PEC2

Yeison Santamaría

13/6/2020

EXTRACCION DE DATOS

library(readxl)  
setwd("~/Downloads/PEC2\_Yeison/pec2")

targets2 <- read\_excel("targets2.xlsx")

Grupos

NIT <- subset(targets2,grepl("^(NIT)", targets2$Group))  
SFI <- subset(targets2,grepl("^(SFI)", targets2$Group))  
ELI <- subset(targets2,grepl("^(ELI)", targets2$Group))

Extraccion de 10 de cada uno

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

set.seed(444)  
  
NIT\_D <- sample\_n(NIT, 10)  
SFI\_D <- sample\_n(SFI, 10)  
ELI\_D <- sample\_n(ELI, 10)

Carga de los datos extraidos

ELI\_10 <- read\_excel("ELI-10.xlsx")

NIT\_10 <- read\_excel("NIT-10.xlsx")

SFI\_10 <- read\_excel("SFI-10.xlsx")

column <- rbind(NIT\_10, SFI\_10, ELI\_10)  
grupos <- as.factor(column$Group)  
colNIT\_SFI <- rbind(NIT\_10,SFI\_10)  
grup1 <- factor(colNIT\_SFI$Group)  
colSFI\_ELI <- rbind(SFI\_10,ELI\_10)  
grup2 <- factor(colSFI\_ELI$Group)  
colNIT\_ELI <- rbind(NIT\_10,ELI\_10)  
grup3 <- factor(colNIT\_ELI$Group)

extraidos <- read\_excel("extraidos.xlsx")

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

library(tidyverse)

## -- Attaching packages ----------------------------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.1 v purrr 0.3.4  
## v tibble 3.0.1 v stringr 1.4.0  
## v tidyr 1.1.0 v forcats 0.5.0  
## v readr 1.3.1

## -- Conflicts -------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x lubridate::as.difftime() masks base::as.difftime()  
## x lubridate::date() masks base::date()  
## x dplyr::filter() masks stats::filter()  
## x lubridate::intersect() masks base::intersect()  
## x dplyr::lag() masks stats::lag()  
## x lubridate::setdiff() masks base::setdiff()  
## x lubridate::union() masks base::union()

basedatos <- extraidos %>%  
 remove\_rownames() %>%  
 column\_to\_rownames(var = 'GO')

dim(basedatos)

## [1] 56202 30

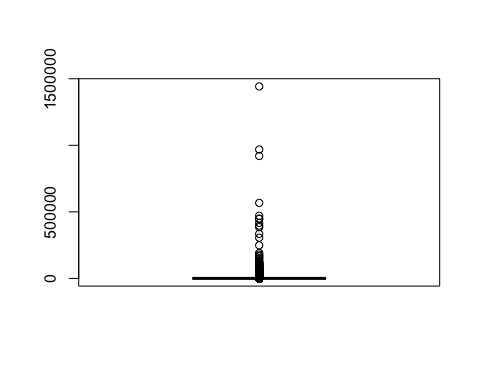
table(is.na(basedatos))

##   
## FALSE   
## 1686060

medigen<-apply(basedatos, 1, mean)  
table(medigen == 0)

##   
## FALSE TRUE   
## 46787 9415

boxplot(medigen)

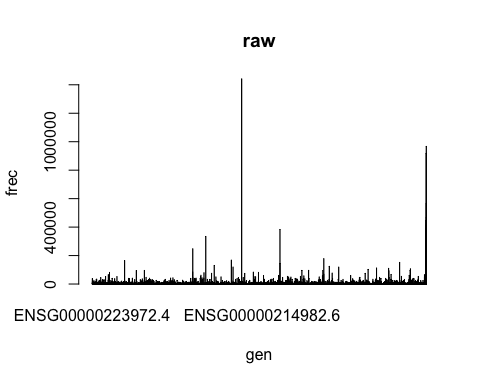


borrados<-basedatos[which(medigen ==0),]  
i<-intersect(rownames(borrados), rownames(basedatos))  
basedatos<-basedatos[!rownames(basedatos)%in% i,]

dim(basedatos)

