Simulation of EU-SILC Population Data: Using the R Package simPopulation

Andreas Alfons
Erasmus Universiteit Rotterdam

Matthias Templ
Vienna University of Technology, Statistics Austria

Peter Filzmoser
Vienna University of Technology

Abstract

This vignette demonstrates the use of **simPopulation** for simulating population data in an application to the EU-SILC example data from the package. It presents a wrapper function tailored specifically towards EU-SILC data for convenience and ease of use, as well as detailed instructions for performing each of the four involved data generation steps separately. In addition, the generation of diagnostic plots for the simulated population data is illustrated.

Keywords: R, synthetic data, simulation, survey statistics, EU-SILC.

1. Introduction

This package vignette is a companion to Alfons, Kraft, Templ, and Filzmoser (2011) that shows how the proposed framework for the simulation of population data can be applied in R (R Development Core Team 2010) using the package **simPopulation** (Alfons and Kraft 2013). The data simulation framework consists of four steps:

1. Setup of the household structure
2. Simulation of categorical variables
3. Simulation of (semi-)continuous variables
4. Splitting (semi-)continuous variables into components

Note that this vignette does not motivate, describe or evaluate the statistical methodology of the framework. Instead it is focused on the R code to generate synthetic population data and produce diagnostic plots. For details on the statistical methodology, the reader is referred to Alfons *et al.* (2011).

The *European Union Statistics on Income and Living Conditions* (EU-SILC) is panel survey conducted in European countries and serves as data basis for the estimation social inclusion indicators in Europe. EU-SILC data are highly complex and contain detailed information on the income of the sampled individuals and households. More information on EU-SILC can be found in Eurostat (2004).

In Alfons *et al.* (2011), three methods for the simulation of the net income of the individuals in the population are proposed and analyzed:

MP Multinomial logistic regression models with random draws from the resulting categories. For the categories corresponding to the upper tail, the values are drawn from a (truncated) generalized Pareto distribution, for the other categories from a uniform distribution.

TR Two-step regression models with trimming and random draws from the residuals.

TN Two-step regression models with trimming and random draws from a normal distribution.

The first two steps of the analysis, namely the simulation of the household structure and additional categorical variables, are performed in exactly the same manner for the three scenarios. While the simulation of the income components is carried out with the same parameter settings, the results of course depend on the simulated net income.

It is important to note that the original Austrian EU-SILC sample provided by Statistics Austria and used in Alfons *et al.* (2011) is confidential, hence the results presented there cannot be reproduced in this vignette. Nevertheless, the code for such an analysis is presented here using the example data from the package, which has been synthetically generated itself. In fact, this example data set is a sample drawn from one of the populations generated in Alfons *et al.* (2011). However, the sample weights have been modified such that the size of the resulting populations is about 1% of the real Austrian population in order to keep the computation time low. Table 1 lists the variables of the example data used in the code examples.

With the following commands, the package and the example data are loaded. Furthermore, the numeric value stored in `seed` will be used as seed for the random number generator in the examples to make the results reproducible.

```
R> library("simPopulation")
R> data("eusilcS")
R> seed <- 1234
```

The rest of this vignette is organized as follows. Section 2 illustrates the use of a convenient wrapper function for the generation of EU-SILC population data. In Section 3, detailed instructions are given for each step in the data generation process as well as for the generation of diagnostic plots. The final Section 4 concludes.

2. Wrapper function for EU-SILC

A convenient way of generating synthetic EU-SILC population data is provided by the wrapper function `simEUSILC()`, which performs the four steps of the data simulation procedure at once. For each step, the names of the variables to be simulated can be supplied. However, the default values for the respective arguments are given by the variables names used in Alfons *et al.* (2011). Since the same names are used in the example data, the complex procedures for the three different methods can be carried out with very simple commands.

Table 1: Variables of the EU-SILC example data in **simPopulation**.

