Simulation Study Tutorial using R on your Laptop/PC Version 1.2

Elise Dusseldorp, Jeffrey Durieux, Juan C. Gonzalez, and Bunga C. Pratiwi

16/10/2018

Contents

1 Introduction

In this document, we describe the set-up of a simulation study and the performance of it using R on your laptop/PC. The document is structured as follows:

- Chapter 2: Specification of a simulation study;
- Chapter 3: Preparation of a simulation study;
- Chapter 4: Performance of a simulation study on a laptop/PC;
- Chapter 5: Collecting the results in one large matrix and evaluation of the results by anova;

2 Specification of a simulation study

In a full factorial simulation study, we consider the following 4 steps:

- Step 1: Research Question
- Step 2: Gauge and Design
- Step 3: Statistical Analysis
- Step 4: Evaluation Criteria

2.1 Step 1: Research Question

Formulate the research question you would like to answer by means of a simulation study. In our example, the research question is: Does the Mann-Whitney U test show higher power (i.e., lower Type II error) than the independent samples t-test? And does the difference in power between the two methods depend on the total sample size and the mean difference between the two groups?

We hypothesize that the power of the Mann-Whitney test is higher than the independent samples t-test, especially in smaller samples.

2.2 Step 2: Gauge and Design

Continue with the formulation of:

- the true scenario(s)/model(s) from which you will generate your artificial data, and specify from which distributions you generate the variables that are included in the model.
  - In our example, we have a continuous, standardized outcome variable and one variable group that has two categories (i.e., two groups).
- specify the design factors (i.e. the factors you are going to vary systematically). In our example, we vary two factors Sample Size (Samp) and Effect Size (Es). These factors can be easily extended with more than two. We refer to these factors as:
  - Samp (sample size, having 4 levels: N = 10, 20, 40, and 80);
  - Es (effect size, having 3 levels: d = 0.2, 0.5 and 0.8). The standardized difference in means between the two groups is going to be considered as the measure of the effect size.
- number of replications within each cell: K. In our small example, $K = 10$.

For our example, our design contains in total 4 (Samp) by 3 (Es) 12 cells. For each cell we will generate $K = 10$ data sets. Thus, in total, we analyze 120 data sets.

2.3 Step 3: Statistical Analysis

Specify the type of analysis you want to perform for each dataset (of each cell of the design). Often you want to compare two types of analysis methods (for example, a new method compared to an old method).

2.4 Step 4: Evaluation Criteria

Specify the evaluation criteria which you want to use to compare the two methods. In our example, we want to know whether the test correcly rejects the null hypothesis of no difference in means between the two groups using $\alpha = 0.5$.

3 Preparation of a simulation study

Write the following functions in `R`:

- one function that generates one data set. The input arguments of this function are the type of true model (true scenario) and the levels of the factors of the simulation design. For the example, we use one scenario and two design factors. We refer to the data generation function as *MyDataGeneration.R*. The output of this function is a simulated data set (a dataframe or matrix). We refer to this output object as SimDat (see *MySimulationCell()*).

- One or two functions that analyse the data according to one or two methods. In our example, we use two methods to analyze each simulated data set and refer to the functions for each method as *Method_new.R* and *Method_old.R*. The input argument of each of these functions is a dataset (SimDat). The output of this function contains the relevant results. In our example, we refer to this output as MyAnalysisResult (see *MySimulationCell()*);
- a function that applies the evaluation criterion. In our example, we refer to this function as *MyEvaluationPC.R*. The input of this function is MyAnalysisResult.

3.1 First steps of your simulation study:

Below we guide you through the main steps of the simulation study step by step. Your design matrix is computed, analyses are performed and results are computed. Moreover, saving your output in a workspace is specified.

Some usefull and sometimes necessary *R* functions:

- expand.grid()
  - usefull functions that creates a data frame from all combinations of factor variables
- set.seed()
  - important for replication of your simulation study
- do.call()
  - neat function to execute a function with a list of supplied arguments

First, we define our design matrix and load the functions that generates the data, run the analyses and applies the evaluation criterion:

```
# Preparation of the analysis


### Initialize the factors of your design:

samp <- c(10, 20, 40, 80)
es <- c(0.2, 0.5, 0.8)

##And create the simulation design matrix (full factorial)
# Design is a data.frame with all possible combinations of the factor levels
# Each row of the design matrix represents a cell of your simulation design

Design <- expand.grid(samp = samp, es = es)


###Preparation of the analysis:

# If you use R packages that are not standard:
# Install the relevant R packages, for example:

#install.packages("ica")
#Always use library() to activate the package
#library(ica)
#NB we do not use this package for our example

### Source the relevant R functions of our example
### These functions are available from:
### https://github.com/Github-MS/Shark/tree/master/Scripts
```

```r
source("MyDataGeneration.R")
source("Method_new.R")
source("Method_old.R")
source("MyEvaluationPC.R")
```