## [1] 46787 30

medgenborra<-apply(basedatos, 1, mean)  
barplot(medgenborra,main = 'raw', xlim=NULL, xlab = 'gen', ylab='frec')



Normalización

library(tweeDEseq)  
normal <- normalizeCounts(basedatos)

## Using edgeR-TMM normalization.

## Calculating normalization factors with the TMM method.

## Estimating common dispersion.

## Estimating tagwise dispersions.

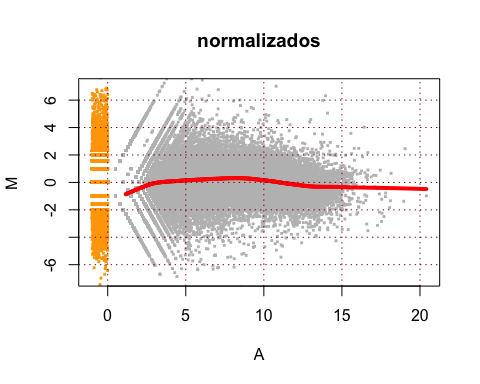
## Calculating effective library sizes.

## Adjusting counts to effective library sizes using tagwise dispersions.

library(edgeR)

## Loading required package: limma

library(limma)  
maPlot(normal[,1], normal[,2],  
 pch=15, cex=.4, ylim=c(-7,7),   
 allCol="grey", lowess=TRUE)  
grid(col="darkred")  
title("normalizados")



Identificación de genes diferencialmente expresados

gr1<- cbind(normal[,1:10],normal[,11:20])  
gr2 <- cbind(normal[,11:20],normal[,21:30])  
gr3 <- cbind(normal[,1:10],normal[,21:30])

NIT vs SFI

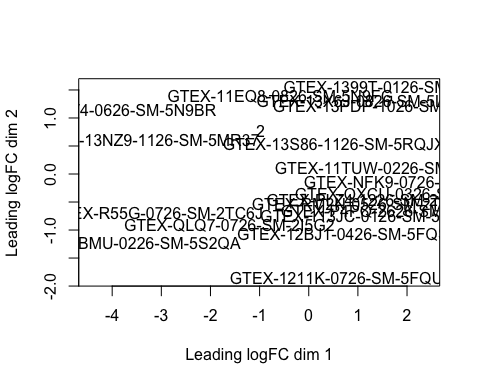
d <- DGEList(counts = gr1, group = grup1)  
d <- calcNormFactors(d)

## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced

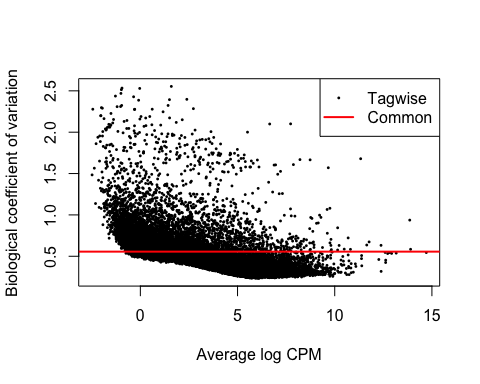
m <- sweep(d$counts, 2, 1e6 / d$samples$lib.size, '\*')  
ridx\_grupo1 <- rowSums(m>1) >= 2  
table(ridx\_grupo1)