Variable	Name	Type	
Region	`db040`	Categorical	9 levels
Household size	`hsize`	Categorical	9 levels
Age	`age`	Categorical	
Gender	`rb090`	Categorical	2 levels
Economic status	`pl030`	Categorical	7 levels
Citizenship	`pb220a`	Categorical	3 levels
Personal net income	`netIncome`	Semi-continuous	
Employee cash or near cash income	`py010n`	Semi-continuous	
Cash benefits or losses from self-employment	`py050n`	Semi-continuous	
Unemployment benefits	`py090n`	Semi-continuous	
Old-age benefits	`py100n`	Semi-continuous	
Survivor's benefits	`py110n`	Semi-continuous	
Sickness benefits	`py120n`	Semi-continuous	
Disability benefits	`py130n`	Semi-continuous	
Education-related allowances	`py140n`	Semi-continuous	
Household sample weights	`db090`	Continuous	
Personal sample weights	`rb050`	Continuous	

```
R> eusilcMP <- simEUSILC(eusilcS, upper = 200000, equidist = FALSE,
+     seed = seed)
R> eusilcTR <- simEUSILC(eusilcS, method = "twostep", seed = seed)
R> eusilcTN <- simEUSILC(eusilcS, method = "twostep", residuals = FALSE,
+     seed = seed)
```

Note that the default is to use the MP procedure. An upper bound for the net income is supplied using the argument `upper`, while the argument `equidist` is set to `FALSE` so that the breakpoints for the discretization of the net income are given by quantiles with non-equidistant probabilities as described in Alfons *et al.* (2011). The twostep regression approaches are performed by setting `method = "twostep"`, in which case the logical argument `residuals` specifies whether variability should be added by random draws from the residuals (TR method, the default) or from a normal distribution (TN method). In both cases, the default trimming parameter `alpha = 0.01` is used.

The synthetic populations generated with the wrapper function are not further evaluated here, instead a detailed illustration of each step along with diagnostic plots is provided in the following section.

3. Step by step instructions and diagnostics

As for the wrapper function `simEUSILC()`, the variable names of the example data set are used as default values for the corresponding arguments of the functions for the different steps of the procedure. Nevertheless, in order to demonstrate how these arguments are used, the names of the involved variables are always supplied in the commands shown in this section.

The first step of the analysis is to set up the basic household structure using the function `simStructure()`. Note that a variable named `"hsize"` giving the household sizes is generated automatically in this example, but the name of the corresponding variable in the sample data can also be specified as an argument. Furthermore, the argument `additional` specifies the variables that define the household structure in addition to the household size (in this case age and gender).

```
R> eusilcP <- simStructure(eusilcS, hid = "db030", w = "db090",
+     strata = "db040", additional = c("age", "rb090"))
```

For the rest of the procedure, combined age categories are used for the individuals in order to reduce the computation time of the statistical models.

```
R> breaks <- c(min(eusilcS$age), seq(15, 80, 5), max(eusilcS$age))
R> eusilcS$ageCat <- as.character(cut(eusilcS$age,
+         breaks=breaks, include.lowest=TRUE))
R> eusilcP$ageCat <- as.character(cut(eusilcP$age,
+         breaks=breaks, include.lowest=TRUE))
```

Additional categorical variables are then simulated using the function `simCategorical()`. The argument `basic` thereby specifies the already generated variables for the basic household structure (age category, gender and household size), while `additional` specifies the variables to be simulated in this step (economic status and citizenship).

```
R> basic <- c("ageCat", "rb090", "hsize")
R> eusilcP <- simCategorical(eusilcS, eusilcP, w = "rb050", strata = "db040",
+     basic = basic, additional = c("pl030", "pb220a"))
```

Mosaic plots are available as graphical diagnostic tools for checking whether the structures of categorical variables are reflected in the synthetic population. They are implemented in the function `spMosaic()` based on the package **vcd** (Meyer, Zeileis, and Hornik 2006, 2010), which contains extensive functionality for customization.

