Producing synthetic and confidential data using R-Package simPop

Bernhard Meindl and Matthias Templ (STAT, TU Vienna)
DwB-Workshop, March, 2015

Why synthetic populations?

- methods comparison (e.g design-based simulation studies)
- policy modelling on individual level (e.g health planning, climate change, demographic change, economic change, …)
- teaching (e.g. teaching of survey methods)
- Creation of public-/scientific-use files with low disclosure risk
- data availability is often a problem (legal issues, costs,…)

Close-to-reality data

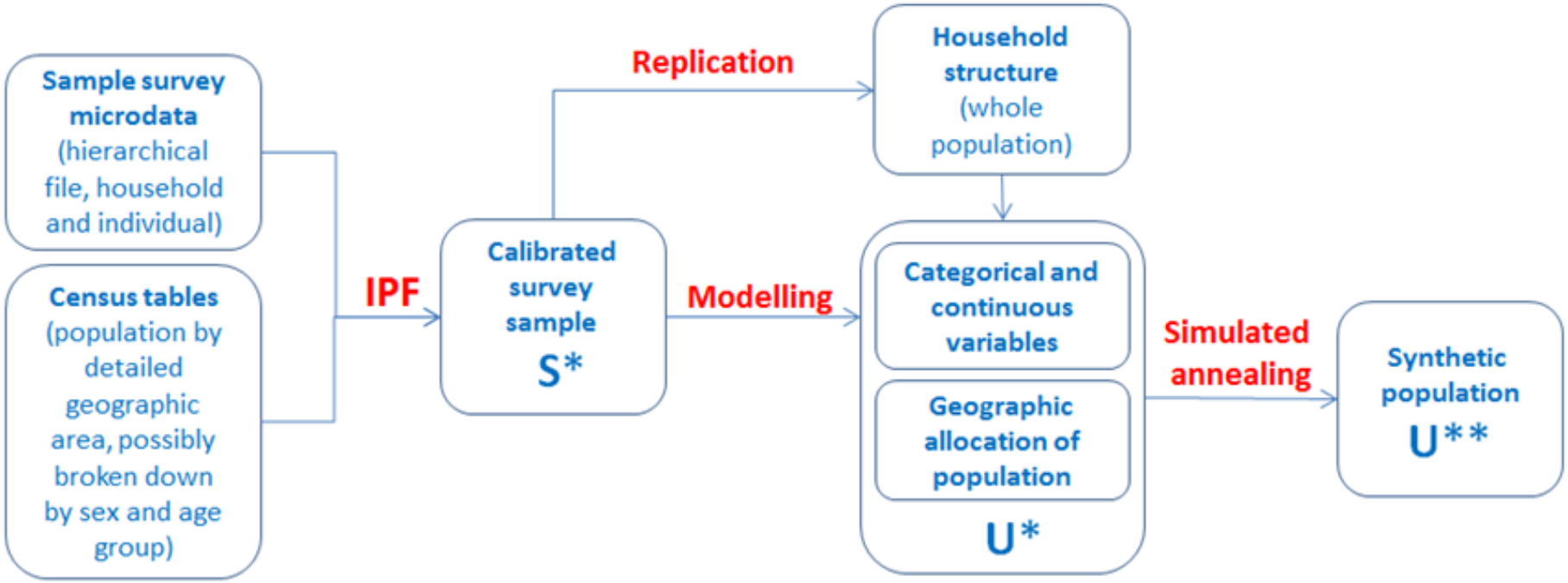
- actual sizes of regions and strata need to be reflected
- marginal distributions and interactions between variables should be represented correctly
- hierarchical and cluster structures have to be preserved
- Data confidentiality must be ensured
- Pure replication of units from the underlying sample should be avoided
- Sometimes some marginal distributions must exactly match known values

Approaches

- choice of methods highly depends on available information
  - survey samples
  - aggregated information from samples
  - known marginal distributions from population
- in **simPop**: model-based approach is forced

Model-based approach (1)

Possible flow, in reality often (much) more complex



Model-based approach (2)

- In general, the procedure consists of four steps:
  - setup of the household structure
  - simulation of categorical variables
  - simulation of continuous variables
  - the splitting continuous variables into components
- Stratification: allows to account for heterogenities (e.g. regional differences)
- Sampling weights are considered in each step

Model-based approach (3)

- Household structure (core-variables): simulated separately for each combination of strata and household size.
- # of households: estimated using the HT-estimator
- As few variables as possible (due to confidentiality reasons) are simulated using a sampling approach
- This builds up a realistic structure of the core variables
- Finally, additional variables are simulated using a regression-based approach using all existing variables.

