Stat4DS / Homework 02

Luca Scofano / Davide Zingaro

19 dicembre 2019

Exercise: Connect your brain

In this task, our goal is to identify and evaluate the dependencies that exist between the various cortical regions.

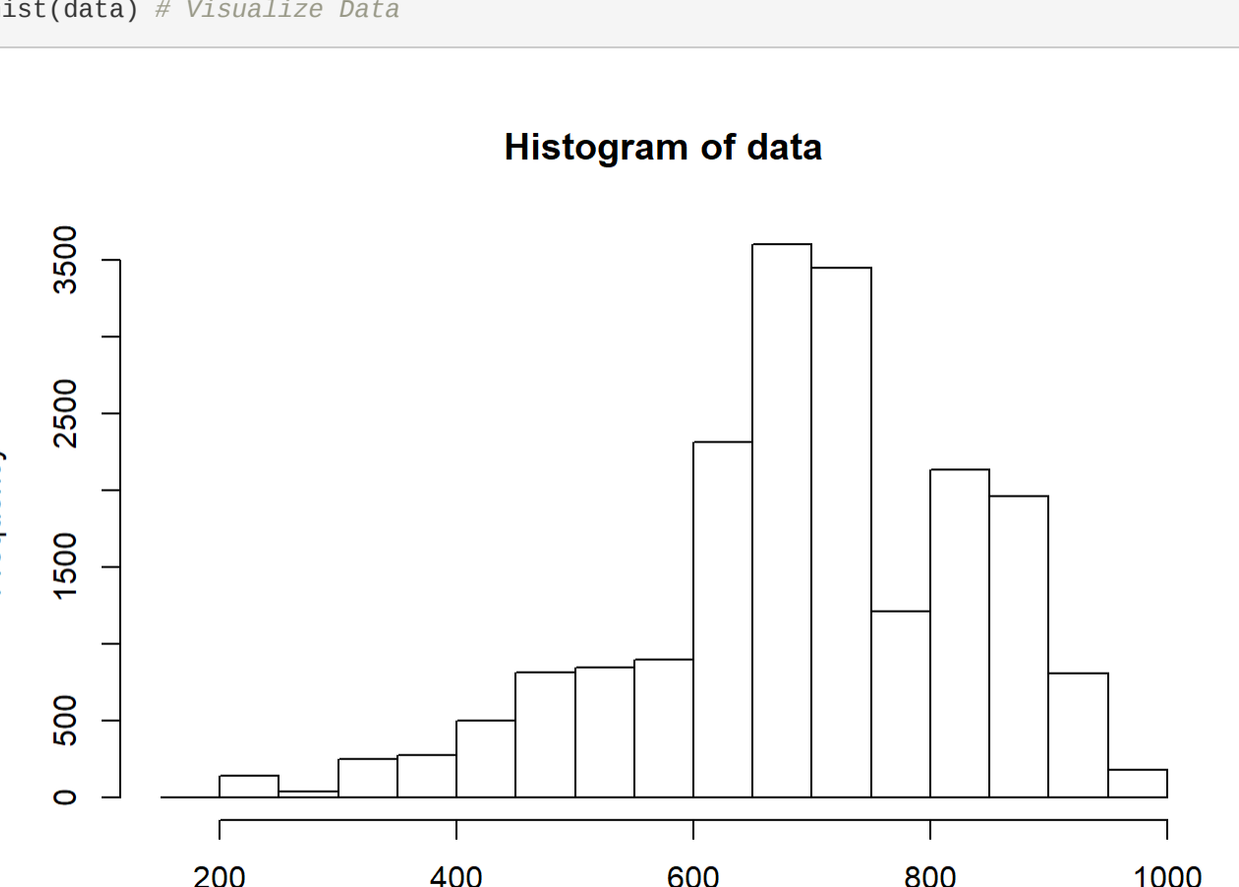
Importing libraries…

```
library(igraph)
library(ggraph)
library(boot)
library(SIN)
```

Quick analysis

Let's start with a quick analysis of the dataset **mts**:

```
load("C:\\Users\\Luca\\Desktop\\WIP\\Homework 2 SDS\\.RData")
data <- mts # Load Data
hist(data) # Visualize Data
```




```
# Variables
n.features <- ncol(data)
n <- nrow(data) # number of observations
data.cor <- cor(data) # matrix of correlations of the sample (Pearson's)
feature.names <- colnames(data)
```

Initialize edge matrix

```
# Initialize edge matrix filled with NA
edge <- matrix(NA, n.features, n.features)
colnames(edge) <- feature.names
rownames(edge) <- feature.names
```

