The R package `simFrame`: An object-oriented approach towards simulation studies in statistics

Andreas Alfons[1] Matthias Templ[1,2] Peter Filzmoser[1]

[1]Department of Statistics and Probability Theory, Vienna University of Technology

[2]Methods Unit, Statistics Austria

The R User Conference, Gaithersburg, July 21, 2010

Content

Motivation

Research projects in statistcs are often based on extensive simulation studies (e.g., project AMELI, see Alfons et al. 2009).

- If many scientists or institutions are involved, a precise outline for the simulation studies is required.
- Otherwise results obtained by different researchers may be incomparable, which in turn makes it impossible to draw meaningful conclusions.

A common framework for simulation may assist statisticians in defining and following such common guidelines. Requirements include:

- running simulations with only a few commands,
- adding contamination to investigate robust methods,
- inserting missing values to evaluate methods for incomplete data,
- visualization of simulation results and
- easy extensibility for special needs.

The R package `simFrame` (Alfons 2010) ...

- is an object-oriented framework for statistical simulation based on S4 classes and methods (Chambers 1998, 2008) focused on applications in robust statistics.
- offers a wide range of simulation designs with a minimal amount of programming.
- gives maximum control over input and output due to the object-oriented implementation, while at the same time providing clear interfaces for extensions by user-defined classes and methods.
- allows certain proportions of the data to be contaminated or set to `NA`.
- selects an appropriate plot method for the simulation results automatically depending on their structure.

Categorization of statistical simulation

- Statistical simulation is frequently divided into the following categories:

  Model-based simulation: data sets are generated repeatedly from a distributional model or a mixture of distributions.

  - Application: Evaluation of methods that make theoretical assuptions about the underlying data.
  - Example: Comparison of outlier detection methods, which typically assume a multivariate normal distribution.

  Design-based simulation: samples are drawn repeatedly from a finite population.

  - Application: Evaluation of methods in survey statistics.
  - Example: Comparison of different estimation methods for point and variance estimates of poverty measures.

- Both types of simulations are supported by `simFrame`.

Design of the framework

- Statistical simulation in R is often done using bespoke use-once-and-throw-away scripts.

  $\longrightarrow$ This approach has its limitations if a research project is based on extensive simulation studies.
- Design of `simFrame`: Simulations are performed using the generic function `runSimulation()`, whose behavior is determined by control objects.
  - The user supplies a function that is applied in every iteration.
  - Switching from one simulation design to another can easily be done by just plugging in different control objects.
  - Only a few lines of code are necessary, the user does not have to worry about loops or collecting the results in a suitable data structure.
  - Comparability of the results from different researchers in a project is ensured if the same control objects are used.

General implementation

- Most of `simFrame` is implemented as S4 classes and methods, except some utility functions and some C code.
- Control classes ...
  - contain all necessary information to perform the corresponding tasks (data generation, sampling, contamination, insertion of missing values).
  - are the basis for method selection for generic functions, which in most cases provides the interfaces for extensions by developers.
- Available package vignettes:
  - `vignette("simFrame-intro")`: detailed discussion of the implementation (Alfons et al. 2010b, submitted to JSS).
  - `vignette("simFrame-eusilc")`: further code examples that show the strengths of the framework.

Simplified UML class diagram

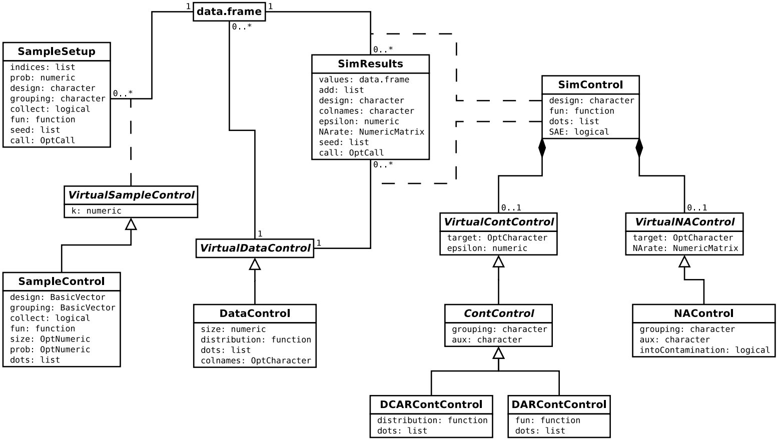


Figure: Simplified UML class diagram of `simFrame`

Running simulations

The generic function `runSimulation()` is the interface for running simulation studies.

