synthpop: An R package for generating synthetic versions of sensitive microdata for statistical disclosure control

Beata Nowok

Administrative Data Research Centre - Scotland, University of Edinburgh, School of GeoSciences, Drummond Street, Edinburgh EH8 9XP, United Kingdom, e-mail: beata.nowok@ed.ac.uk

Abstract. The *synthpop* package is an add-on package to the statistical software **R**. It provides routines to generate synthetic versions of original microdata containing confidential information so that they are safe to be released to users. Synthetic data are generated by replacing sensitive records with values simulated from probability distributions specified to preserve key features of the actual observed data. The package offers also tools to assess quality of the synthetic data sets, to apply additional means of confidentiality protection and to facilitate data import and export.

1 Introduction

Synthetic data techniques have been developed to allow for the release of high-quality microdata without compromising confidentiality (Rubin 1993). The general idea is to replace values at high risk of disclosure with simulations from models fitted to the original confidential data. In the most conservative case when all actual values are replaced, synthetic data comprise records of artificial individuals only and provide high level of data protection. Synthetic data have been recognized as an alternative to traditional disclosure control measures that may cause severe distortion of the statistical information contained in the original data. The use of synthetic data has been, however, limited. It is attributed partially to the fact that generating useful synthetic data sets is a complex process that requires expertise in both the data to be synthesised and statistical methods. The recent developments in this area and application of non-parametric methods, as suggested first by Reiter (2005), can simplify the creation of synthetic data considerably and expand its role played in confidentiality protection, especially when appropriate software is available. The **R** package *synthpop* aims to fill a gap in tools for generating and evaluating synthetic data of various kind. The goal of this paper is to present the current version of the software (*synthpop 1.2-0*). After brief presentation of the implemented synthesising method in Section 2, the functionality of the package is described and illustrated in Section 3. Section 4 contains concluding remarks.

2 Synthesising procedure in a nutshell

The specification of a joint distribution of all the variables in a data set is usually difficult in real data application. Therefore, an approximation by a series of conditional distributions is used instead, which is also the case in the *synthpop* package. Synthesis is performed on a variable by variable basis by fitting a sequence of regression models and drawing synthetic values from the corresponding predictive distributions. The fitted models are conditioned on the original variables that are earlier in the synthesis sequence, so the number of covariates increases for subsequent variables with the last being conditioned on all other variables. In this approach models, which can be parametric or non-parametric, can be defined for each variable separately and structural features of the data such as logical constraints or missing data patterns can be taken into account. Note that missing values are not imputed prior to synthesis and rather the observed patterns of missing data are aimed to be synthesised.

3 Using synthpop

The *synthpop* package is an add-on package to the statistical software **R** (R Core Team 2015). It is freely available from the Comprehensive **R** Archive Network (CRAN) at `http://CRAN.R-project.org/package=synthpop`. It can be downloaded and installed, for example, from inside an **R** session via

```
> install.packages("synthpop")
```

Once the *synthpop* package is installed, it needs to be attached to the current **R** session by the command

```
> library(synthpop)
```

The other documents available from CRAN include the so called vignette describing the package functionality and giving examples, the `NEWS` file documenting changes made in new releases of the package and reference manual. A help page for a specific function can be displayed using `?` operator, e.g.

```
> ?synthpop
```

The *synthpop* package includes also a microdata set `SD2011`, a subset of which selected below and stored as `ods` object, which stands for observed data set, is used to illustrate package functionality in the subsections that follow (see Appendix A for description of the variables used in this paper and `?SD2011` for more information about the data set)

```
> vars <- c("sex", "agegr", "placesize", "edu", "socprof", "marital",
+  "income", "smoke")
> ods <- SD2011[, vars]
```

3.1 Synthesis

The `syn()` function, the central function of the *synthpop* package, generates synthetic versions of a data set provided as its argument. The process of producing synthetic data can be largely automated, if default settings are used, because only data to be synthesised need to be given

```
> sds <- syn(ods)
```