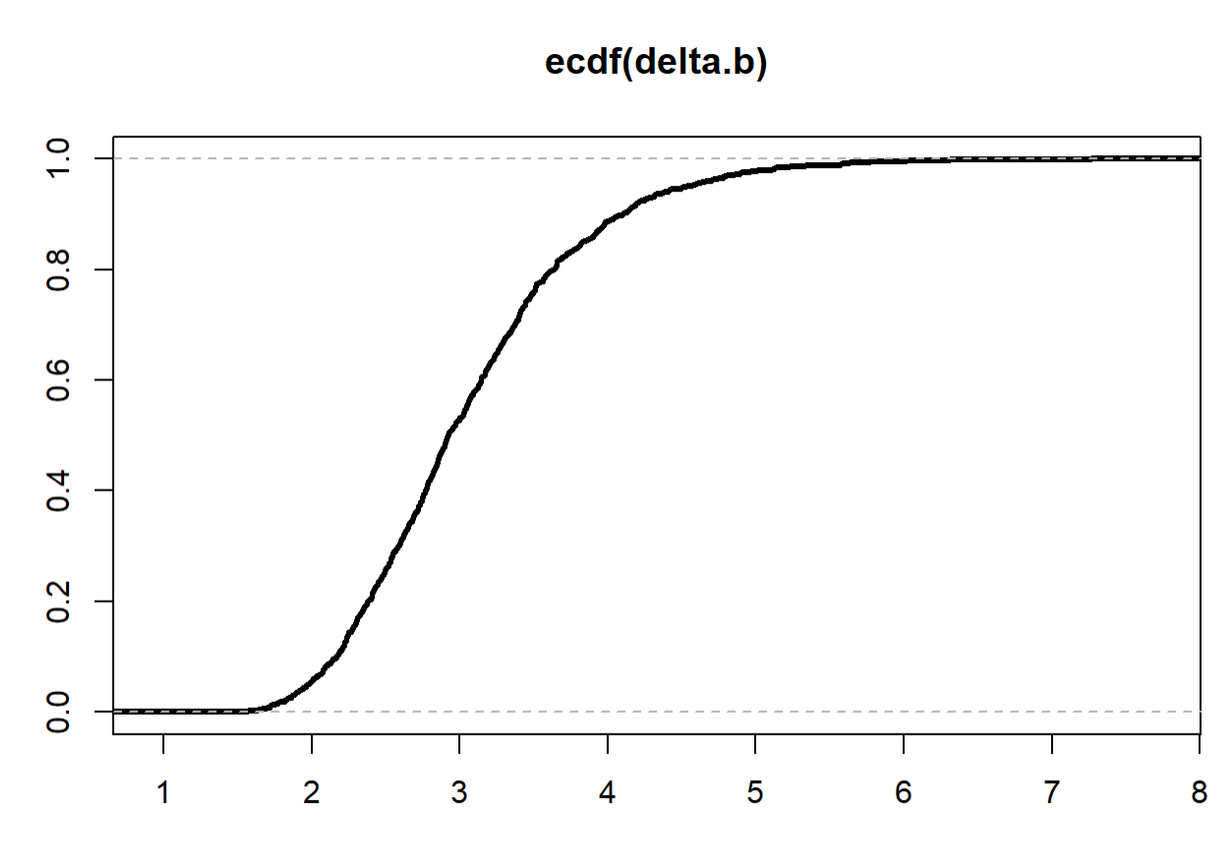
Non Parametric Bootstrap procedure and ecdf

With this data, let's consider the Pearson correlation, and implement the bootstrap procedure described according to "simultaneous bootstrapped ci's for a generic association measure ρ".

```
set.seed(123)
library(boot)

B <- 1000
delta.beta <- function(data, indices) {
  d <- data[indices,] # allows boot to select sample
  pearson <- cor(d) # creates a correlation matrix for each boostrap
  data.cor <- cor(data) # matrix of correlations of the sample
  delta <- sqrt(nrow(data))*max(abs(pearson-data.cor))
  return(delta)
}

bootdelta <- boot(data = data, statistic = delta.beta, R = B)
```

Now we have the list of delta.b and we can compute the ecdf

```
delta.b <- bootdelta$t
f.hat <- ecdf(delta.b)

# Plot the ecdf
plot(f.hat, ylab="Fn(x)", verticals = FALSE, lwd = 3)
```



Confidence Intervals

We compute he confidence intervals to create the EDGE matrix. We place 1 in the matrix (the edge exists) only if:

$$[-\epsilon, +\epsilon] \cap C_{n,\alpha} = \varnothing$$

```
# let's start with alpha = 0.95
m <- choose(81, 2)
alpha1 <- 0.05
t.alpha = quantile(f.hat, probs = (1-alpha1), type =1) # inverse of the ECDF function
ci.lower = data.cor-t.alpha/sqrt(n) # lower bound of the CI
ci.upper = data.cor+t.alpha/sqrt(n) # upper bound of the CI

# and alpha = 0.99
alpha2 <- 0.01
t.alpha2 = quantile(f.hat, probs = (1-alpha2), type =1) # inverse of the ECDF function
ci.lower2 = data.cor-t.alpha2/sqrt(n) # lower bound of the CI
ci.upper2 = data.cor+t.alpha2/sqrt(n) # upper bound of the CI
```

Since we have the correlaction matrix and the confidence intervals relative to alpha = 0.95, we can plot our association graph.