- Methods for model-based and design-based simulation are implemented.
- In addition, the control class `SimControl` determines how the simulation runs are performed.
  - A function to be applied in every simulation run needs to be specified.
  - Control objects for contamination and the insertion of missing values may be supplied.

Notes on the function to be applied in each simulation run:

- There are some requirements, which are listed in Alfons et al. (2010b).
- It is evaluated using `try` to prevent loss of results if computations fail in one of the simulation runs.

Visualization

Visualization methods for the simulation results are based on `lattice` graphics (Sarkar 2008, 2010).

`simBwplot()`: boxplots of the simulation results.

`simDensityplot()`: densityplot of the simulation results.

`simXyplot()`: for different contamination levels or missing value rates, the (average) results are plotted against the contamination levels or missing value rates.

The `plot()` method for the simulation results selects a suitable graphical representation automatically.

Parallel computing

Statistical simulation is embarrassingly parallel, hence computational performance can be increased by parallel computing. In `simFrame`, parallel computing is implemented using the R package `snow` (Rossini et al. 2007, Tierney et al. 2008, 2009).

`clusterSetup()`: setting up multiple samples on a cluster.

`clusterRunSimulation()`: running simulations on a cluster.

- All objects and packages required for the computations (including `simFrame`) need to be made available on every worker process.
- In order to ensure reproducibility of the simulation results, random number streams (L'Ecuyer et al. 2002, Sevcikova and Rossini 2009) should be used. These are supported by `snow` via the function `clusterSetupRNG()`.

Example: Gini coefficient I

- EU-SILC is a complex household survey used as basis for measuring risk-of-poverty and social cohesion in Europe.
- The Gini coefficient is a well-known measure of inequality.
- In this simple example, different estimation methods for the Gini coefficient are compared in a design-based simulation study.
- The standard estimation method (Eurostat definition; EU-SILC 2004) is compared to two semiparametric approaches, which fit a Pareto distribution to the upper tail of the data.
  - Hill (1975) introduced the maximum-likelihood estimator, which is therefore referred to as Hill estimator.
  - The partial density component estimator (PDC; Vandewalle et al. 2007) follows a robust approach.
- All these methods are implemented in the R package `laeken` (Alfons et al. 2010a).

Example: Gini coefficient II

First, the required packages and the data set are loaded, and the seed of the random number generator is set.

```
> library(simFrame)
> library(laeken)
> data(eusilcP)
> set.seed(12345)
```

Next, a control object for drawing 100 samples from the population is defined (stratified by regions, sampling of whole households).

```
> sc <- SampleControl(design = "region", grouping = "hid",
+     size = c(75, 250, 250, 125, 200, 225, 125, 150, 100),
+     k = 100)
```

Then, a control object for contamination is defined. The contamination level is set to 0.5% because EU-SILC data typically contain a very low amount of outliers.

```
> cc <- DCARContControl(target = "eqIncome", epsilon = 0.005,
+     grouping = "hid", dots = list(mean = 5e+05, sd = 10000))
```

Example: Gini coefficient III

The function for the simulation runs is quite simple as well.

```
> sim <- function(x, k) {
+     g <- gini(x$eqIncome, x$.weight)$value
+     eqIncHill <- fitPareto(x$eqIncome, k = k, method = "thetaHill",
+         groups = x$hid)
+     gHill <- gini(eqIncHill, x$.weight)$value
+     eqIncPDC <- fitPareto(x$eqIncome, k = k, method = "thetaPDC",
+         groups = x$hid)
+     gPDC <- gini(eqIncPDC, x$.weight)$value
+     c(standard = g, Hill = gHill, PDC = gPDC)
+ }
```

With all necessary objects available, running the simulation experiment is only one more command. Note that simulations are performed separately for each gender.

```
> results <- runSimulation(eusilcP, sc, contControl = cc,
+     design = "gender", fun = sim, k = 125)
```

