207
 frequencies after perturb.:     5     2 --> 1        33
                                 6     2 --> 2       118
   1    2    3    4             7     2 --> 3         4
2571  160  178 1091             8     2 --> 4         7
                                 9     3 --> 1        20
                                 10    3 --> 2         3
```

```
11     3 --> 3        130
...     ...           ...
```

6 Open Source Initiative

As mentioned above, the whole code is free and can be downloaded on *http://cran.r-project.org*. So you can learn from this code, can change code for yourself or develop it further. Instead of keeping the developed code to yourself you are invited to contribute to this package. Every response and bug reports will be helpful in achieving a higher quality for the package. Note that the quality of the package was highly improved by comments and bug reports from many users from statistical offices and companies up to now.

Every function has its own author the copyright of the function is help by the author. This copyright means that nobody can use your function for a commercial software product and in the other hand the intellectual property is also ensured. But, at the same time all the functions are open-source and everybody can use it.

7 Conclusion

This package allows a flexible and explorative use of the most well known methods plus of various new methods. It allows the use of various comparison plots which are more informative than usual measures of information loss. New functionality like additional methods for synthetic data generation, blanking and imputation, etc. will be implemented soon. The potential capacity of this package can be very high, the package has a realistic chance to become the most important implementation for SDC in microdata protection. The respond of users from all over the world is very positive and the package is already used in the production process (see also [19]). The users are still satisfied with the command line interface and so an implementation of a graphical user interface has not been made yet. In addition to this package, the entire power of **R** can be used to boost the results in any way. Everybody is invited to contribute to this package, especially in funded future research projects.

References

[1] N. Anwar. Micro-aggregation - the small aggregates method. In *Internal report*. Luxembourg: Eurostat, 1993.

[2] L. Borchsenius. New developements in the danish system for access to micro data. In *Monographs of official statistics, Work session on statistical data confidentiality*. Eurostat, Luxembourg, 2005.

[3] R. Brand. Microdata protection through noise addition. In *Privacy in Statistical Databases. Lecture Notes in Computer Science. Springer*, pages 347–359, 2004.

[4] A. Capobianchi, S. Polettini, and M. Lucarelli. Strategy for the implementation of individual

risk methodology into μ-ARGUS. Technical report, Report for the CASC project. No: 1.2-D1, 2001.

[5] T. Dalenius and S. Reiss. Data-swapping: A technique for disclosure control. In *Proceedings of the Section on Survey Research Methods*, volume 6, pages 73–85, 1982.

[6] D. Defays and Anwar M.N. Masking microdata using micro-aggregation. *Journal of Official Statistics*, 14(4):449–461, 1998.

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

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

[9] M. Elliot, A. Hundepool, E.S. Nordholt, J-L. Tambay, and T. Wende. Glossary on statistical disclosure control, 2005.

[10] L. Franconi and S. Polettini. Individual risk estimation in μ-ARGUS: a review. In *Privacy in Statistical Databases. Lecture Notes in Computer Science. Springer*, pages 262–272, 2004.

[11] J. Heitzig. The 'jackkife' method: confidentiality protection for complex statistical analyses. In *Joint UNECE/Eurostat work session on statistical data confidentiality, Geneva, Switzerland*, 2005.

[12] J. Heitzig. Using the jackknife method to produce safe plots of microdata. In *Privacy in Statistical Databases. Lecture Notes in Computer Science. Springer*, pages 139–151, 2006.

[13] A. Hundepool. The casc project. In *Privacy in Statistical Databases. Lecture Notes in Computer Science. Springer*, pages 199–212, 2004.

[14] A. Hundepool and P. De Wolf. Onsite@home: Remote access at statistics netherlands. In *Joint UNECE/Eurostat work session on statistical data confidentiality, Geneva, Switzerland*, 2005.

[15] A. Hundepool, A. Van deWetering, Ramaswamy R., L. Franconi, A. Capobianchi, P-P. DeWolf, J. Domingo-Ferrer, V. Torra, R. Brand, and S. Giessing. μ-argus version 4.1 software and userŠs manual, 2006.

[16] P. Kooiman, L. Willenbourg, and J. Gouweleeuw. A method for disclosure limitation of microdata. Technical report, Research paper 9705, Statistics Netherlands, Voorburg, 2002.

[17] Friedrich Leisch. Sweave, part I: Mixing R and LaTeX. *R News*, 2(3):28–31, December 2002.

[18] J.M. Mateo-Sanz, A. Martínez-Ballesté, and J. Domingo-Ferrer. Fast generation of accurate synthetic microdata. In J. In: Domingo-Ferrer, editor, *Privacy in Statistical Databases, Lecture Notes in Computer Science*, pages 298–306. Springer, 2004.

[19] B. Meindl and M. Templ. The anonymisation of the CVTS2 and income tax dataset. an approach using R-package sdcmicro. In *to appear in: Joint UNECE/Eurostat Work Session on Statistical Data Confidentiality. Monographs of Official Statistics.*, 2007.

[20] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2007. ISBN 3-900051-07-0.

[21] P.J. Rousseeuw. Least median of squares regression. *Journal of the American Statistical Association*, 79:871–880, 1984.

[22] P. Steel and A. Reznek. Issues in designing a confidential preserving model server. In *Monographs of official statistics, Work session on statistical data confidentiality*. Eurostat, Luxembourg, 2005.

[23] M. Templ. Software development for SDC in R. In *Privacy in Statistical Databases. Lecture Notes in Computer Science. Springer*, pages 347–359, 2006.

[24] M. Templ. *sdcMicro. Manual and Package.* Statistics Austria and Vienna University of Technology, Vienna, Austria, 2007. *http://cran.r-project.org/src/contrib/Descriptions/sdcMicro.html*.

[25] D. Ting, S. Fienberg, and M. Trottini. Romm methodology for microdata release. In *Monographs of official statistics, Work session on statistical data confidentiality*. Eurostat, Luxembourg, 2005.