R package simPop

- Developed with support of the
  - International Household Survey Network
  - Department for International Development (DFID)
- latest version on **CRAN**
- highly object-oriented approach, similar to sdcMicro
- lets you produce synthetic confidential data
- efficiently programmed to work for (very) large data sets
- parallel computing is automatically be applied

Example: EU-SILC

```
library(simPop)
str(origData)
```

```
'data.frame':   11725 obs. of  18 variables:
 $ db030    : int  1 1 2 3 4 4 4 5 5 5 ...
 $ hsize    : int  2 2 1 1 3 3 3 5 5 5 ...
 $ db040    : Factor w/ 9 levels "Burgenland","Carinthia",..: 4 4 7 5 7 7 7 4 4 4 ...
 $ age      : int  72 66 56 67 70 46 37 41 35 9 ...
 $ rb090    : Factor w/ 2 levels "male","female": 1 2 2 2 2 1 1 1 2 2 ...
 $ pl030    : Factor w/ 7 levels "1","2","3","4",..: 5 5 2 5 5 3 1 1 3 NA ...
 $ pb220a   : Factor w/ 3 levels "AT","EU","Other": 1 1 1 1 1 1 3 1 1 NA ...
 $ netIncome: num  22675 16999 19274 13319 14366 ...
 $ py010n   : num  0 0 19274 0 0 ...
 $ py050n   : num  0 0 0 0 0 ...
 $ py090n   : num  0 0 0 0 0 ...
 $ py100n   : num  22675 0 0 13319 14366 ...
 $ py110n   : num  0 0 0 0 0 0 0 0 0 NA ...
 $ py120n   : num  0 0 0 0 0 0 0 0 0 NA ...
 $ py130n   : num  0 16999 0 0 0 ...
 $ py140n   : num  0 0 0 0 0 0 0 0 0 NA ...
 $ db090    : num  7.82 7.82 8.79 8.11 7.51 ...
 $ rb050    : num  7.82 7.82 8.79 8.11 7.51 ...
```

Structure your data (once to be defined)

Create an object of class *dataObj* with function **specifyInput()**.

```
inp <- specifyInput(data=origData, hhid="db030", hhsize="hsize", strata="db040", weight="rb050"); inp
```

```
 --------------
survey sample of size 11725 x 19

 Selected important variables:

 household ID: db030
 personal ID: pid
 variable household size: hsize
 sampling weight: rb050
 strata: db040
 --------------
```

Additional information

- (external) Population charateristics on EU-SILC variables can be specified as
  - data frame or
  - n-dimensional table

```
data(totalsRGtab); totalsRGtab # here: 2-dimensional table
```

```
        db040
rb090    Burgenland Carinthia Lower Austria Salzburg Styria  Tyrol
  female     146980    285797        828087   722883 274675 619404
  male       140436    270084        797398   702539 259595 595842
        db040
rb090    Upper Austria Vienna Vorarlberg
  female        368128 916150     190343
  male          353910 850596     184939
```

- Calibration to this given known totals:

```
addWeights(inp) <- calibSample(inp, totalsRGtab)
```

Simulating the basic structure

```
synthP <- simStructure(data=inp, method="direct", basicHHvars=c("age", "rb090", "db040"))
class(synthP)
```

```
[1] "simPopObj"
attr(,"package")
[1] "simPop"
```

- Output object (“*synthP*”) is of class *simPopObj*
- various functions can be directly applied such objects

Simulation of categorical variables

```
synthP <- simCategorical(synthP, additional = c("pl030","pb220a"), method = "multinom")
synthP
```

```
--------------
synthetic population  of size
 85057 x 10

build from a sample of size
11725 x 19
--------------

variables in the population:
db030,hsize,db040,age,rb090,db040.1,pid,weight,pl030,pb220a
```

Simulating continuous variables

```
# multinomial model with random draws
synthP <- simContinuous(synthP, additional = "netIncome", upper = 2e+05, equidist = FALSE)
```

```
--------------
synthetic population  of size
 85057 x 12

build from a sample of size
11725 x 19
--------------

variables in the population:
db030,hsize,db040,age,rb090,db040.1,pid,weight,pl030,pb220a,netIncomeCat,netIncome
```

- Simulation of components with **simComponents()**
- Simulation of finer regional variables (like districts) with **simInitSpatial()**

