Exercise 4

Group C

17/1/2022

Preparing the data

```
source('exercise4_da_sistemare.R')
library(org.Hs.eg.db)#Homo sapiens database call
library(GO.db)
```

We load the 'dati4.r' file containing selzionati, a vector with the names of genes selected as DE in exercise 3, and totali, a vector with the names of all genes remained after filtering in exercise 3

```
load('dati4.r')
```

We select from the Homo sapiens database the information relative to our genes of interest

```
#selection of information relative to our genes
gene_GO<-select(org.Hs.eg.db,keys=totali,columns=c('ENTREZID','GOALL'),keytype='SYMBOL')
#eliminate evidenceall column
gene_GO<-gene_GO[,-4]
#select lines that are repeated
doppi<-duplicated(gene_GO[,2:3])
#eliminate repeated lines
gene_GO<-gene_GO[doppi==FALSE,]
```

We obtain a dataframe and here we visualize the first rows

```
head(gene_GO)
```

```
##   SYMBOL ENTREZID      GOALL ONTOLOGYALL
## 1   RFC2     5982 GO:0000166          MF
## 2   RFC2     5982 GO:0003674          MF
## 6   RFC2     5982 GO:0003676          MF
## 7   RFC2     5982 GO:0003677          MF
## 8   RFC2     5982 GO:0003678          MF
## 9   RFC2     5982 GO:0003689          MF
```

We add a column to keep track if the gene belongs to the selected ones and we eliminate rows containing NAs

```
#add a column to track if a gene is part of the selected
SELECTED<-""
gene_GO<-cbind(gene_GO,SELECTED)
gene_GO$SELECTED<-gene_GO$SYMBOL%in%selezionati
#eliminate NAs
gene_GO_NA<-na.omit(gene_GO)
head(gene_GO_NA)
```

```
##   SYMBOL ENTREZID      GOALL ONTOLOGYALL SELECTED
## 1   RFC2     5982 GO:0000166          MF    FALSE
## 2   RFC2     5982 GO:0003674          MF    FALSE
## 6   RFC2     5982 GO:0003676          MF    FALSE
## 7   RFC2     5982 GO:0003677          MF    FALSE
## 8   RFC2     5982 GO:0003678          MF    FALSE
## 9   RFC2     5982 GO:0003689          MF    FALSE
```

We now split the dataframe in 3 different dataframes based on the ONTOLOGYALL denomination, we will run separate analysis for Molecular Function, Cellular Component and Biological Process

```
matriceMF<-gene_GO_NA[gene_GO_NA$ONTOLOGYALL=='MF',]
matriceCC<-gene_GO_NA[gene_GO_NA$ONTOLOGYALL=='CC',]
matriceBP<-gene_GO_NA[gene_GO_NA$ONTOLOGYALL=='BP',]
```

Enrichement Analysis

Molecular Function

We now do the step by step analysis for one of the three ontology classes (Molecular Function), for the other ones we will use functions we built and show the results.

Firstly we create a matrix containing, for each unique GO,the total number of occurrences and the number of occurrences in correspondence to the selected genes.

We select the unique GOs

```
GO_present<-duplicated(matriceMF$GOALL)
GO<-matriceMF$GOALL[GO_present==FALSE]
```

And we create the matrix of counts applying the function ‘assigned’ to each GO. The function ‘assigned’ counts the occurrences we want to know by looking in the dataframe

```
count<-sapply(GO,assigned,matriceMF$GOALL,matriceMF$SELECTED)
count<-t(count)
colnames(count)<-c("N_GENES","N_SELECTED")
```

We then remove the GOs for which we have just 1 total count and the ones for which we don’t have any selected gene occurrence

```
index_one<-which(count[,1]==1)
FINAL<-count[-index_one,]
FINAL<-FINAL[-which(FINAL[,2]<1),]
```

And we visualize the first rows of the matrix we obtained

```
head(FINAL)
```

```
##            N_GENES N_SELECTED
## GO:0000166    1906          9
## GO:0003674   14449        100
## GO:0003676    3379          5
## GO:0003677    2096          4
## GO:0003824    4672         29
## GO:0005488   13316         92
```

Now that we have our counts, we prepare for each G0 term the matrix that we need as input for the fisher exact test.