The resulting object called here `sds`, which stands for synthetic data set, is a list that contains synthetic data set or sets (`sds$syn`) and values of parameters used in the synthesis. The `print` method displays its selected components. As presented in the `sds` output printed below, by default a single synthetic data set is generated (`sds$m`) and the "`ctree`" implementation of a classification and regression trees model (CART) (from *party* package) is used for all variables except the first one in the visit sequence (`sds$method`). The first variable to be synthesised does not have predictors and its synthetic values are generated by sampling with replacement.

```
> sds

Call:
($call) syn(data = ods)

Number of synthesised data sets:
($m)  1

First rows of synthesised data set:
($syn)
     sex agegr             placesize                     edu
1 FEMALE 16-24           RURAL AREAS                SECONDARY
2 FEMALE 25-34 URBAN 100,000-200,000 POST-SECONDARY OR HIGHER
3 FEMALE 45-59  URBAN 20,000-100,000     PRIMARY/NO EDUCATION
4 FEMALE 16-24           RURAL AREAS       VOCATIONAL/GRAMMAR
5   MALE 35-44           RURAL AREAS       VOCATIONAL/GRAMMAR
6   MALE 25-34           RURAL AREAS POST-SECONDARY OR HIGHER
                     socprof marital income smoke
1           PUPIL OR STUDENT  SINGLE     NA    NO
2           PUPIL OR STUDENT  SINGLE     NA    NO
3 EMPLOYED IN PRIVATE SECTOR MARRIED    960    NO
4           PUPIL OR STUDENT  SINGLE     NA    NO
5              SELF-EMPLOYED MARRIED   2000    NO
6  EMPLOYED IN PUBLIC SECTOR MARRIED   2300    NO
...

Synthesising methods:
($method)
      sex     agegr placesize       edu   socprof   marital    income     smoke
 "sample"   "ctree"   "ctree"   "ctree"   "ctree"   "ctree"   "ctree"   "ctree"
```

```
Order of synthesis:
($visit.sequence)
      sex     agegr placesize       edu   socprof   marital    income     smoke
        1         2         3         4         5         6         7         8

Matrix of predictors:
($predictor.matrix)
          sex agegr placesize edu socprof marital income smoke
sex         0     0         0   0       0       0      0     0
agegr       1     0         0   0       0       0      0     0
placesize   1     1         0   0       0       0      0     0
edu         1     1         1   0       0       0      0     0
socprof     1     1         1   1       0       0      0     0
marital     1     1         1   1       1       0      0     0
income      1     1         1   1       1       1      0     0
smoke       1     1         1   1       1       1      1     0
```

The `syn()` function offers a variety of options to customize a default synthesis and take into account specific characteristics of an individual data set. A model type that is provided using `method` parameter can be set for each variable separately. It can be chosen out of the methods currently implemented (see help page for function `syn()` for a list of methods) or a new synthesising method can be introduced by writing a function named `syn.newmethod()` and then specifying `method` parameter as `"newmethod"`. Next to model type specification, users can select the set of variables to include as predictors using `predictor.matrix` argument. The choice of explanatory variables is restricted by the order in which variables are synthesised. The synthesis order can be changed via `visit.sequence` and it is also possible to include as predictors variables that do not belong to the data set to be synthesised. The `syn()` function also includes procedures to deal with common data problems and to mimic the characteristics of the original confidential data in many possible ways. Optional parameters can be specified to take into account for instance structural features of the data such as missing data patterns, restricted values, linear constraints and deterministic relations between variables. Besides, there are options that are designed to increase data protection. For instance, a user can specify a minimum size of a final node that a CART model can produce. For `"ctree"`, `"cart"`, `"normrank"` and `"sample"` method the `smoothing` parameter can be set to `"density"` in order to apply Gaussian kernel density smoothing to the synthesised values. The following command illustrates the use of a few optional parameters. It generates five synthetic data sets using an alternative CART method (as implemented in *rpart* package), with minbucket set to ten in order to decrease disclosure risk. In addition a continuous variable income is smoothed and missing data codes (`NA` and `-8`) are synthesised separately

```
> sds <- syn(ods, m = 5, method = "cart", cart.minbucket = 10,
    cont.na = list(income = c(NA, -8)), smoothing = list(income = "density"))
```

3.2 Diagnostics

The *synthpop* package offers also tools to assess quality of the synthetic data sets. They compare, in tabular and graphical form, features of the synthetic data and models fitted to