3.2 Simulation function for one cell of the design.

We first perform a simulation within 1 cell of the design. For each cell of the design, we generate K data sets and we save the results in a workspace. The results are collected in the matrix MyResult. The number of rows of this matrix is K and the number of columns equals the number of outcome variables times the number of methods. In the example, we compare 2 methods: Method 1 (*Method_new.R*) and Method 2 (*Method_old.R*) and we have 1 outcome variable: whether the p-value of the test is significant or not. Thus 2 columns are sufficient. We recommend to start testing this function using a small number of replications (e.g. $K = 2$)

```r
MySimulationCell<- function(Design = Design, RowOfDesign = 1, K = 2){
# Input arguments:
  #Design = designmatrix
  # RowOfDesign: number that refers to the row of the design matrix = one cell
  # K: Total number of replications = number of data sets generated in one cell

#Create matrix or dataframe to store the results:

MyResult <- matrix(NA, nrow = K, ncol=2)

#create a loop over the replications k = 1 to K:
for (k in 1:K){

# Generate data
# set a random number seed to be able to replicate the result exactly
set.seed((k + 1000)*RowOfDesign)
SimDat <- do.call(MyDataGeneration, Design[RowOfDesign,] )

# Analyze data set with Method_new
MyAnalysisResult1 <- Method_new(SimDat)

#Analyze data set with Method_old
MyAnalysisResult2 <- Method_old(SimDat)


#Combine relevant results of the analysis by the two methods in a vector (optional)
MyAnalysisResult <- c(MyAnalysisResult1$p.value, MyAnalysisResult2$p.value)

#Evaluate the analysis results of Method_new (Result1) and Mehtod_old (Result2)
MyResult1 <- MyEvaluationPC(MyAnalysisResult1)
MyResult2 <- MyEvaluationPC(MyAnalysisResult2)

#store the results in the right row k of your result matrix:
#We only store the second result which is the evaluation criterion
MyResult[k, ] <- c(MyResult1, MyResult2)

}
#save the time to run the analyses of K data sets in one cell of the design.
```

```
return(MyResult)
}

Row <- 1
tmp <- proc.time()
MyResult <- MySimulationCell(Design = Design, RowOfDesign = Row, K = 2 )
time <- proc.time() - tmp
# Write output of one cell of the design
save(MyResult, file =paste("MyResult", "Row", Row,".Rdata" , sep =""))
#optional to save timing of analyses of K replications in 1 cell
save(time, file =paste("Time", "Row", Row, ".Rdata" , sep =""))
```

3.3 Start your full simulation

Now it is time to create a for-loop for the rows of your Design matrix and perform the full simulation study. Optionally (if time allows) increase the number of replications K in a cell. If you would like to have more replications in a fast way, you may consider to submit you jobs to Shark (cluster computer).

```
#Total number of cells
TotalCells <- nrow(Design)
for (i in 1:TotalCells){
  Row <- i
 MyResult <- MySimulationCell(Design = Design, RowOfDesign = Row, K = 10 )
 # Write output of one cell of the design
save(MyResult, file =paste("MyResult", "Row", Row,".Rdata" , sep =""))
}
```

4 Collecting the results

After the full analysis, our number of saved workspaces equal the number of cells in our design. We are going to collect these workspaces in one data matrix. Below you can find a script that collects the right output and automatically stores the results in a matrix. This matrix can be used for further analyses (optionally in SPSS) such as an ANOVA.

```
# Collect all results and put it in a matrix
# original design is in the matrix Design
# Total number of rows of Design = TotalCells

#first load one workspace, for example, cell 1 of the Design matrix
#save the number of replications within that cell

Row <- 1
load(paste("MyResult", "Row", Row,".Rdata" , sep =""))
K <- nrow(MyResult)

#Initialize a very large matrix with number of rows = K * TotalCells

Results_sim <- matrix(NA, ncol = ncol(MyResult), nrow = K*TotalCells)


#Fill in this matrix with the results that you obtained for each cell of the design (= each of the work
```

```r
#Thus loop over the rows of Design

for (i in 1:TotalCells){
  Row <- i
  load(paste("MyResult", "Row", Row,".Rdata" , sep =""))
  Results_sim[(K*(i-1)+1):(i*K), ] <- MyResult
}


#rbind the Design matrix  K times
Results_des <- do.call(what = rbind, args = replicate(K, Design, simplify = F))
#rbind the vector 1:K TotalCelss times
Results_K <- do.call(what = rbind, args = replicate(TotalCells, matrix(c(1:K), ncol = 1), simplify = F))

#Create the final results matrix
ResultsSimAll <-as.data.frame(cbind(Results_des, K = Results_K, Results_sim))


#Give the columns the right name:
names(ResultsSimAll) <- c("Samp", "Es", "K", "Method1", "Method2")
head(ResultsSimAll)
```

```
##   Samp  Es K Method1 Method2
## 1   10 0.2 1       0       0
## 2   20 0.2 2       0       0
## 3   40 0.2 3       0       0
## 4   80 0.2 4       0       0
## 5   10 0.5 5       0       0
## 6   20 0.5 6       0       0
```

```r
#save the combined results in one workspace
save(ResultsSimAll, file = "AllResultsSim.Rdata")

#Optionally save it as a .sav file

library(haven)
workingDirectory <- paste(getwd(), "/", sep = "")
filenameToSave <- paste(workingDirectory, "AllResultsSim", ".sav", sep = "")
write_sav(ResultsSimAll, filenameToSave)
```

5 Analyzing the final results of the simulation study

Now you can perform your repeated measures ANOVA in R loading the workspace "AllResults.Rdata" or in SPSS using the file "AllResultsSim.sav". For the example, the within factor is Method (1 or 2) and the between factors are Samp and Es.