We count the number of genes remained in our dataframe after the GO selection and the number of selected genes remained.

```
gene_GO_NA<-matriceMF[matriceMF$GOALL%in%rownames(FINAL),]
gene_tot<-duplicated(matriceMF$SYMBOL)
ind_gene<-which(gene_tot==FALSE)
tot_genes<-length(ind_gene)
tot_selected<-length(which(matriceMF$SELECTED[ind_gene]==TRUE))
```

Then we create the input matrix in the following way for each GO remained, we see an example for the first one

```
tot_ac<-FINAL[1,1]
a<- FINAL[1,2]
c<- tot_ac-a
b<- tot_selected-a
d<- tot_genes-c-tot_selected
#create matrix
mat<-matrix(c(a,c,b,d),2,2)
rownames(mat)<-c('Selected','Non Selected')
colnames(mat)<-c('GO','not GO')
print(mat)
```

```
##                GO not GO
## Selected        9     91
## Non Selected 1897  12452
```

We apply to each GO remained the function we built 'fisher_test' that creates the input matrix and then applies the fisher exact test and extracts the pvalue. We obtain a list with the GOs and corresponding pvalues, we visualize the first elements.

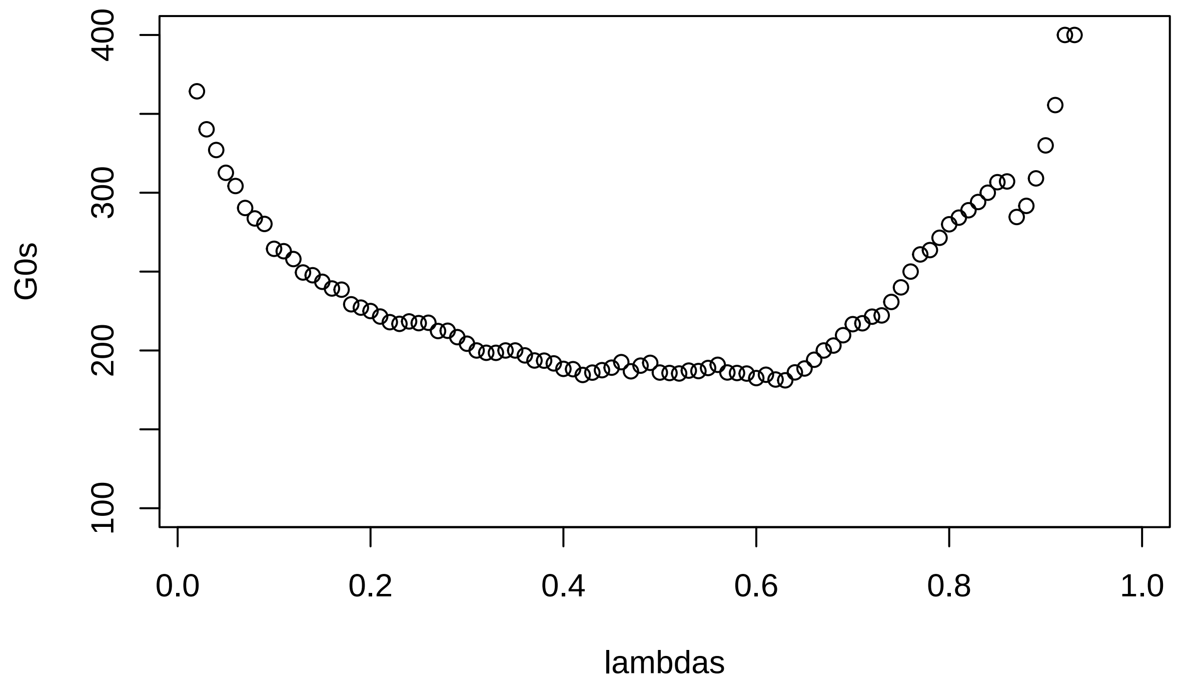
```
p_value<-apply(FINAL,1,fisher_test,tot_selected,tot_genes)
head(p_value)
```

```
## GO:0000166 GO:0003674 GO:0003676 GO:0003677 GO:0003824 GO:0005488
##  0.9242558  1.0000000  0.9999999  0.9998571  0.7934283  0.6157983
```

We now have to select the enriched GOs. To to this we decided to find a confidence level alpha for which the FDR is 0.05 and we select all the GOs with p_value less or equal than alpha.