them with the characteristics and estimate results for the original data. A generic function `compare()` invokes particular methods depending on the class of the first argument called `object`. If a synthetic data object is provided it compares relative frequency distributions of each variable in tabular and graphic form. By default the function generates plots for all variables in the synthetic data set but below it is used for selected variables specified by a `vars` argument. Output for a quantitative variable `income` is presented below and in Figure 1. Missing data categories are plotted on the same plot as non-missing values and they are indicated by `miss.` suffix. If a synthetic data object contains multiple data sets by default pooled synthetic data are used for comparison. An argument `msel` can be used to compare the observed data with a single or multiple individual synthetic data sets, which is illustrated below and in Figure 2 for an education factor variable (`edu`).

```
> compare(sds, ods, vars = "income")

Comparing percentages observed with synthetic

$income
              0  1000   2000  3000  4000  5000 6000 7000  8000  9000 10000 11000
observed  24.58 34.98  9.440 2.660 1.360 0.480 0.26 0.20 0.080 0.100 0.040 0.020
synthetic 21.46 35.46 10.796 3.796 1.516 0.736 0.40 0.24 0.072 0.056 0.048 0.044
          12000 13000 14000 15000 16000 miss.-8 miss.NA
observed  0.000     0  0.06 0.020 0.000  12.060  13.660
synthetic 0.012     0  0.04 0.028 0.004  11.804  13.488


> compare(sds, ods, vars = "edu", msel = 1:3)

Comparing percentages observed with synthetic

$edu
          PRIMARY/NO EDUCATION VOCATIONAL/GRAMMAR SECONDARY
observed                 19.24              32.26     29.64
syn=1                    20.40              31.46     28.46
syn=2                    19.68              31.38     30.12
syn=3                    19.52              32.18     30.18
          POST-SECONDARY OR HIGHER <NA>
observed                     18.72 0.14
syn=1                        19.54 0.14
syn=2                        18.72 0.10
syn=3                        17.96 0.16
```

If a result of a model fitted to the synthetic data set or sets using function `glm.synds()` or `lm.synds()` is provided to function `compare()` as its `object` argument, it compares the estimates based on the synthesised data set or sets with those based on the original data and presents the results in both tabular and graphical form (see Figure 3; the tabular output is suppressed for space reasons).

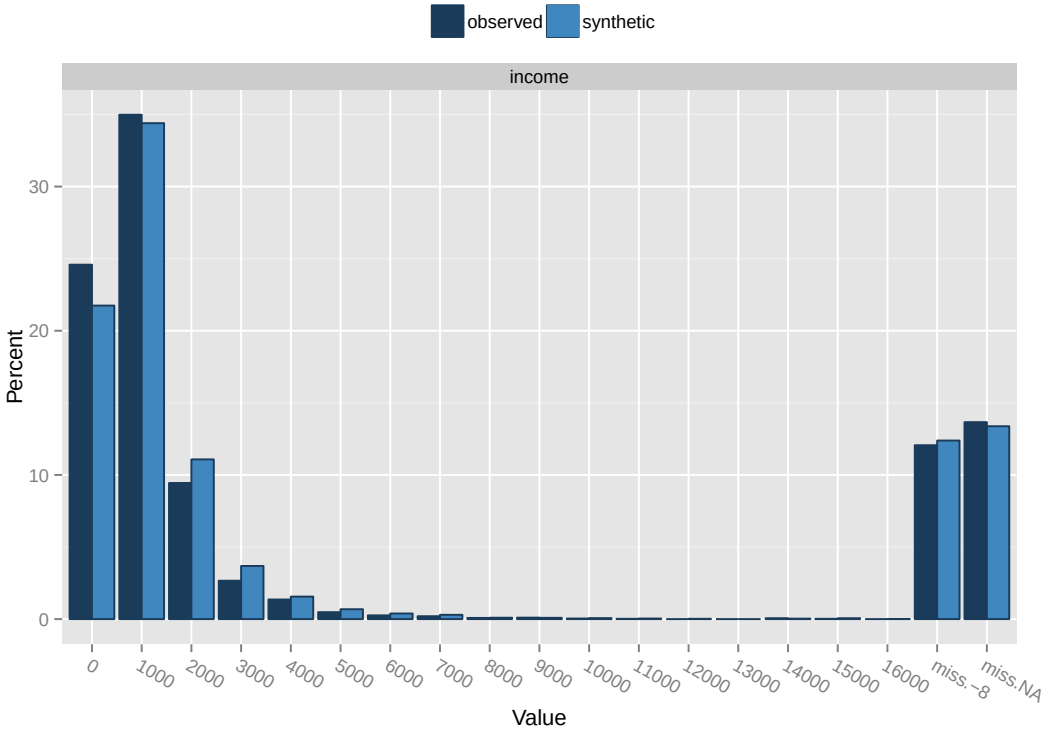

Figure 1: Relative frequency distribution of non-missing values and missing data categories for income variable (`income`) for observed and synthetic data.