With the following commands, mosaic plots for the variables gender, region and household size are created (see Figure 1, *top*). The function `labeling_border()` from package **vcd** is thereby used to set shorter labels for the different regions and to display more meaningful labels for the variables.

```
R> abb <- c("B", "LA", "Vi", "C", "St", "UA", "Sa", "T", "Vo")
R> nam <- c(rb090 = "Gender", db040 = "Region", hsize = "Household size")
R> lab <- labeling_border(set_labels = list(db040 = abb),
+     set_varnames = nam)
R> spMosaic(c("rb090", "db040", "hsize"), "rb050", eusilcS,
+     eusilcP, labeling = lab)
```

In addition, mosaic plots for the variables gender, economic status and citizenship are produced (see Figure 1, *bottom*). Also in this case, `labeling_border()` is used for some fine tuning. In particular, the categories of citizenship are abbreviated and again more meaningful labels for the variables are set.

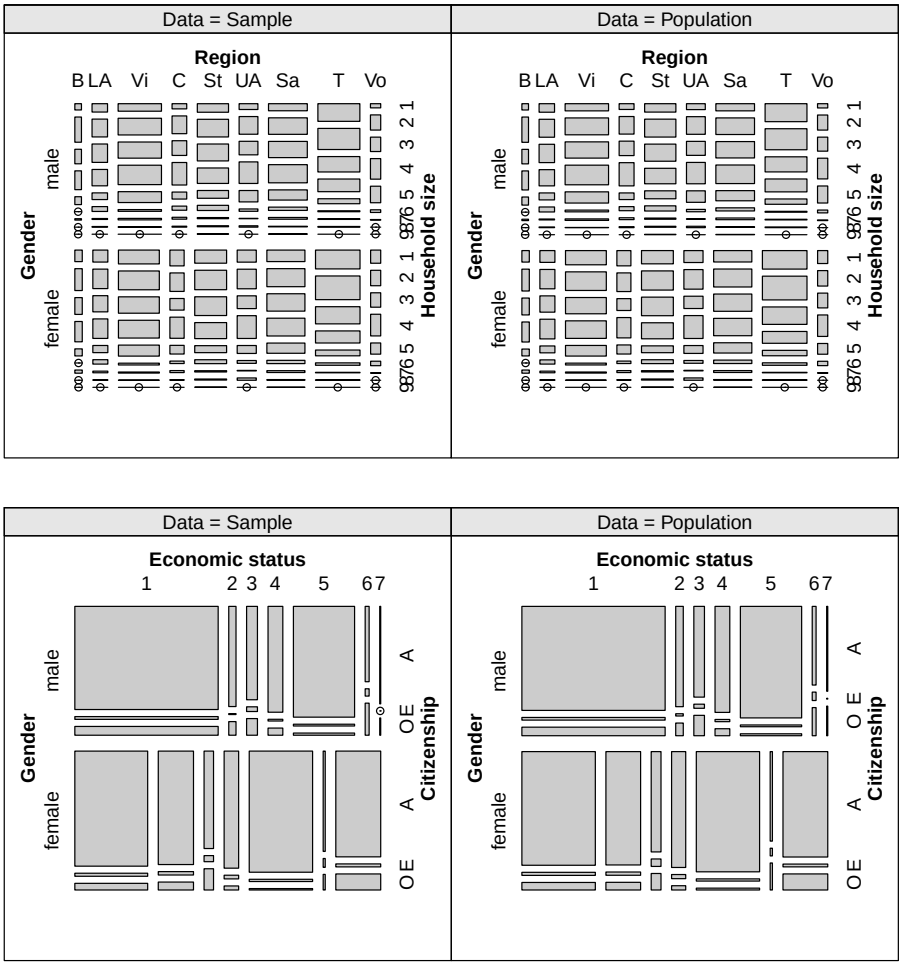

Figure 1: *Top*: Mosaic plots of gender, region and household size. *Bottom*: Mosaic plots of gender, economic status and citizenship.

```
R> nam <- c(rb090 = "Gender", pl030 = "Economic status",
+     pb220a = "Citizenship")
R> lab <- labeling_border(abbreviate = c(FALSE, FALSE, TRUE),
+     set_varnames = nam)
R> spMosaic(c("rb090", "pl030", "pb220a"), "rb050", eusilcS,
+     eusilcP, labeling = lab)
```

Next, the function `simContinuous()` is used to simulate the net income according to the three proposed methods. The same parameter settings as in Section 2 are thereby used for each of the methods. In any case, the argument `basic` specifies the predictor variables (age category, gender, household size, economic status and citizenship), while the argument `additional` specifies the variable to be simulated.