## ridx\_grupo1  
## FALSE TRUE   
## 27764 19023

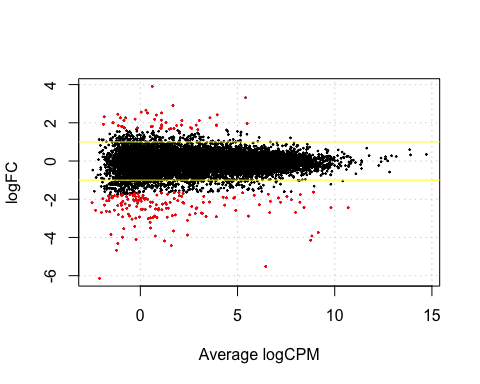
d <- d[ridx\_grupo1,]  
plotMDS(d)



d1 <- estimateCommonDisp(d)  
dtag1 <- estimateTagwiseDisp(d1)  
res.common1 <- exactTest(d1, pair=c("NIT", "SFI"), dispersion="common")  
res.tagwise1 <- exactTest(dtag1, pair=c("NIT", "SFI"), dispersion="tagwise")  
  
plotBCV(dtag1, cex=0.4)



dec1 <- decideTestsDGE(res.common1,p=0.001, adjust="BH")  
dtag\_grupo1 <- rownames(d1)[as.logical(dec1)]  
plotSmear(res.common1, de.tags = dtag\_grupo1)  
abline(h=c(-1,1),col="yellow")



d1\_df <- as.data.frame(dec1)  
d1\_df[,2] <- rownames(res.common1)  
up1 <- d1\_df[which(d1\_df$`SFI-NIT`==1),]  
down1 <- d1\_df[which(d1\_df$`SFI-NIT`==-1),]  
summary(dec1)

## SFI-NIT  
## Down 154  
## NotSig 18822  
## Up 47

gengrp1 <- c(up1$V2,down1$V2)

SFI vs ELI

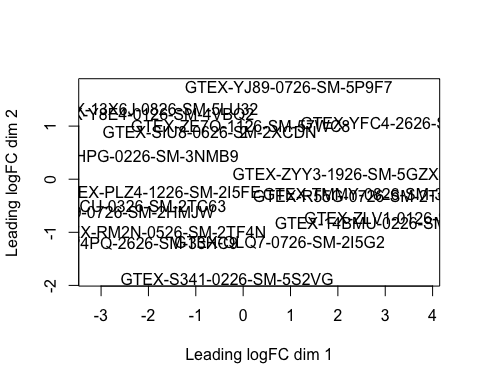
d <- DGEList(counts = gr2, group = grup2)  
d <- calcNormFactors(d)

## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced

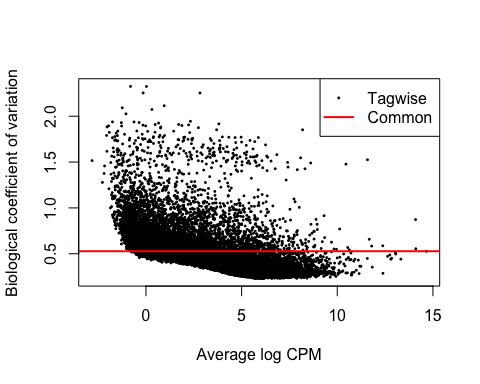
m <- sweep(d$counts, 2, 1e6 / d$samples$lib.size, '\*')  
ridx\_grupo2 <- rowSums(m>1) >= 2  
table(ridx\_grupo2)

## ridx\_grupo2  
## FALSE TRUE   
## 27665 19122

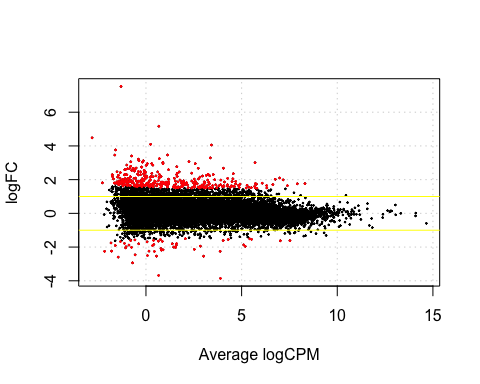
d <- d[ridx\_grupo2,]  
plotMDS(d)



d2 <- estimateCommonDisp(d)  
dtag2 <- estimateTagwiseDisp(d2)  
res.common2 <- exactTest(d2, pair=c("SFI", "ELI"), dispersion="common")  
res.tagwise2 <- exactTest(dtag2, pair=c("SFI", "ELI"), dispersion="tagwise")  
  