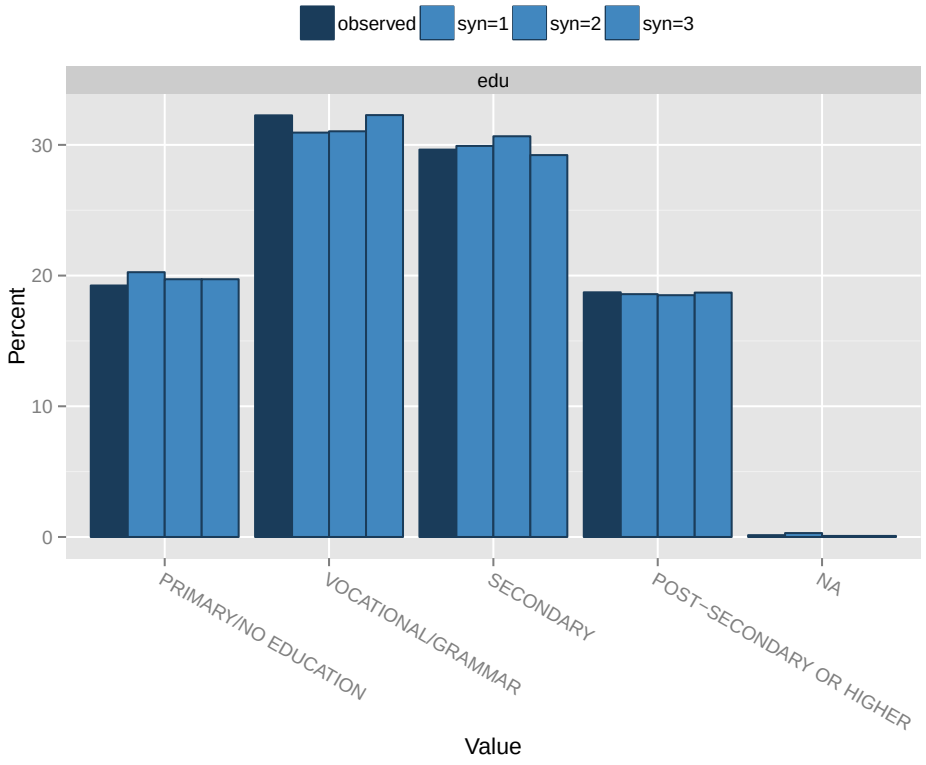

Figure 2: Relative frequency distribution of education variable (`edu`) for observed and synthetic data.

```
> fsds <- glm.synds(I(smoke=="YES") ~ sex + agegr + placesize + edu +
   socprof + marital + income, data = sds, family = "binomial")
> summary(fsds)
> compare(fsds, ods)
```