Census information to calibrate

- Assumption: external information (n-dimensional table) is available, e.g marginals on *region* $\times$ *gender* $\times$ *exonomic status*.
- We add these marginals to the object and calibrate afterwards

```
synthP <- addKnownMargins(synthP, margins) # add margins
```

```
# calibration using simulated annealing
synthPadj <- calibPop(synthP, split="db040", temp=1, eps.factor=0.00005, maxiter=200,
temp.cooldown=0.975, factor.cooldown=0.85, min.temp=0.001, verbose=FALSE)
```

Continued, we end up with

```
synthPadj
```

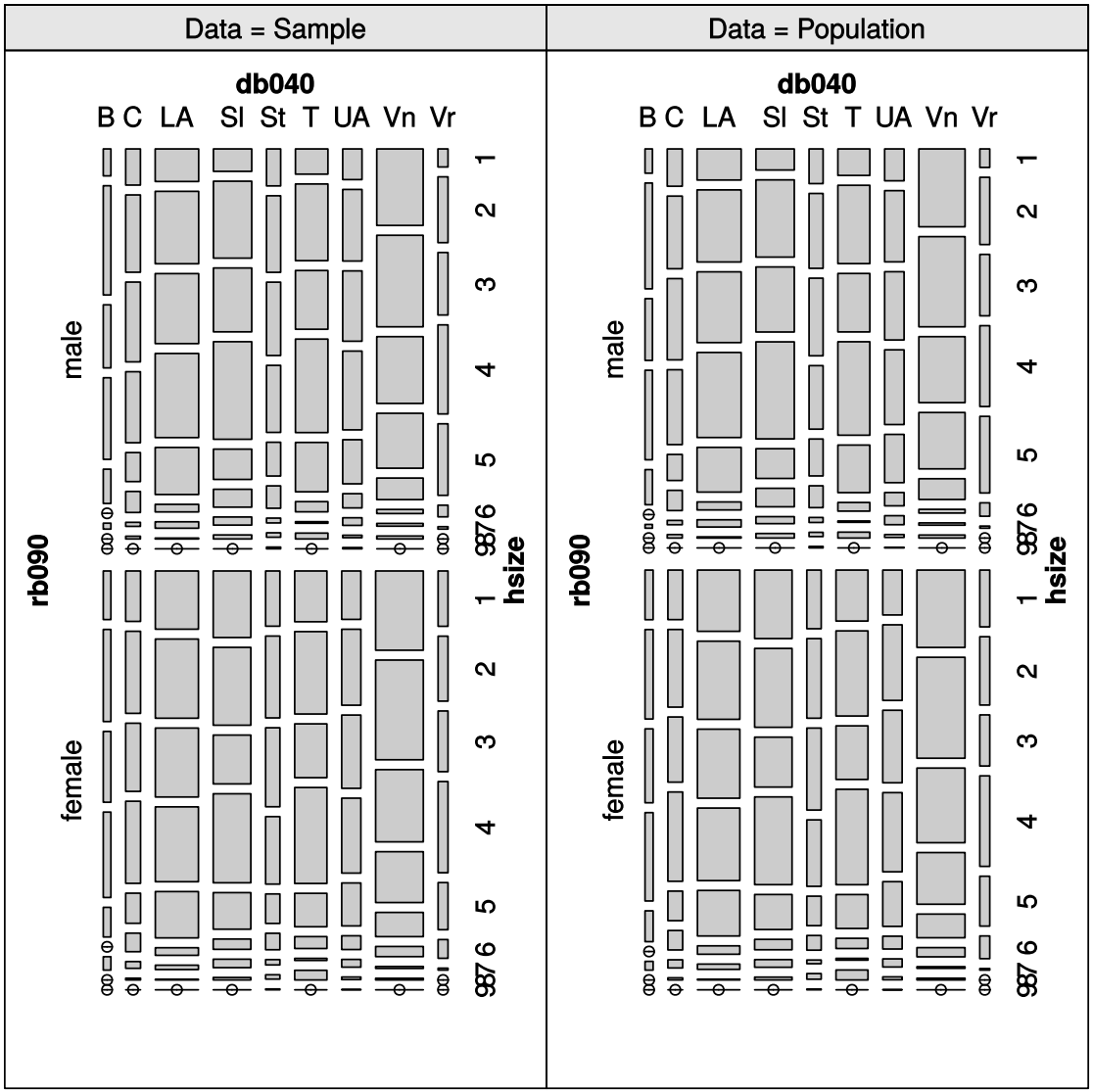
```
--------------
synthetic population  of size
 84975 x 12

build from a sample of size
11725 x 19
--------------

variables in the population:
db030,hsize,db040,age,rb090,db040.1,pid,weight,pl030,pb220a,netIncomeCat,netIncome
```