plotBCV(dtag2, cex=0.4)



dec2 <- decideTestsDGE(res.common2,p=0.001, adjust="BH")  
dtag\_segun <- rownames(d2)[as.logical(dec2)]  
plotSmear(res.common2, de.tags = dtag\_segun)  
abline(h=c(-1,1),col="yellow")



d2\_df <- as.data.frame(dec2)  
d2\_df[,2] <- rownames(res.common2)  
up2 <- d2\_df[which(d2\_df$`ELI-SFI`==1),]  
down2 <- d2\_df[which(d2\_df$`ELI-SFI`==-1),]  
summary(dec2)

## ELI-SFI  
## Down 50  
## NotSig 18733  
## Up 339

gengrp2 <- c(up2$V2,down2$V2)

NIT vs ELI

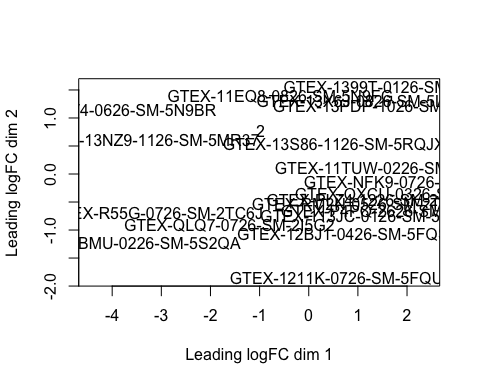
d <- DGEList(counts = gr1, group = grup3)  
d <- calcNormFactors(d)

## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced  
  
## Warning in .calcFactorTMM(obs = x[, i], ref = x[, refColumn], libsize.obs =  
## lib.size[i], : NaNs produced

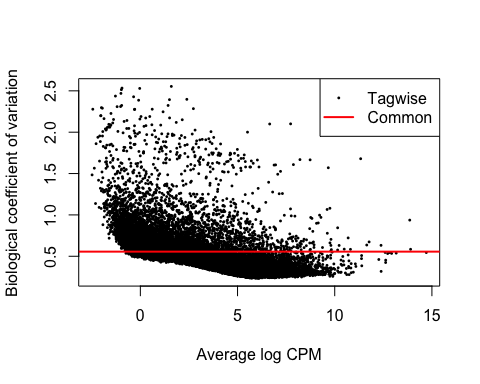
m <- sweep(d$counts, 2, 1e6 / d$samples$lib.size, '\*')  
ridx\_grupo3 <- rowSums(m>1) >= 2  
table(ridx\_grupo3)

## ridx\_grupo3  
## FALSE TRUE   
## 27764 19023

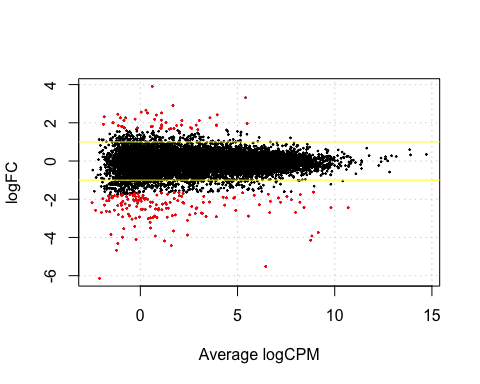
d <- d[ridx\_grupo3,]  
plotMDS(d)



d3 <- estimateCommonDisp(d)  
dtag3 <- estimateTagwiseDisp(d3)  
res.common3 <- exactTest(d3, pair=c("NIT", "ELI"), dispersion="common")  
res.tagwise3 <- exactTest(dtag3, pair=c("NIT", "ELI"), dispersion="tagwise")  
  
plotBCV(dtag3, cex=0.4)



dec3 <- decideTestsDGE(res.common3,p=0.001, adjust="BH")  
dtag\_terc <- rownames(d3)[as.logical(dec3)]  
plotSmear(res.common3, de.tags = dtag\_terc)  
abline(h=c(-1,1),col="yellow")



d3\_df <- as.data.frame(dec3)  
d3\_df[,2] <- rownames(res.common3)  
up3 <- d3\_df[which(d3\_df$`ELI-NIT`==1),]  
down3 <- d3\_df[which(d3\_df$`ELI-NIT`==-1),]  
summary(dec3)

