# simSummary: Simulation summary

by Gregor Gorjanc

May 16, 2012

**Abstract**

**simSummary** is a small utility package which eases the process of summarizing simulation results. Simulations often produce intermediate results - some focal statistics that need to be summarized over several scenarios and many replications. This step is in principle easy, but tedious. The package **simSummary** fills this niche by providing a generic way of summarizing the focal statistics of simulations. The useR must provide properly structured input, holding focal statistics, and then the summary step can be performed with one line of code, calling the `simSummary` function.

## 0.1 Introduction

Simulations are often used to study complex processes. In statistics, simulations are often used to study the uncertainty of estimates which is due to the sampling variability of their inputs (data). Almost any simulation can be divided into two parts. The first part is to carry out a simulation of a complex process. This part differs from simulation to simulation in essential ways. Often some focal statistics are computed to summarize the simulated process, e.g., the mean. To capture the variability of a studied process, several replications are created and for each replication the focal statistics are saved. The second part of a simulation is to summarize the focal statistics collected. This is in principle a very simple task, though often a tedious one due to the need to summarize the focal statistics over several simulation scenarios and many replications. Since this task is structurally always the same and is independent of the simulation, there is an open niche for a generic tool. **simSummary** is a small utility package with a single function of the same name (`simSummary`), which tries to fulfill this need by easing the process of summarizing focal statistics over simulation scenarios and replications.

## 0.2 Description with example

A simple example will be used to demonstrate the use of the **simSummary** package. Say we are studying lamb growth around 60 days using a linear regression of body weight ($y$) on age ($x$):

$$y_i = \alpha + \beta(x_i - 60) + e_i,$$

where $\alpha$ is an intercept (the average lamb body weight at 60 days) and $\beta$ is a slope (the average lamb growth rate or daily gain) of a regression line. We would like to quantify the variability of some focal statistics in relation to the sample size. Three scenarios with differing numbers of observations will be tested, and the focal statistics will be the estimates of the parameters $\hat{\alpha}$ and $\hat{\beta}$ and the coefficient of determination $\hat{R}^2$. This is not a very complex simulation, but nicely shows the usage of the **simSummary** package.

First we need to set the parameters of the studied process and simulation.

```
> ## Process (lamb growth) parameters
> xMin  <- 40            ## minimal age
> xMax  <- 80            ## maximal age
> alpha <- 20            ## ave. weight at 60 days
> beta  <- 0.35          ## growth rate
> sdE   <- 2             ## residual variation
> ## Simulation parameters
> nY <- c(10, 100, 1000) ## scenarios
> nS <- length(nY)       ## no. scenarios
> nR <- 100              ## no. replications
```

Then, containers for the focal statistics need to be set up. There are several ways to set up such containers, but for the use of the **simSummary** package, we must follow two rules:

- set up the "outer" list of length equal to the number of replications (`nR`) and
- each element of an "outer" list must also be a list (the "inner" list).

For the lamb growth simulation, the two parameter estimates and the coefficient of determination for each scenario will be stored in the corresponding "inner" list. The way the focal statistics are stored in the "inner" list is free as long as the elements of this list are either numeric vectors, matrices, or arrays. These three object classes should cover the majority of needs.

```
> ## Outer list
> sim <- vector(mode="list", length=nR)
> ## Inner list
> simI <- vector(mode="list", length=2)
> names(simI) <- c("coef", "R2")
> simI$coef <- matrix(nrow=2, ncol=nS)
> simI$R2   <- matrix(nrow=1, ncol=nS)
> colnames(simI$coef) <- colnames(simI$R2) <- nY
> rownames(simI$coef) <- c("alpha", "beta")
```

The simulation of the lamb growth process could be performed as shown below, which completes the first part of the simulation. In order to show the structure of the "inner" lists, the focal statistics collected from the first two replications are printed out.

```
> for(i in 1:nR) {
+   sim[[i]] <- simI
+   for(j in 1:nS) {
+     x <- runif(n=nY[j], min=xMin, max=xMax) - 60
+     y <- alpha + beta * x + rnorm(n=nY[j], sd=sdE)
+     tmp <- lm(y ~ x)
+     sim[[i]]$coef[, j] <- coef(tmp)
+     sim[[i]]$R2[j] <- summary(tmp)$r.squared
+   }
+ }
> sim[1:2]

[[1]]
[[1]]$coef
              10        100      1000
alpha 19.5779988 20.1111013 19.967707
beta   0.3211597  0.3637645  0.344984

[[1]]$R2
            10       100      1000
[1,] 0.7167131 0.8626802 0.7893356


[[2]]
[[2]]$coef
              10        100       1000
alpha 19.3975637 19.7316175 20.0177078
beta   0.4361056  0.3404666  0.3547691

[[2]]$R2
            10       100      1000
[1,] 0.9194269 0.8314889 0.8187996
```