Note that the current state of the random number generator is stored beforehand so that the different methods can all be started with the same seed. Furthermore, the random seed after

each of the methods has finished is stored so that the simulation of the income components can later on continue from there.

```
R> seedP <- .Random.seed
R> basic <- c(basic, "pl030", "pb220a")
R> eusilcMP <- simContinuous(eusilcS, eusilcP, w = "rb050",
+     strata = "db040", basic = basic, additional = "netIncome",
+     upper = 200000, equidist = FALSE, seed=seedP)
R> seedMP <- .Random.seed
R> eusilcTR <- simContinuous(eusilcS, eusilcP, w = "rb050",
+     strata = "db040", basic = basic, additional = "netIncome",
+     method="lm", seed=seedP)
R> seedTR <- .Random.seed
R> eusilcTN <- simContinuous(eusilcS, eusilcP, w = "rb050",
+     strata = "db040", basic = basic, additional = "netIncome",
+     method="lm", residuals=FALSE, seed=seedP)
R> seedTN <- .Random.seed
```

Two functions are available as diagnostic tools for (semi-)continuous variables: `spCdfplot()` for comparing the cumlative distribution functions, and `spBwplot()` for comparisons with box-and-whisker plots. Both are implemented based on the package **lattice** (Sarkar 2008, 2010).

The following commands are used to produce the two plots in Figure 2. For better visibility of the differences in the main parts of the cumulative distribution functions, only the parts between 0 and the weighted 99% quantile of the sample are plotted (see Figure 2, *left*). Furthermore, the box-and-whisker plots by default do not display any points outside the extremes of the whiskers (see Figure 2, *right*). This is because population data are typically very large, which almost always would result in a large number of observations ouside the whiskers. Also note that a list containing the three populations is supplied as the argument `dataP` of the plot functions.

```
R> subset <- which(eusilcS[, "netIncome"] > 0)
R> q <- quantileWt(eusilcS[subset, "netIncome"],
+     eusilcS[subset, "rb050"], probs = 0.99)
R> listP <- list(MP=eusilcMP, TR=eusilcTR, TN=eusilcTN)
R> spCdfplot("netIncome", "rb050", dataS=eusilcS, dataP=listP, xlim=c(0, q))
R> spBwplot("netIncome", "rb050", dataS=eusilcS, dataP=listP, pch="|")
```

One of the main requirements in the simulation of population data is that heterogeneities between subgroups are reflected (see Alfons *et al.* 2011). Since `spCdfplot()` and `spBwplot()` are based on **lattice**, this can easily be checked by producing conditional plots. With the following commands, the box-and-whisker plots in Figure 3 are produced. The conditioning variables gender (*top left*), citizenship (*top right*), region (*bottom left*) and economic status (*bottom right*) are thereby used. For finetuning, the layout of the panels is specified with the `layout` argument provided by the **lattice** framework.

```
R> spBwplot("netIncome", "rb050", "rb090", dataS=eusilcS,
+     dataP=listP, pch="|", layout=c(1,2))
```