Edge matrix

1. EPSILON = 0

```
eps <- 0 # we initially set epsilon as 0

for (i in 1:n.features){
  for(j in 1:n.features){

    # If 0 is not included in the CI it means there is an edge between the two nodes
    if ((ci.lower[i,j] < eps & eps < ci.upper[i, j]) == FALSE) edge[i,j] = 1 # we set one in the adjacency matrix
    else edge[i,j] = 0 # if 0 is included in CI we set 0 in the adjacency matrix
    if (i == j) edge[i,j] = 0 # self edge is excluded
  }
}

graph.eps1 = graph.adjacency(edge, mode = "undirected")
```

```
# Density
density.eps1 <- edge_density(graph.eps1, loops=FALSE)
density.eps1
```

```
## [1] 0.862963
```

As we can see the graph is very dense, since the maximum value is 1 it means that not all the nodes are connected to each other although it's very close to this result.

```
# Connected components
components(graph.eps1) # all nodes are connected
```

```
## $membership
##  1R  2R  3R  4R  5R  6R  7R  8R  9R 10R 11R 13R 17R 18R 19R 20R 21R 22R 23R 24R
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 25R 28R 29R 30R 31R 32R 34R 35R 36R 37R 38R 39R 40R 41R 42R 43R 44R 46R 47R  1L
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  2L  3L  4L  5L  6L  7L  8L  9L 10L 11L 13L 17L 18L 19L 20L 21L 22L 23L 24L 25L
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 27L 28L 29L 30L 31L 32L 33L 34L 35L 36L 37L 38L 39L 40L 41L 42L 43L 44L 45L 46L
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 47L
##   1
##
## $csize
## [1] 81
##
## $no
## [1] 1
```

All the nodes are connected.

```
# Number of edges
sum(edge)
```

```
## [1] 5592
```

2. EPSILON = 0.5

```
eps2 <- 0.5 # we initially set epsilon as 0
edge2 <- matrix(NA, n.features, n.features)

for (i in 1:n.features){
  for(j in 1:n.features){
    # If 0 is not included in the CI it means there is an edge between the two nodes
    if (eps2 < ci.lower[i, j] | -eps2 > ci.upper[i, j]) edge2[i,j] = 1
    else edge2[i,j] = 0
    if (i == j) edge2[i,j] = 0 # self edge is excluded
  }
}
graph.eps2 = graph.adjacency(edge2, mode = "undirected")

# Components
components(graph.eps2)
```

```
## $membership
##  [1]  1  1  1  2  3  1  3  1  1  1  1  1  1  1  1  1  1  1  1  1  4  1  1  5
## [26]  1  1  1  1  1  1  6  7  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
## [51]  1  1  1  1  1  1  1  1  1  8  1  1  1  1  1  1  1  1  9  1  1  1  1 10  1
## [76]  1  1  1  1  1  1
##
## $csize
##  [1] 71  1  2  1  1  1  1  1  1  1
##
## $no
## [1] 10
```

```
# Number of edges
sum(edge2) # number of edges
```

```
## [1] 2958
```

3. EPSILON = 0.65

```
eps3 <- 0.65 # we initially set epsilon as 0
edge3 <- matrix(NA, n.features, n.features)

for (i in 1:n.features){
  for(j in 1:n.features){
    # If 0 is not included in the CI it means there is an edge between the two nodes
    if (eps3 < ci.lower[i, j] | -eps3 > ci.upper[i, j]) edge3[i,j] = 1
    else edge3[i,j] = 0
    if (i == j) edge3[i,j] = 0 # self edge is excluded
  }
}
graph.eps3 = graph.adjacency(edge3, mode = "undirected")

# Components
components(graph.eps3)
```

```
## $membership
##  [1]  1  1  1  2  3  1  4  1  1  5  1  6  7  8  1  1  1  1  1  1  1  9 10 11 12
## [26]  1  1  1 13 14  1 15 16  1  1 17  1  1 18  1  1  1  1  1  1 19  1  1  1 20
## [51] 21  1  1  1 22  1  1  1  1 23 24  1 25 26 27  1  1 20 28  1  1  1 29 30 31
## [76]  1  1  1  1  1  1
##
## $csize
##  [1] 50  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  2  1  1  1  1
## [26]  1  1  1  1  1  1
##
## $no
## [1] 31
```

```
# Number of edges
sum(edge3) # number of edges
```

```
## [1] 998
```