The second part of the simulation is to summarize the collected focal statistics over the simulation scenarios and replications. With the use of the `simSummary` function, this is very easy. If the interest is in means and standard deviations, the following three lines of code will: i) install the package from CRAN, ii) load the package, and iii) summarize the simulation.

```
> # install.packages(pkg="simSummary")
> library(package="simSummary")
> simSummary(x=sim, FUN=c("mean", "sd"))

$mean
$mean$coef
              10        100       1000
```

```
alpha 20.0211092 20.0099041 19.9893846
beta   0.3515219  0.3535567  0.3501057

$mean$R2
            10       100      1000
[1,] 0.7974498 0.8090594 0.8037297


$sd
$sd$coef
               10         100         1000
alpha 0.72720165 0.21432759 0.058253783
beta  0.06077136 0.01905526 0.004965014

$sd$R2
            10         100         1000
[1,] 0.121727 0.02853429 0.009320538
```

The only two requirements for the `simSummary` function are i) properly structured input (an "outer" list of "inner" lists holding numeric vectors, matrices, or arrays) and ii) the summary functions must return a single value, such as `length`, `mean`, etc., but not `range`, `table`, etc. An error message is thrown when any of these two requirements is not met. The output of `simSummary` is also an "outer" list of "inner" lists, where its "inner" lists have the same structure as in the input. There is one instance of an "inner" list for each summarizing function.

Simulations are often time consuming, and often run as a (parallel) batch job. The summary step can then be performed when all jobs finish. However, to obtain preliminary results, a script can be set up that creates the structured input and fills it with the available values. Missing values can then be accommodated in the `simSummary` function with the argument `na.rm=TRUE` which is passed to the summarizing functions.

```
> ## Mimick simulation in progress
> sim[[100]]$coef[] <- NA
> sim[[100]]$R2[]   <- NA
> simSummary(x=sim, FUN=c("nobs", "mean"),
+            na.rm=TRUE)

$nobs
$nobs$coef
      10 100 1000
alpha 99  99   99
beta  99  99   99

$nobs$R2
     10 100 1000
[1,] 99  99   99


$mean
```

```
$mean$coef
              10        100       1000
alpha 20.0250648 20.0091014 19.9894409
beta   0.3507131  0.3535304  0.3501186

$mean$R2
           10       100      1000
[1,] 0.7962999 0.8089798 0.8038047
```

## 0.3 Development

The published version of the **simSummary** package is hosted at CRAN[1]. For the development **inlinedocs** package, (Inlinedocs development team, 2011) is used to mix the code and documentation in one file, while the **svUnit** package (Grosjean, 2012) is used for unit testing. Any useR can run the package unit tests with `simSummary_unitTests()`. Contributions to the package are welcome.

## 0.4 Summary

**simSummary** is a small utility package which eases the process of summarizing selected focal statistics in simulations. The only effort needed is to properly structure the input, while the summary step is easy to perform. By using the **simSummary** package useRs can devote more time to the analysis of simulation results than to the tedious development of simulation specific summarizing code.

# R session information

```
> sessionInfo()

R version 2.15.0 (2012-03-30)
Platform: x86_64-pc-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.utf8       LC_NUMERIC=C
 [3] LC_TIME=en_US.utf8        LC_COLLATE=C
 [5] LC_MONETARY=en_US.utf8    LC_MESSAGES=en_US.utf8
 [7] LC_PAPER=C                LC_NAME=C
 [9] LC_ADDRESS=C              LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.utf8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] simSummary_0.1.0 svUnit_0.7-5     abind_1.4-0
```

[1] http://CRAN.R-project.org/package=simSummary

```
loaded via a namespace (and not attached):
[1] gdata_2.8.0  gtools_2.6.2 tools_2.15.0
```

# Bibliography

Inlinedocs development team. inlinedocs: Convert inline comments to documentation. 2011. URL `http://CRAN.R-project.org/package=inlinedocs`.

Grosjean, P. SciViews-R: A GUI API for R. UMONS, Mons, Belgium. 2012. URL `http://www.sciviews.org/SciViews-R`.