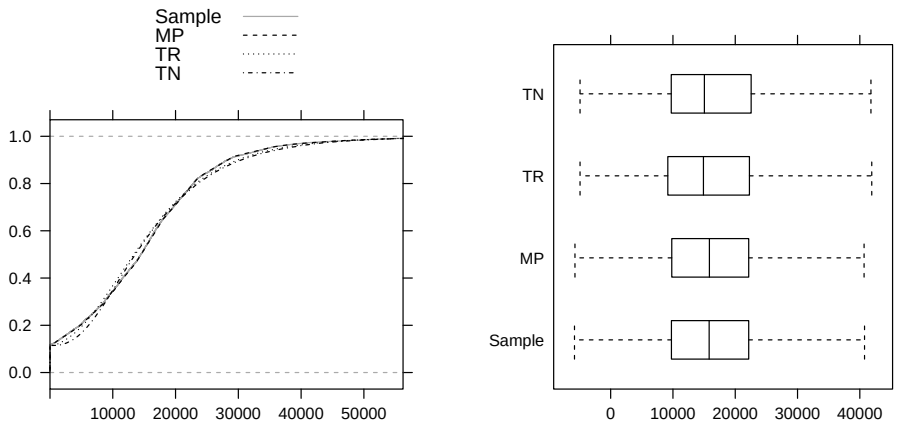

Figure 2: *Left*: Cumulative distribution functions of personal net income. For better visibility, the plot shows only the main parts of the data. *Right:* Box plots of personal net income. Points outside the extremes of the whiskers are not plotted.

```
R> spBwplot("netIncome", "rb050", "pb220a", dataS=eusilcS,
+     dataP=listP, pch="|", layout=c(1,3))
R> spBwplot("netIncome", "rb050", "db040", dataS=eusilcS,
+     dataP=listP, pch="|", layout=c(1,9))
R> spBwplot("netIncome", "rb050", "pl030", dataS=eusilcS,
+     dataP=listP, pch="|", layout=c(1,7))
```

The last step of the analysis is to simulate the income components. This is done based on resampling of fractions conditional on net income category and economic status. Therefore, the net income categories need to be constructed first. With the function `getBreaks()`, default breakpoints based on quantiles are computed. In this example, the argument `upper` is set to `Inf` to avoid problems with different maximum values in the three synthetic populations, and the argument `equidist` is set to `FALSE` such that non-equidistant probabilities as described in Alfons *et al.* (2011) are used for the calculation of the quantiles.

```
R> breaks <- getBreaks(eusilcS$netIncome, eusilcS$rb050,
+     upper = Inf, equidist = FALSE)
R> eusilcS$netIncomeCat <- getCat(eusilcS$netIncome, breaks)
R> eusilcMP$netIncomeCat <- getCat(eusilcMP$netIncome, breaks)
R> eusilcTR$netIncomeCat <- getCat(eusilcTR$netIncome, breaks)
R> eusilcTN$netIncomeCat <- getCat(eusilcTN$netIncome, breaks)
```

Once the net income categories are constructed, the income components are simulated using the function `simComponents()`. The arguments `total`, `components` and `conditional` thereby specify the variable to be split, the variables containing the components, and the conditioning variables, respectively. In addition, for each of the three populations the seed of the random number generator is set to the corresponding state after the simulation of the net income.

```
R> components <- c("py010n", "py050n", "py090n",
```