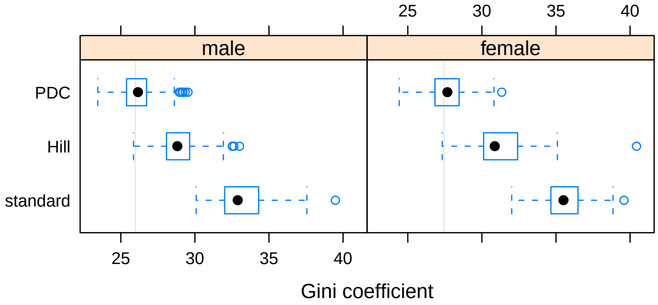
Example: Gini coefficient IV

In order to add a reference line to the plot of the results, the true values are computed from the population.

```
> tv <- simSapply(eusilcP, "gender", function(x) gini(x$eqIncome)$value)
```

An appropriate visualization method is selected automatically by the `plot()` method.

```
> plot(results, true = tv, xlab = "Gini coefficient")
```



Conclusions and outlook

- The flexible, object-oriented implementation of `simFrame` allows researchers to make use of a wide range of simulation designs with a minimal amount of programming.
- Control classes are used to handle data generation, sampling, contamination and the insertion of missing values.
- Developers can easily extend the existing framework with user-defined classes and methods.
- An appropriate plot method for the simulation results is selected automatically.
- Parallel computing using `snow` is supported to increase computational performance.
- Future plans include to extend the framework with different sampling methods and more specialized contamination and missing data models.

Acknowledgments

This work was partly funded by the European Union (represented by the European Commission) within the 7$^{\mathrm{th}}$ 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.

References I

A. Alfons. simFrame: Simulation framework, 2010. URL http://CRAN.R-project.org/package=simFrame. R package version 0.2.

A. Alfons, J. Holzer, and M. Templ. laeken: Laeken indicators for measuring social cohesion, 2010a. URL http://CRAN.R-project.org/package=laeken. R package version 0.1.1.

A. Alfons, M. Templ, and P. Filzmoser. An object-oriented framework for statistical simulation: The R package simFrame. Submitted to the Journal of Statistical Software, 2010b.

A. Alfons, M. Templ, P. Filzmoser, S. Kraft, and B. Hulliger. Intermediate report on the data generation mechanism and on the design of the simulation study. AMELI Deliverable 6.1, Vienna University of Technology, 2009. URL http://ameli.surveystatistics.net.

References II

J.M. Chambers. Programming with Data. Springer, New York, 1998. ISBN 0-387-98503-4.

J.M. Chambers. Software for Data Analysis: Programming with R. Springer, New York, 2008. ISBN 978-0-387-75935-7.

EU-SILC. Common cross-sectional EU indicators based on EU-SILC; the gender pay gap. EU-SILC 131-rev/04, Working group on Statistics on Income and Living Conditions (EU-SILC), Eurostat, Luxembourg, 2004.

M. Fowler. UML Distilled: A Brief Guide to the Standard Object Modeling Language. Addison-Wesley, 3rd edition, 2003. ISBN 978-0-321-19368-1.

B.M. Hill. A simple general approach to inference about the tail of a distribution. The Annals of Statistics, 3(5): 1163–1174, 1975.

P. L'Ecuyer, R. Simard, E.J. Chen, and W.D. Kelton. An object-oriented random-number package with many long streams and substreams. Operations Research, 50(6): 1073–1075, 2002.

References III

A.J. Rossini, L. Tierney, and N. Li. Simple parallel statistical computing in R. Journal of Computational and Graphical Statistics, 16(2): 399–420, 2007.

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

D. Sarkar. `lattice`: Lattice Graphics, 2010. URL `http://CRAN.R-project.org/package=lattice`. R package version 0.18-8.

H. Sevcikova and T. Rossini. `rlecuyer`: R Interface to RNG with Multiple Streams, 2009. URL `http://CRAN.R-project.org/package=rlecuyer`. R package version 0.3-1.

L. Tierney, A.J. Rossini, and N. Li. `snow`: A parallel computing framework for the R system. International Journal of Parallel Programming, 37(1): 78–90, 2009.

References IV

L. Tierney, A.J. Rossini, N. Li, and H. Sevcikova. snow: Simple Network of Workstations, 2008. URL http://CRAN.R-project.org/package=snow. R package version 0.3-3.

B. Vandewalle, J. Beirlant, A. Christmann, and M. Hubert. A robust estimator for the tail index of Pareto-type distributions. Computational Statistics & Data Analysis, 51(12): 6252–6268, 2007.