## ELI-NIT  
## Down 154  
## NotSig 18822  
## Up 47

gengrp3 <- c(up3$V2,down3$V2)

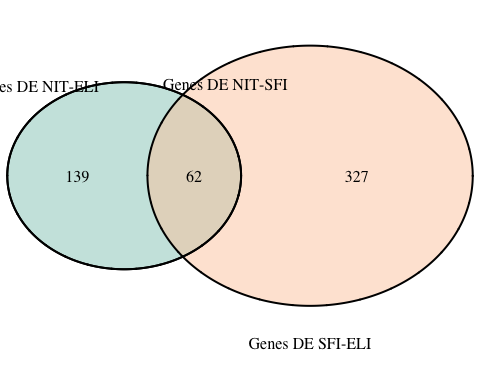
Genes comunes

library(VennDiagram)

## Loading required package: grid

## Loading required package: futile.logger

library(RColorBrewer)  
diagrama <- venn.diagram(x = list("Genes DE NIT-SFI" = gengrp1, "Genes DE SFI-ELI" = gengrp2,"Genes DE NIT-ELI" = gengrp3), fill = brewer.pal(3, "Pastel2"), filename = NULL)  
grid.draw(diagrama)



comunes <- intersect(intersect(gengrp1,gengrp2),gengrp3)  
comunes

## [1] "ENSG00000233750.3" "ENSG00000160808.5" "ENSG00000260265.1"   
## [4] "ENSG00000248923.1" "ENSG00000133710.11" "ENSG00000185303.11"  
## [7] "ENSG00000122852.10" "ENSG00000092054.12" "ENSG00000234648.1"   
## [10] "ENSG00000211939.2" "ENSG00000269154.1" "ENSG00000211639.2"   
## [13] "ENSG00000211672.2" "ENSG00000226958.1" "ENSG00000232177.1"   
## [16] "ENSG00000117215.10" "ENSG00000203814.5" "ENSG00000143536.7"   
## [19] "ENSG00000163209.10" "ENSG00000173110.6" "ENSG00000162897.10"  
## [22] "ENSG00000230937.5" "ENSG00000123689.5" "ENSG00000144045.9"   
## [25] "ENSG00000211619.2" "ENSG00000242534.2" "ENSG00000158050.4"   
## [28] "ENSG00000236841.3" "ENSG00000188282.8" "ENSG00000151790.4"   
## [31] "ENSG00000197409.6" "ENSG00000197846.3" "ENSG00000198518.5"   
## [34] "ENSG00000187990.4" "ENSG00000124575.5" "ENSG00000197459.2"   
## [37] "ENSG00000124635.7" "ENSG00000203813.4" "ENSG00000237988.2"   
## [40] "ENSG00000181126.9" "ENSG00000232810.3" "ENSG00000204389.8"   
## [43] "ENSG00000204388.5" "ENSG00000147465.7" "ENSG00000160882.7"   
## [46] "ENSG00000172724.7" "ENSG00000110680.8" "ENSG00000200879.1"   
## [49] "ENSG00000255733.1" "ENSG00000111537.4" "ENSG00000120659.10"  
## [52] "ENSG00000100604.8" "ENSG00000170054.10" "ENSG00000211893.3"   
## [55] "ENSG00000137868.14" "ENSG00000263934.2" "ENSG00000141682.11"  
## [58] "ENSG00000125657.3" "ENSG00000132002.3" "ENSG00000248099.3"   
## [61] "ENSG00000160224.12" "ENSG00000188263.6"

length(comunes)