PLOTS

```
# Change colours
V(graph.eps1)$color[1:39] <- "red"
```

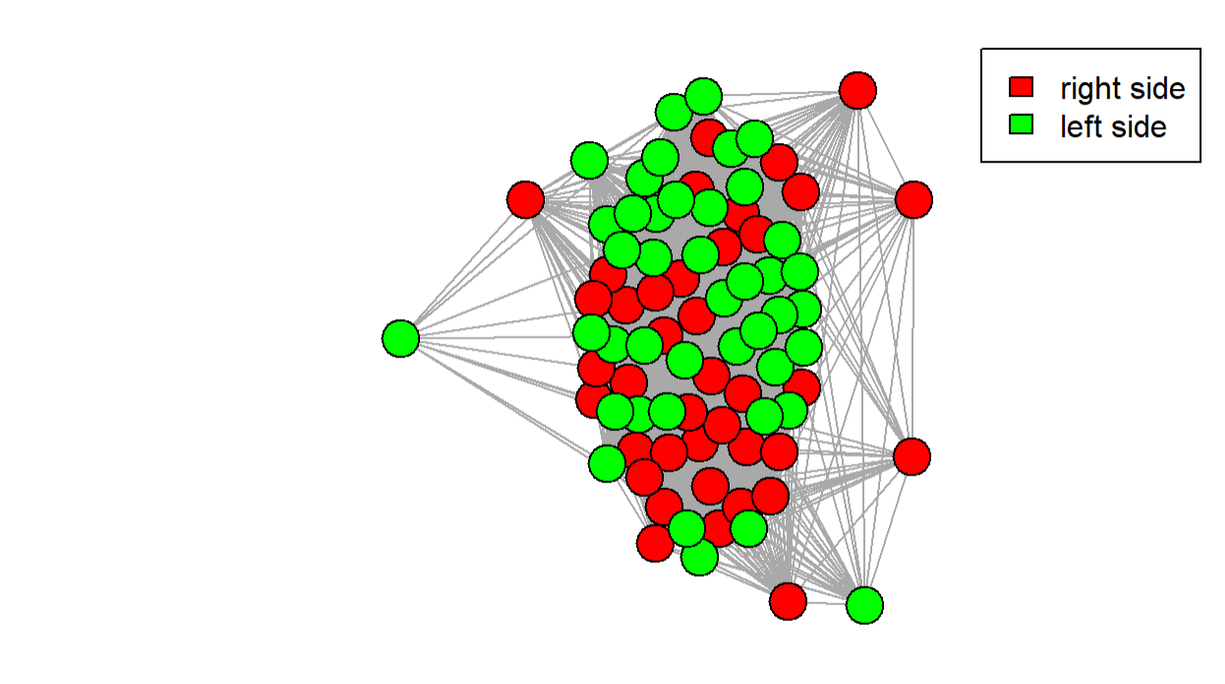
```
## Warning in vattrs[[name]][index] <- value: number of items to replace is not a
## multiple of replacement length
```

```
V(graph.eps1)$color[40:81] <- "green"

# Change colours
V(graph.eps2)$color[1:39] <- "red"
```

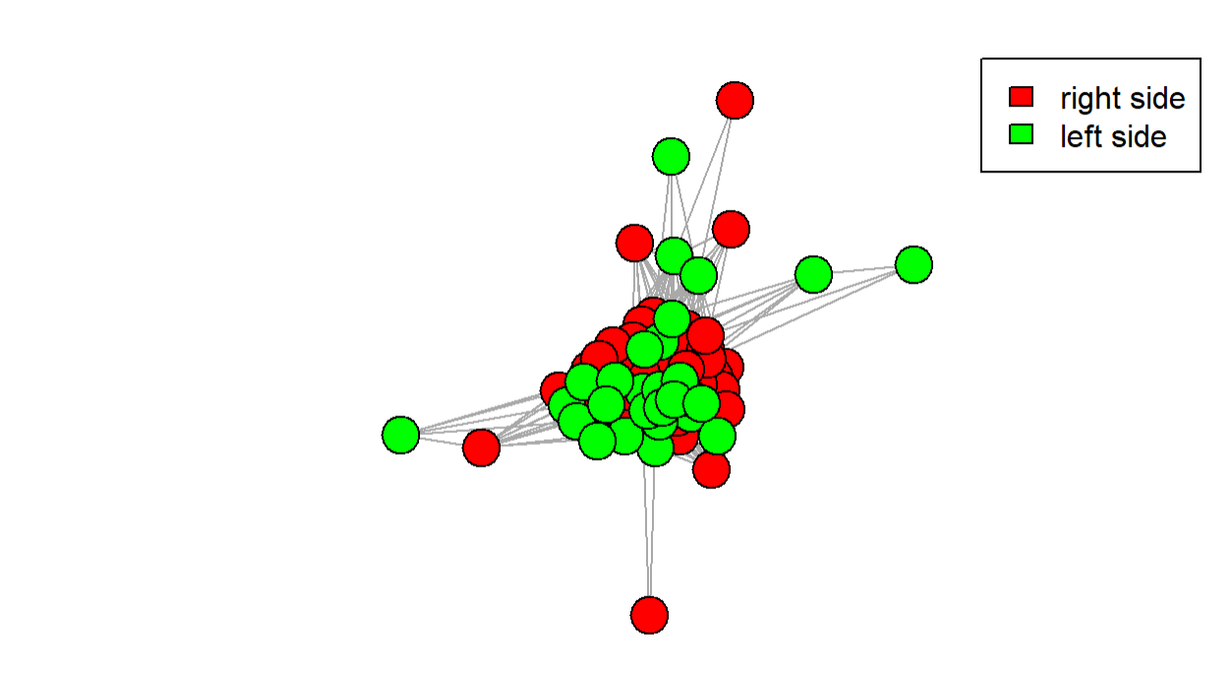
```
## Warning in vattrs[[name]][index] <- value: number of items to replace is not a
## multiple of replacement length
```

```
V(graph.eps2)$color[40:81] <- "green"
```

