# Tutorial

We extended the analyses with the *berksonGenerator* function. The R-code can be found in a separate file in the Supplementary Materials. In the following tutorial, we first briefly explain the purpose of the *berksonGenerator* function, show how to use the function to simulate a dataset and finally explain how to run a simulation study by combining the function with *netSimulator* from the *bootnet* R-package.

## Purpose of berksonGenerator.

This function can be used in the freely available programming language R (Team & R Development Core Team, 2016). The function is designed to generate data from which the sum-scores of the variables in the network exceed or are equal to the selected cut-off value. It generates the data given a network structure; either partial correlation network for ordinal data or a weighted adjacency matrix for binary data. The purpose of this function is to provide a way to easily conduct simulation studies on the performance of network estimations with varying sample sizes and cut-off values. To run such a simulation study, the *berksonGenerator* function can be used in combination with the *netSimulator* function from the *bootnet* package.

## Generating a new dataset.

In all subsequent code we use to illustrate the function, the input is an Ising network with 10 nodes:

N <- 10 # Number of nodes

Weight <- 0.5

library(igraph)

Graph <- as.matrix(get.adjacency(watts.strogatz.game(1, N, 1, p = 0)))\*

 Weight

Thresh <- -rowSums(Graph) / 2

input <- list(graph = Graph, intercepts = Thresh)

 As a first argument, *berksonGenerator* takes a list containing the elements graph and thresholds (or intercepts). The second argument specifies the sample size of the to be generated dataset. The third argument can be used to define the cut-off value to select sum-scores on. The fourth argument can be set to either “GGM” or “Ising” and indicates the method used to generate new data. The last argument is the maximum number of iterations the function lasts without adding a new observation to the dataset. The default is 1000. This argument is added to the function, to make sure the function does not get stuck in an infinite loop (e.g., when the cut-off sum-score is set higher than the maximum sum-score). For example, the following code generates a binary dataset with 500 observations and a cut-off value of 2:

source("berksonGenerator.R")

newDatasetIsing <- berksonGenerator(input, n = 500, cutoff = 2,

 default = "Ising")

## Conducting a simulation study.

To not only generate a dataset, but to see the effect on the performance of an estimated network when only selecting observations with a certain sum-score, the function can be used as the dataGenerator argument in the *netSimulator* function of the *bootnet* package. Details about the use of *netSimulator* have been published elsewhere (Epskamp & Fried, 2017). When the code in the Supplementary materials is saved in a file in the current working directory called *berksonGenerator.R*, the following code produces and plots a simulation with one selected cut-off value of 2 and for sample sizes of 500 and 1000:

library(bootnet)

Sim1 <- netSimulator(input, dataGenerator = function(n, input) {

 source("berksonGenerator.R")

 berksonGenerator(input, n, cutoff = 2,

 default = "IsingFit")},

 default = "IsingFit", nCores = 8, nReps = 100, nCases = c(500, 1000))

plot(Sim1)

 If we want to compare the effect of multiple cut-off sum-scores with one another, we can combine different *netSimulator* outputs in one data frame. The following code shows how you can implement multiple selections rules:

Sim1$cutoff <- 2

Sim2 <- netSimulator(input,dataGenerator = function(n, input){

 source("berksonGenerator.R")

 berksonGenerator(input, n, cutoff = 4, default = "IsingFit")

 }, default = "IsingFit", nCores = 8, nReps = 100, nCases = c(500, 1000))

Sim2$cutoff <- 4

totalSim <- rbind(Sim1, Sim2)

plot(totalSim, color = "cutoff")

With color = "cutoff" we specified to plot both cut-off scores as different categories. See Figure 7 for an example of the resulting plots. The plots may be used to gain insight into the correlation, sensitivity, and specificity of the generated networks. As expected, on average, the networks with the highest cut-off value have the lowest correlation, sensitivity, and specificity.



*Figure 7*. The output of the simulation for Ising model estimations with different cut-off values (2 and 4) and sample sizes (500 and 1000). Every condition was simulated 100 times. The boxplots represent the distribution of those measures. The vertical panels indicate the different measures: correlation, sensitivity, and specificity.