## [1] 62

Anotación de resultados

library(biomaRt)  
mart <- useMart(biomart = "ensembl", dataset = "hsapiens\_gene\_ensembl")  
  
gen\_comunes <-getBM(attributes = c("hgnc\_symbol"), filters = "ensembl\_gene\_id\_version", values =comunes, mart = mart)

gen\_comunes <- gen\_comunes[,1]  
gen\_comunes

## [1] "CRNN" "SNORD14E" "IGLV4-60" "IGLV4-3" "MTND4P24" "CICP27"   
## [7] "IGKV2D-28" "MTND5P11" "LINC02562" "BNIP3P13"

Análisis de significancia biológica

total\_1<-getBM(attributes = c("hgnc\_symbol","go\_id"), filters = "ensembl\_gene\_id\_version", values =rownames(normal), mart = mart)

head(total\_1,15)

## hgnc\_symbol go\_id  
## 1 SNORA77   
## 2 RNU6-880P   
## 3 RNU6-828P   
## 4 SNORD46   
## 5 RNU4-61P   
## 6 RNU1-155P   
## 7 RNU6-1319P   
## 8 RN7SKP19   
## 9 RNU6-1199P   
## 10 RNU6-1062P   
## 11   
## 12 SNORA14B   
## 13 RNU6-1205P   
## 14 RN7SKP195   
## 15 SCARNA18B

excluidos <- which(total\_1$go\_id=="")  
dim(total\_1)

## [1] 17843 2

total\_1 <- total\_1[-excluidos,]  
dim(total\_1)

## [1] 8927 2

list\_genes <- unique(total\_1$hgnc\_symbol)  
lista <- list()  
for (i in list\_genes) {  
 lista[[i]] = total\_1[which(total\_1$hgnc\_symbol==i),]$go\_id  
}  
head(lista,2)

## $MIR215  
## [1] "GO:0005615" "GO:1903231" "GO:0035195" "GO:1903561" "GO:0045391"  
##   
## $MIR429  
## [1] "GO:1903231" "GO:0035195"

gen2 <- names(lista)  
compar <- factor(as.integer(gen2 %in% gen\_comunes))  
table(compar)

## compar  
## 0 1   
## 764 4

library(topGO)

## Loading required package: BiocGenerics

## Loading required package: parallel

##   
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':  
##   
## clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,  
## clusterExport, clusterMap, parApply, parCapply, parLapply,  
## parLapplyLB, parRapply, parSapply, parSapplyLB

## The following object is masked from 'package:limma':  
##   
## plotMA

## The following objects are masked from 'package:lubridate':  
##   
## intersect, setdiff, union

## The following objects are masked from 'package:dplyr':  
##   
## combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':  
##   
## IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':  
##   
## Filter, Find, Map, Position, Reduce, anyDuplicated, append,  
## as.data.frame, basename, cbind, colnames, dirname, do.call,  
## duplicated, eval, evalq, get, grep, grepl, intersect, is.unsorted,  
## lapply, mapply, match, mget, order, paste, pmax, pmax.int, pmin,  
## pmin.int, rank, rbind, rownames, sapply, setdiff, sort, table,  
## tapply, union, unique, unsplit, which, which.max, which.min

## Loading required package: graph

##   
## Attaching package: 'graph'

## The following object is masked from 'package:stringr':  
##   
## boundary

## Loading required package: Biobase

## Welcome to Bioconductor  
##   
## Vignettes contain introductory material; view with  
## 'browseVignettes()'. To cite Bioconductor, see  
## 'citation("Biobase")', and for packages 'citation("pkgname")'.