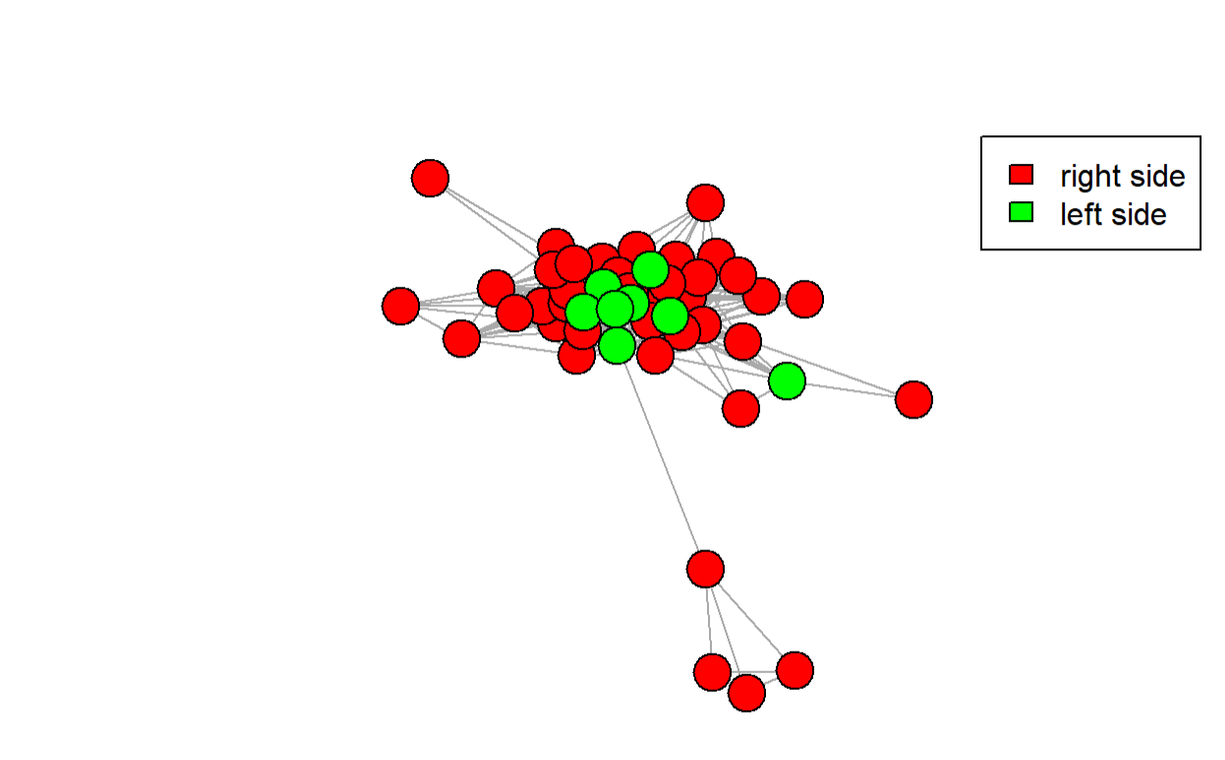
```
# Plot only nodes with more than two edges
clean <- function(graph){
  Isolated = which(degree(graph)<2)
  G2 = delete.vertices(graph, Isolated)
  return (G2)
}

# Epsilon = 0

plot.igraph(clean(graph.eps1), ylab = expression(epsilon == 0), vertex.color=V(graph.eps1)$colorx, vertex.label =
NA)
legend("topright",
       c("right side","left side"),
       fill=c("red","green"))
```




```
# Epsilon = 0.5
plot.igraph(clean(graph.eps2), ylab = expression(epsilon == 0.5), vertex.color=V(graph.eps2)$color, vertex.label
 = NA)
legend("topright",
       c("right side","left side"),
       fill=c("red","green"))
```




```
# Epsilon = 0.65
plot.igraph(clean(graph.eps3), ylab = expression(epsilon == 0.65), vertex.color=V(graph.eps2)$color, vertex.label
= NA)
legend("topright",
       c("right side","left side"),
       fill=c("red","green"))
```



Graphically it's very cleat that the more epsilon grows while:

$$\alpha = 0.5$$

the less edges we have (since the threshold for the correlation grows). But there is another interesting result to analyze, most of the nodes that remain are the ones corrisponding to the **right** part of the brain.