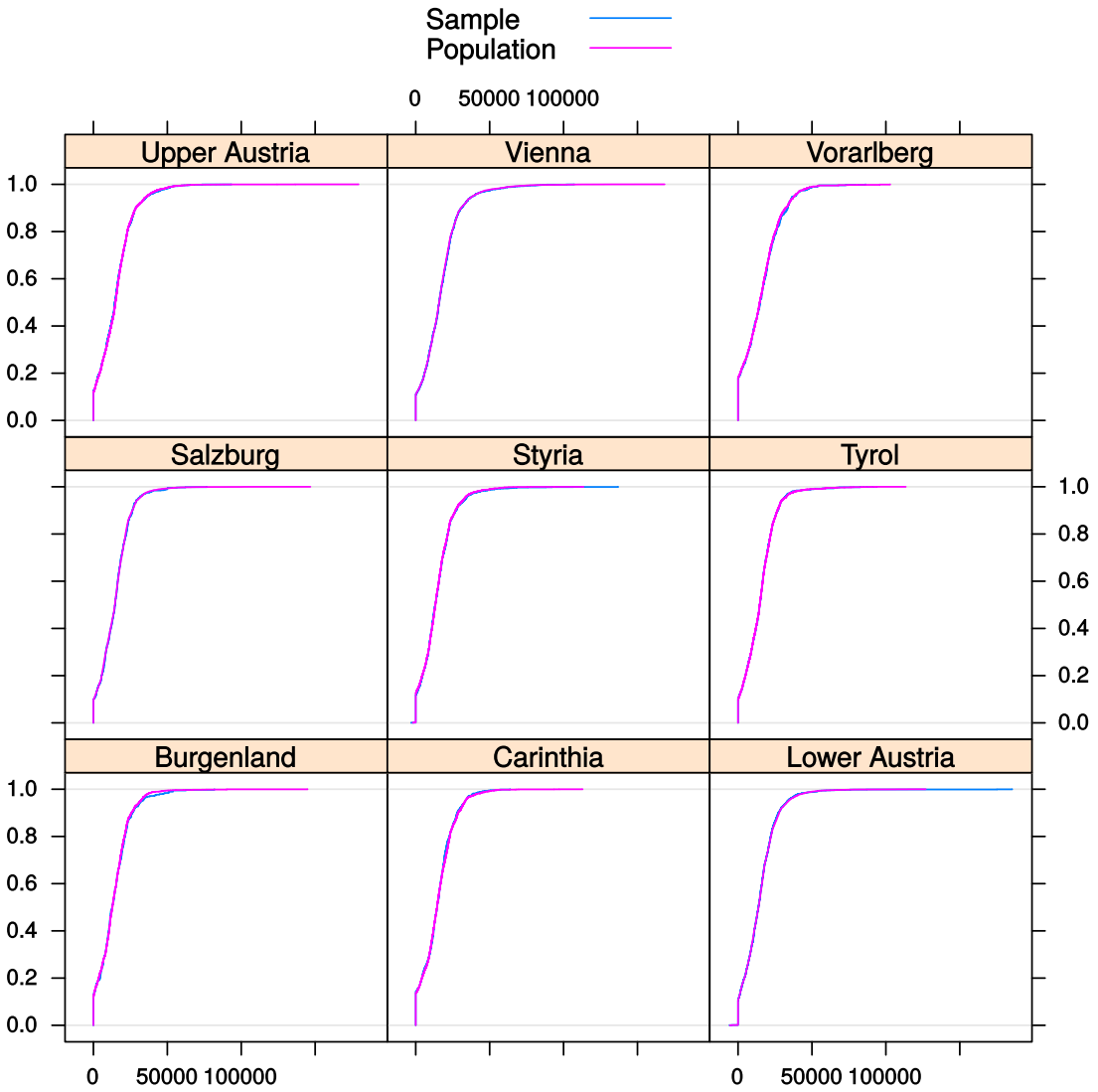
Results

```
tab <- spTable(synthP, select=c("rb090","db040","hsize"))
spMosaic(tab, labeling=labeling_border(abbreviate=c(db040=TRUE)))
```



Results

```
spCdfplot(synthPadj, "netIncome", cond="db040", layout=c(3, 3))
```



Conclusions

- margins of synthetic populations are calibrated
- all statistics can be very precisely estimated
- the synthetic populations are confidential
- code of **simPop** is quite efficient
- many other methods (IPU, IPF) are also included