## Loading required package: GO.db

## Loading required package: AnnotationDbi

## Loading required package: stats4

## Loading required package: IRanges

## Loading required package: S4Vectors

##   
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:tidyr':  
##   
## expand

## The following objects are masked from 'package:lubridate':  
##   
## second, second<-

## The following objects are masked from 'package:dplyr':  
##   
## first, rename

## The following object is masked from 'package:base':  
##   
## expand.grid

##   
## Attaching package: 'IRanges'

## The following object is masked from 'package:purrr':  
##   
## reduce

## The following object is masked from 'package:lubridate':  
##   
## %within%

## The following objects are masked from 'package:dplyr':  
##   
## collapse, desc, slice

##   
## Attaching package: 'AnnotationDbi'

## The following object is masked from 'package:dplyr':  
##   
## select

##

## Loading required package: SparseM

##   
## Attaching package: 'SparseM'

## The following object is masked from 'package:base':  
##   
## backsolve

##   
## groupGOTerms: GOBPTerm, GOMFTerm, GOCCTerm environments built.

##   
## Attaching package: 'topGO'

## The following object is masked from 'package:IRanges':  
##   
## members

## The following object is masked from 'package:grid':  
##   
## depth

names(compar) <- gen2  
  
GO\_data <- new("topGOdata", ontology="BP", allGenes=compar,annot = annFUN.gene2GO, gene2GO = lista)

##   
## Building most specific GOs .....

## ( 1520 GO terms found. )

##   
## Build GO DAG topology ..........

## ( 4438 GO terms and 10237 relations. )

##   
## Annotating nodes ...............

## ( 683 genes annotated to the GO terms. )

resFisher = runTest(GO\_data, algorithm = 'classic', statistic = 'fisher')

##   
## -- Classic Algorithm --   
##   
## the algorithm is scoring 143 nontrivial nodes  
## parameters:   
## test statistic: fisher

resFisher

##   
## Description:   
## Ontology: BP   
## 'classic' algorithm with the 'fisher' test  
## 4438 GO terms scored: 2 terms with p < 0.01  
## Annotation data:  
## Annotated genes: 683   
## Significant genes: 4   
## Min. no. of genes annotated to a GO: 1   
## Nontrivial nodes: 143

Nodes = 20  
allRes = GenTable(GO\_data, classicFisher = resFisher, topNodes = Nodes)  
head(allRes)

## GO.ID Term Annotated Significant  
## 1 GO:0002377 immunoglobulin production 61 3  
## 2 GO:0002440 production of molecular mediator of immu... 63 3  
## 3 GO:0051092 positive regulation of NF-kappaB transcr... 2 1  
## 4 GO:0014066 regulation of phosphatidylinositol 3-kin... 3 1  
## 5 GO:0009408 response to heat 4 1  
## 6 GO:0014065 phosphatidylinositol 3-kinase signaling 4 1  
## Expected classicFisher  
## 1 0.36 0.0025  
## 2 0.37 0.0028  
## 3 0.01 0.0117  
## 4 0.02 0.0175  
## 5 0.02 0.0233  
## 6 0.02 0.0233

# Plots

plotEnrich = function(allRes, title){

# Plotting!

layout(t(1:2), widths = c(8,1))

par(mar=c(4, .5, .7, .7), oma = c(3, 15, 3, 4), las = 1)

rbPal = colorRampPalette(c('red', 'white', 'blue'))

pvalue = as.numeric(gsub("<", "", allRes$classicFisher))

max\_value = as.integer(max(-log(pvalue))) + 1

pv\_range = exp(-seq(max\_value, 0, -1))

allRes$Color = rbPal(max\_value) [cut(pvalue, pv\_range)]

o = order(allRes$Significant, decreasing = T)

barplot(allRes$Significant[o], names.arg = allRes$Term[o], las = 2, horiz = T, col = allRes$Color[o],

xlab = "Number of sequences", main = title, cex.names = 0.85)

image(0, seq(1, max\_value), t(seq\_along(seq(1, max\_value))), col = rev(rbPal(max\_value)), axes = F, ann = F)

pv\_label = exp(-seq(log(1), -log(min(pvalue)), l = 6))

pv\_label = formatC(pv\_label, format = "e", digits = 2)

axis(4, at = seq(1, max\_value, length = 6), labels = c(1, pv\_label[2:6]), cex.axis = 0.85)

title("p.value", cex.main = 0.6)

}

plotEnrich(allRes = allRes, title = 'Enrichment Analysis')

A screenshot of a cell phone

Description automatically generated