Summary

```
recap <- matrix(c(sum(edge), sum(edge2), sum(edge3)), nrow = 1, ncol = 3)
colnames(recap) <- c("Eps = 0","Eps = 0.5","Eps = 0.65")
rownames(recap) <- c("Number of nodes")
recap
```

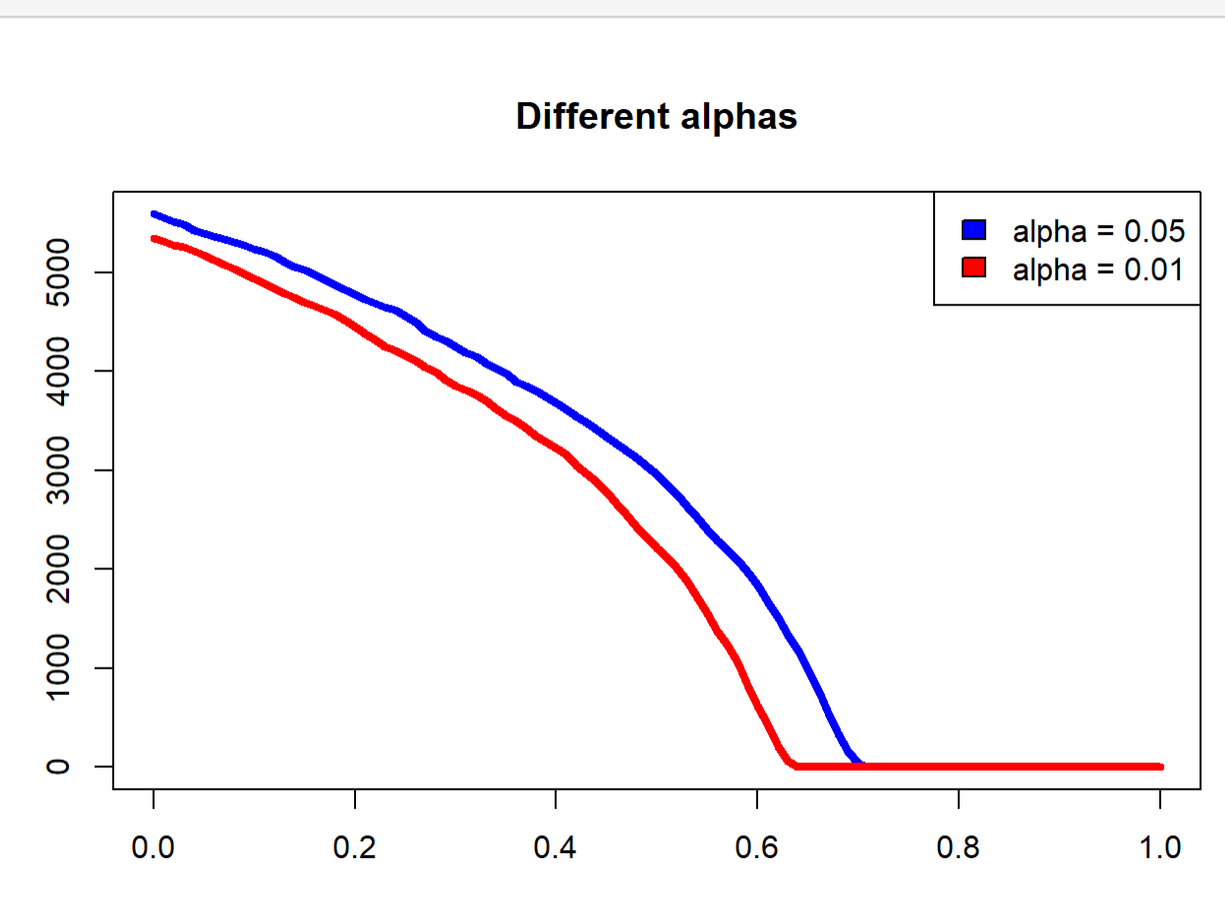
```
##                 Eps = 0 Eps = 0.5 Eps = 0.65
## Number of nodes    5592      2958        998
```

Relationship between epsilon and edges

```
# we repeat the same function for different values of epsilon
n.edges <- function(epsilon, n.features, l, u){
  e <- matrix(NA, n.features, n.features)
  for (i in 1:n.features){
    for(j in 1:n.features){
      # If 0 is not included in the CI it means there is an edge between the two nodes
      if (epsilon < l[i, j] | -epsilon > u[i, j]) e[i,j] = 1
      else e[i,j] = 0
      if (i == j) e[i,j] = 0 # self edge is excluded
    }
  }
  return (sum(e))
}

a <- seq(0,1, 0.01) # epsilon from 0 to 1

# alpha = 0.05
n.edges.vec <- list()
for (eps in a){
  n.edges1 <- n.edges(eps, n.features, ci.lower, ci.upper)
  n.edges.vec <- append(n.edges.vec, n.edges1)
}

# alpha = 0.01
n.edges.vec2 <- list()
for (eps in a){
  n.edges2 <- n.edges(eps, n.features, ci.lower2, ci.upper2)
  n.edges.vec2 <- append(n.edges.vec2, n.edges2)
}
```

Plot

```
par(mfrow=c(1,1))
plot(a, lwd = 4,  main="Different alphas", n.edges.vec, type = 'l',xlab = 'epsilon', ylab = 'n. edges', col = 'bl
ue')
lines(a, n.edges.vec2, col="red", lwd = 4)
legend("topright",
       c("alpha = 0.05","alpha = 0.01"),
       fill=c("blue","red"))
```



The number of edges decreases when epsilon increases, as we said before it means that less and less correlations are included in the confidence interval. This effect increses as alpha decreases, as we know **(1-alpha)** is the probability that a value falls into the confidence interval, the more alpha increases, the more likely it is this happens.