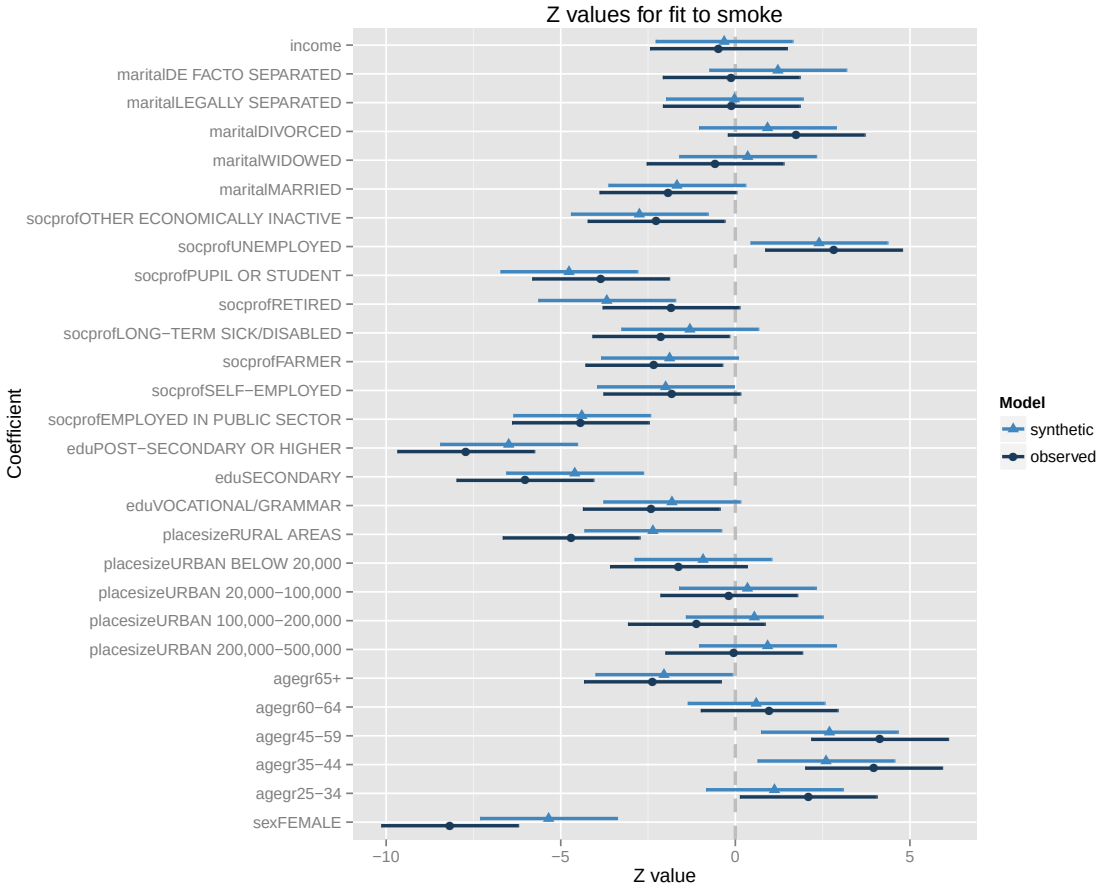

Figure 3: Estimates and 95% confidence intervals for Z statistics from a logistic regression of smoking for observed and synthetic data.

Next to narrow measures of data utility based on comparison of model estimates a user can also use general utility measures implemented in a function `utility.synds()`. They are based on propensity scores and summarize distributional comparison of synthesised data set with the original observed data set. Note that this function is still under development.

```
> utility.synds(sds, ods)
```

3.3 Enhanced data protection

Prior to releasing synthetic data sets, additional post-processing can be conducted via `sdc()` function (`sdc` stands for statistical disclosure control) for enhanced protection of the information confidentiality. The function allows top and bottom coding, adding labels to the synthetic data sets to make it clear that the data are fake so no one mistakenly believe them to be real and removing from the synthetic data set any unique cases with variable sequences that are identical to unique individuals in the real dataset. The last tool reduces the chances of a person who is in the real data believing that their actual data is in the synthetic data. The following command adds a column with a flag `"false_data"`, remove replicated unique units if they exist and applies top and bottom coding for variable `income` but excludes missing data codes from recoding process.

```
> sdssdc <- sdc(sds, ods, label = "false_data",
   rm.replicated.uniques = TRUE, recode.vars = "income",
   bottom.top.coding = c(200, 5000), recode.exclude = -8)
```

3.4 Data import and export

The *synthpop* package offers also auxiliary functions that facilitate data import (`read.obs()`) and export (`write.syn()`). One of the main aims of these functions is to preserve correspondence between numeric codes and labels used in other statistical packages so that the same script can be used for both original and synthetic data. Note that this functionality is available in *synthpop 1.2-0* for SPSS files only. When saving synthesised data sets, information about the synthesis is written into a separate text file with `"synthesis_info_"` prefix. If multiple synthetic data sets are produced they are saved into separate files of selected format. In addition, if `save.complete` parameter is set to `TRUE`, a complete synthesised data set object is saved into `"synobject.rda"` file. In order to save the synthetic data sets from the `sdssdc` object into **R** data files the following command can be used

```
> write.syn(sdssdc, filename = "sdssdc", filetype = "rda")
```

To load an **R** data file with the first synthetic data set, function `load()` should be used, as illustrated below

```
> load("sdssdc1.rda")
```

4 Concluding remarks

In this paper we presented the **R** package *synthpop* for generating synthetic versions of sensitive microdata for statistical disclosure control. Interested readers can consult the package documentation for more detailed information and further examples. We very much welcome users' comments on the *synthpop* package, especially on how to improve and extend its functionality.

References

Nowok, B., Raab, G.M. and Dibben, C. (2015) *synthpop: Bespoke creation of synthetic data in **R***. Package vignette, `http://cran.r-project.org/web/packages/synthpop/vignettes/synthpop.pdf`.

R Core Team (2015) ***R**: A language and environment for statistical computing*, **R** Foundation for Statistical Computing, Vienna, Austria, `http://www.R-project.org`.

Reiter, J.P. (2005) Using CART to generate partially synthetic, public use microdata. *Journal of Official Statistics*, **21**, 441–462.

Rubin, D.B. (1993) Discussion: Statistical disclosure limitation. *Journal of Official Statistics*, **9(2)**, 461–468.

A Appendix - A list of variables

Variable name	*Description*
`sex`	Sex
`agegr`	Age group
`placesize`	Category of the place of residence
`edu`	Highest educational qualification
`socprof`	Socio-economic status
`marital`	Marital status
`income`	Personal monthly net income
`smoke`	Smoking cigarettes