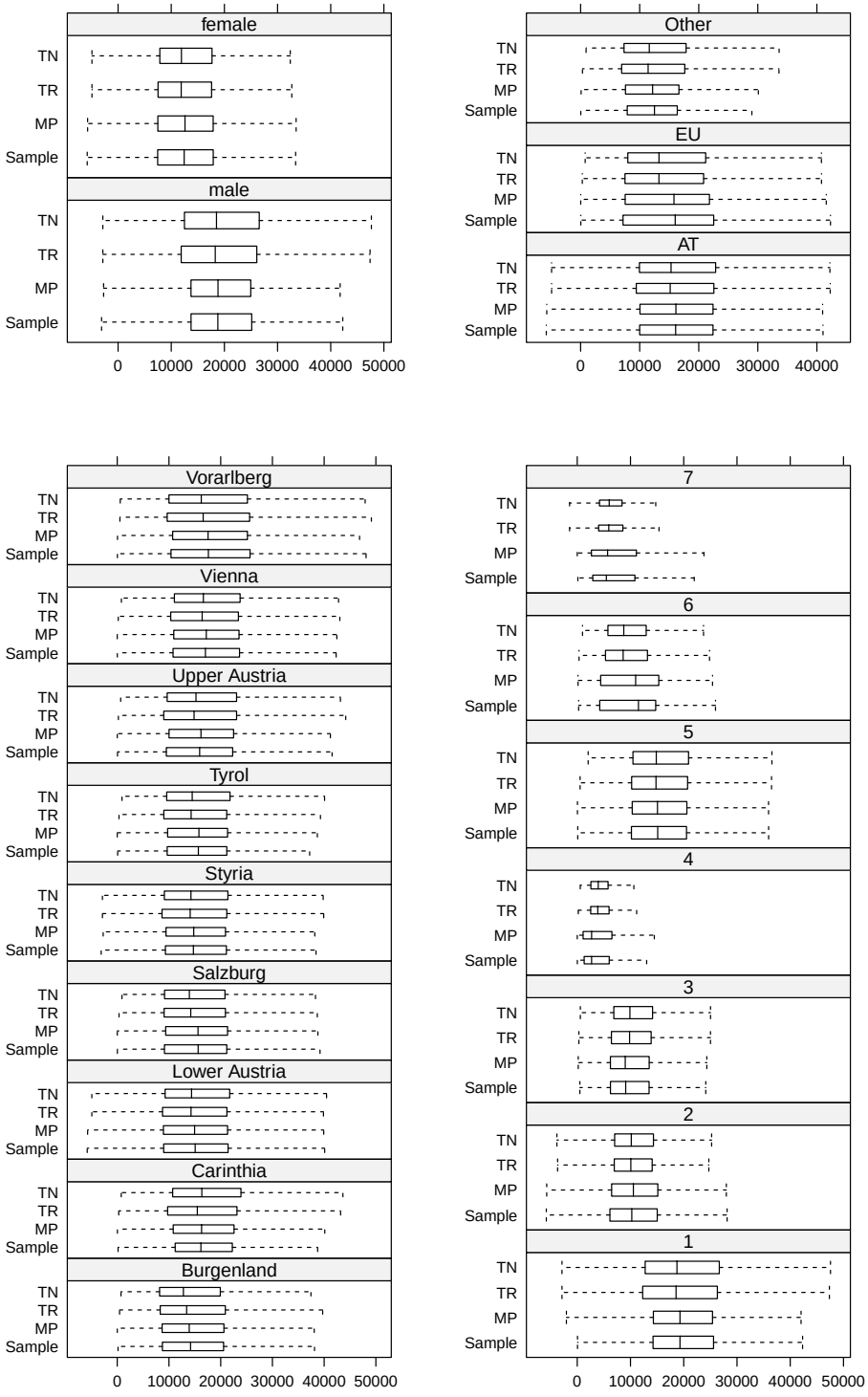

Figure 3: Box plots of personal net income split by gender (*top left*), citizenship (*top right*), region (*bottom left*) and economic status (*bottom right*). Points outside the extremes of the whiskers are not plotted.