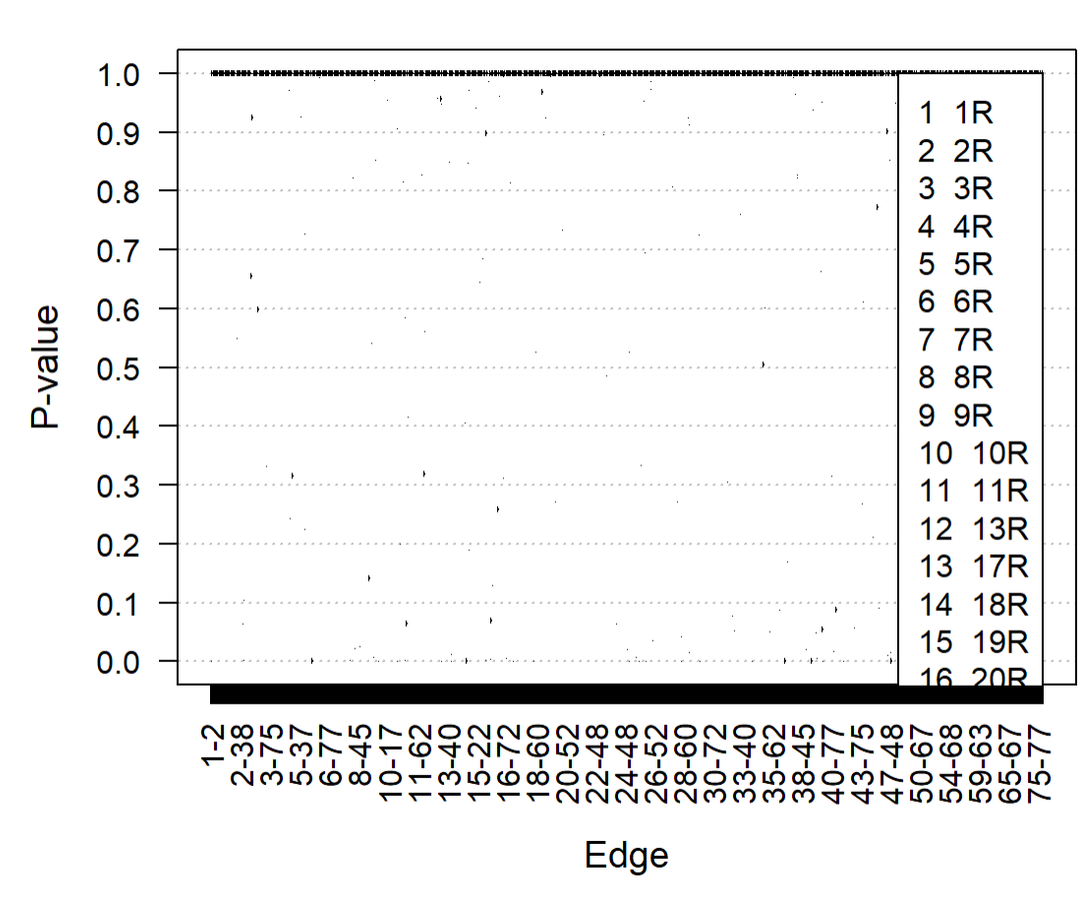
Analysis

```
# We are analyzing epsilon = 0 and alpha = 0.05
edgeR <- edge [1:39, 1:39] # we get all the right-right edges
edgeL <- edge [40:81, 40:81] # we get all the left-left edges
edgeLR <- edge [40:81, 1:39] # we get all the left-right edges

# We are analyzing epsilon = 0.5 and alpha = 0.05
edge2R <- edge2 [1:39, 1:39] # we get all the right-right edges
edge2L <- edge2 [40:81, 40:81] # we get all the left-left edges
edge2LR <- edge2 [40:81, 1:39] # we get all the left-right edges

relation <- matrix(c(sum(edgeR), sum(edgeL), sum(edgeLR), sum(edge2R), sum(edge2L), sum(edge2LR) ), nrow = 2, nco
l = 3)
colnames(relation) <- c("R-R","L-L","R-L")
rownames(relation) <- c("N. nodes, eps = 0", "N. nodes, eps = 0.5")
relation
```

```
##                      R-R  L-L R-L
## N. nodes, eps = 0   1226 1418 858
## N. nodes, eps = 0.5 1530  606 747
```

Partial Correlation

```
# Run & take a look
out <- sinUG(data.cor, n)
out <- round(out,3) # estimated partial correlation matrix
plotUGpvalues(out) # take a look at the p-values, they are almost all at 1
```