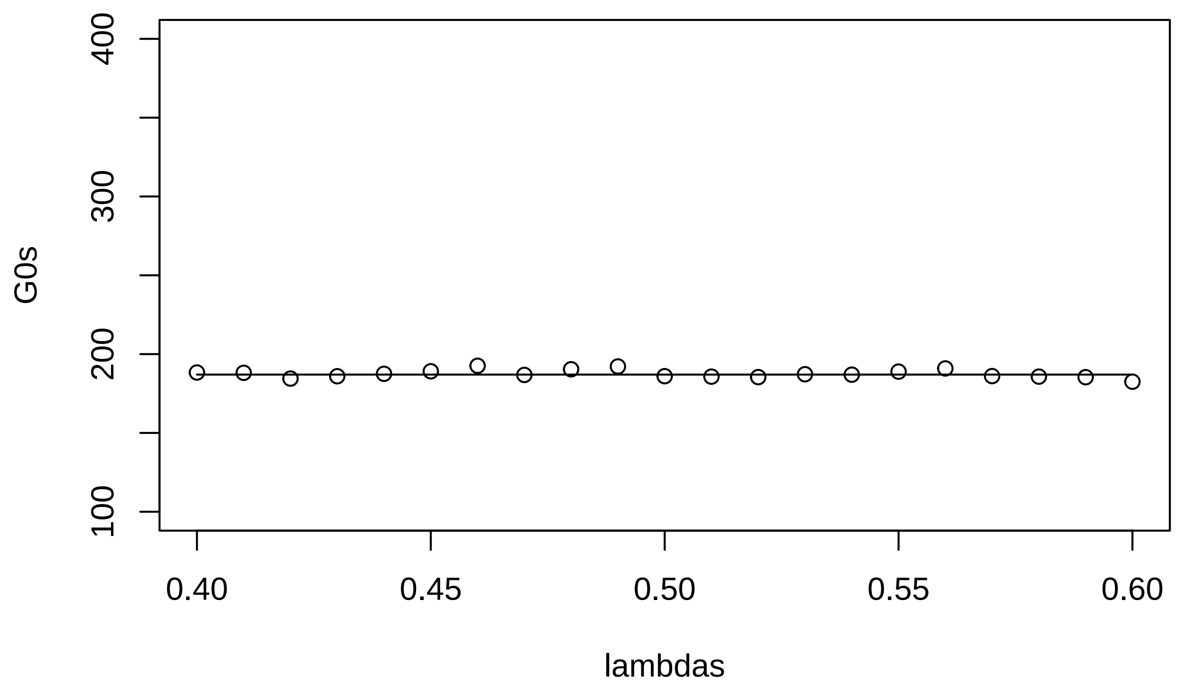
We apply the same procedure we used in exercise 3, so first of all we estimate the value of G0.

```
prima<-G0estimate(p_value,ylim1 = 100,ylim2=400, disegna=FALSE)
```



And we focus on the portion of graph where the estimate gets stable between 0.4 and 0.6

```
stima<-G0estimate(p_value,0.4,0.6,100,400)
```



Then we select the alpha that gives the FDR closest to 0.05

```
coppia<-select_alfa_FDR(p_value,stima,0.05)
names(coppia)<-c('alpha','FDR')
print(coppia)
```

```
##      alpha        FDR
## 0.01626077 0.05067939
```

```
alfa<-coppia[1]
```

And we select the G0s with pvalue smaller o equal than alpha

```
enrichedMF<-selectgenes(p_value,alfa)
```

We then associate to every GO the corresponding GO term coming from the GO.db database

```
descriptionMF<-select(GO.db,keys=enrichedMF,columns='TERM',keytype='GOID')
```

We obtain a final matrix containing the enriched GOs and the corresponding GO terms, of which we visualize the first rows

```
head(descriptionMF)
```

```
##         GOID                                    TERM
## 1 GO:0004888 transmembrane signaling receptor activity
## 2 GO:0038023              signaling receptor activity
## 3 GO:0060089            molecular transducer activity
## 4 GO:0030234                enzyme regulator activity
## 5 GO:0098772             molecular function regulator
## 6 GO:0044877        protein-containing complex binding
```

We now repeat the analysis for the other two terms.

Biological Process

For 'Biological Process' we obtain the following.(These are just the first rows)

```
pvaluesBP<-pvfisher(matriceBP)
enrichedBP<-enrichedGO(pvaluesBP,0.7,0.9,500,3000)
```

```
descriptionBP<-select(GO.db,keys=enrichedBP,columns='TERM',keytype='GOID')
head(descriptionBP)
```

```
##         GOID                                    TERM
## 1 GO:0006950                      response to stress
## 2 GO:0009893  positive regulation of metabolic process
## 3 GO:0043085 positive regulation of catalytic activity
## 4 GO:0044093 positive regulation of molecular function
## 5 GO:0048518 positive regulation of biological process
## 6 GO:0050789          regulation of biological process
```

Cellular Component

For 'Cellular Component' we obtain the following.(These are just the first rows)

```
pvaluesCC<-pvfisher(matriceCC)
enrichedCC<-enrichedGO(pvaluesCC,0.4,0.6,100,400)
```

```
descriptionCC<-select(GO.db,keys=enrichedCC,columns='TERM',keytype='GOID')
head(descriptionCC)
```

```
##         GOID                 TERM
## 1 GO:0005576 extracellular region
## 2 GO:0005615  extracellular space
## 3 GO:0005886      plasma membrane
## 4 GO:0012505  endomembrane system
## 5 GO:0016020             membrane
## 6 GO:0030141    secretory granule
```