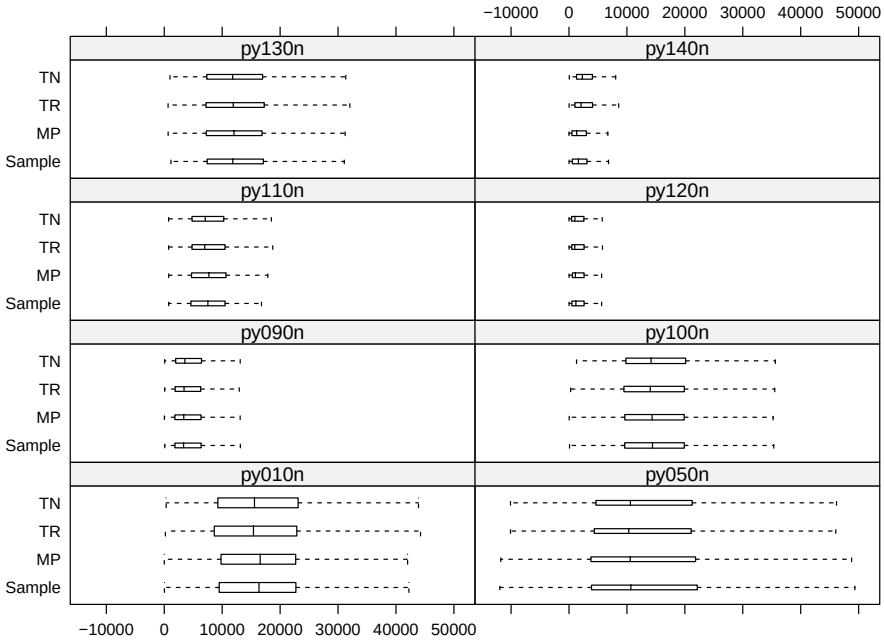

Figure 4: Box plots of the income components. Points outside the extremes of the whiskers are not plotted.

```
+         "py100n", "py110n", "py120n", "py130n", "py140n")
R> eusilcMP <- simComponents(eusilcS, eusilcMP, w = "rb050",
+     total = "netIncome", components = components,
+     conditional = c("netIncomeCat", "pl030"), seed = seedMP)
R> eusilcTR <- simComponents(eusilcS, eusilcTR, w = "rb050",
+     total = "netIncome", components = components,
+     conditional = c("netIncomeCat", "pl030"), seed=seedTR)
R> eusilcTN <- simComponents(eusilcS, eusilcTN, w = "rb050",
+     total = "netIncome", components = components,
+     conditional = c("netIncomeCat", "pl030"), seed=seedTN)
```

Finally, diagnostic box-and-whisker plots of the income components are produced with the function `spBwplot()`. Since the box widths correspond to the ratio of non-zero observations to the total number of observed values and most of the components contain large proportions of zeros, a minimum box width is specified using the argument `minRatio`. Figure 4 contains the resulting plots.

```
R> listP <- list(MP=eusilcMP, TR=eusilcTR, TN=eusilcTN)
R> spBwplot(components, "rb050", dataS=eusilcS,
+     dataP=listP, pch="|", minRatio=0.2, layout=c(2,4))
```

4. Conclusions

In this vignette, the use of **simPopulation** for simulating population data has been demonstrated in an application to the EU-SILC example data from the package. Both the simulation of synthetic population data and the generation of diagnostic plots have been illustrated in a similar analysis as in Alfons *et al.* (2011).

The code examples show that the functions are easy to use and that the arguments have sensible default values. Nevertheless, the behavior of the functions is highly customizable. In particular the functions for the diagnostic plots benefit from the implementations based on the packages **vcd** and **lattice**.

Acknowledgments

This work was partly funded by the European Union (represented by the European Commission) within the 7th framework programme for research (Theme 8, Socio-Economic Sciences and Humanities, Project AMELI (Advanced Methodology for European Laeken Indicators), Grant Agreement No. 217322). Visit http://ameli.surveystatistics.net for more information on the project.

References

Alfons A, Kraft S (2013). ***simPopulation**: Simulation of Synthetic Populations for Surveys based on Sample Data*. R package version 0.4.1, URL http://CRAN.R-project.org/package=simPopulation.

Alfons A, Kraft S, Templ M, Filzmoser P (2011). "Simulation of close-to-reality population data for household surveys with application to EU-SILC." *Statistical Methods & Applications*, **20**(3), 383–407.

Eurostat (2004). "Description of Target Variables: Cross-sectional and Longitudinal." *EU-SILC 065/04*, Eurostat, Luxembourg.

Meyer D, Zeileis A, Hornik K (2006). "The `strucplot` Framework: Visualizing Multi-way Contingency Tables with **vcd**." *Journal of Statistical Software*, **17**(3), 1–48.

Meyer D, Zeileis A, Hornik K (2010). ***vcd**: Visualizing Categorical Data*. R package version 1.2-9, URL http://CRAN.R-project.org/package=vcd.

R Development Core Team (2010). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Sarkar D (2008). *Lattice: Multivariate Data Visualization with R*. Springer, New York. ISBN 978-0-387-75968-5.

Sarkar D (2010). ***lattice**: Lattice Graphics*. R package version 0.19-13, URL http://CRAN.R-project.org/package=lattice.

Affiliation:

Andreas Alfons
Erasmus School of Economics
Erasmus Universiteit Rotterdam
PO Box 1738
3000DR Rotterdam, Netherlands
E-mail: alfons@ese.eur.nl
URL: http://people.few.eur.nl/alfons/