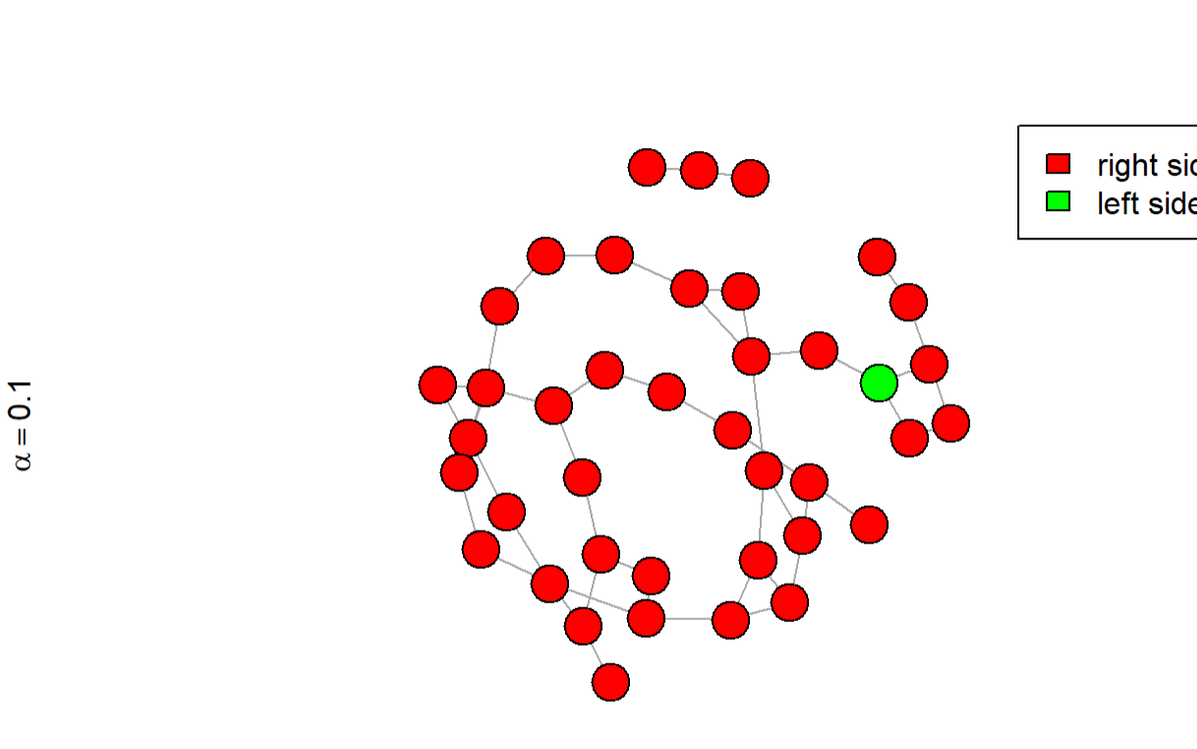



```
# the edges grow when alpha increases
par(mfrow = c(1,1))
alpha = 0.1
E.SIN = getgraph(out, alpha)
G.SIN = graph.adjacency(E.SIN, mode = "undirected")
```

```
# Change colours
V(G.SIN)$color[1:39] <- "red"
```

```
## Warning in vattrs[[name]][index] <- value: number of items to replace is not a
## multiple of replacement length
```

```
V(G.SIN)$color[40:81] <- "green"

plot.igraph(clean(G.SIN), ylab = expression(alpha == 0.1), vertex.color=V(G.SIN)$color, vertex.label = NA)
legend("topright",
       c("right side","left side"),
       fill=c("red","green"))
```



Using the partial correlation we can see that only one node corrisponding to the left side of the brain is connected with at least two other nodes (this is the reason why we are using the **clean** function).

```
# As we can see in this summary the number of edges grow when alpha grows
components(G.SIN, mode = c("weak", "strong"))
```

```
## $membership
##  1R  2R  3R  4R  5R  6R  7R  8R  9R 10R 11R 13R 17R 18R 19R 20R 21R 22R 23R 24R
##   1   1   2   3   4   1   5   1   1   1   6   7   1   1   1   6   8   9  10  11
## 25R 28R 29R 30R 31R 32R 34R 35R 36R 37R 38R 39R 40R 41R 42R 43R 44R 46R 47R  1L
##  12  13  14   1   1   1  15   1   1  16   8   1  17  18   1   1  19   1   1   1
##  2L  3L  4L  5L  6L  7L  8L  9L 10L 11L 13L 17L 18L 19L 20L 21L 22L 23L 24L 25L
##   1   1   1   1   1   1   1   1   1   1   9   1   1   1   1  20   1   1   1   1
## 27L 28L 29L 30L 31L 32L 33L 34L 35L 36L 37L 38L 39L 40L 41L 42L 43L 44L 45L 46L
##  21   6  22   1   1   1  23   6  24   6   1  14   1  17   1  25  26   1   1   1
## 47L
##   1
##
## $csize
##  [1] 48  1  1  1  1  5  1  2  2  1  1  1  1  2  1  1  2  1  1  1  1  1  1  1  1
## [26]  1
##
## $no
## [1] 26
```