ADO_PEC2

Lorena Gallardo

Contents

Enlace al GitHub:https://github.com/LorenaGaMo/Gallardo_Lorena_ADO_PEC2.git

ABSTRACT

En esta PEC se quiere hacer un análisis de datos RNA-seq de treinta muestras a partir de un estudio con muestras de células del tejido tiroidal, a las que se les ha aplicado diferentes tratamientos. Estas treinta muestras están divididas en tres grupos de diez que corresponde a los tres tratamientos de infiltración que se llevó a cabo. Con el análisis de datos se buscaron las diferencias significativas de la expresión génica de los diferentes grupos.

OBJETIVOS

El objetivo de este trabajo es comparar los niveles de expresión génica de estos tres grupos de muestras de tejido tiroidal dos a dos para obtener un listado de genes que se expresan de manera diferencial tanto los sobre expresados como subexpresados. En este caso haremos un experimento de comparación de clases. Las tres comparaciones posibles: - NIT vs SFI: Para comparar el efecto del tratamiento SFI con no tratados. - NIT vs ELI: Para comparar el efecto del tratamiento ELI con no tratados. - ELI vs SFI: Para comparar los dos tratamientos.

MATERIAL Y MÉTODOS

Software y naturaleza de los datos

El software utilizado para realizar este estudio es el siguiente: R [1] (versión 3.6.1) con el interfaz RStudio [2] y los paquetes de Bioconductor Affy [3] (versión 3.10). Se ha utilizado el paquete `DESeq` para el filtraje de los datos y determinar cuantitativamente la diferencia de expresión en las diferentes comparaciones.Para realizar el análisis Over-Representation Analysis se ha utilizado el paquete `clusterProfiler` y la base de datos de anotaciones del genoma humano `org.Hs.eg.db`. Este trabajo se ha realizado a partir de unos datos extraídos del repositorio del proyecto Genotype-Tissue Expression (GTEx) ˆ [https://www.gtexportal.org/home/]. Estos datos se corresponden a un análisis de 292 muestras de tiroides en el que se comparan tres tipos de infiltraciones: • *Not infiltrated tissues* (NIT): 236 muestras • *Small focal infiltrados* (SFI): 42 muestras • *Extensive lymphoid infiltrados* (ELI): 14 muestras

Se trata de un archivo con los datos de conteos ya procesados `counts.csv` y de un archivo con los grupos y las covariables `targets.csv`.

Preparación de los datos

Antes de empezar con el analisi propiamente se debe preparar el entorno de trabajo con un sistema de archivos. El hecho de tener la información bien estructurada facilita el proceso del analisis. Es recomendable tener instaladas las librerias necesarias ya que el proceso puede ser largo. Para simplificar los cálculos y para igualar el número de muestras de los distintos grupos, en este trabajo se pidió utilizar 10 muestras aleatorias de cada grupo. En el que en primer lugar se separaron del archivo `targets.csv` las muestras según el grupo (ELI, SFI o NIT) y después con la función `sample` se aleatorizaron diez muestras de cada uno., las cuales se unieron en un mismo dataframe posteriormente. El siguiente paso fue seleccionar los contajes correspondientes a las muestras donde se utilizó la variable `Sample_Name` como filtro.

Para simplificar los cálculos y para igualar el número de muestras de los distintos grupos, en este trabajo se pidió utilizar 10 muestras aleatorias de cada grupo. En el que en primer lugar se separaron del archivo

[1] https://cran.r-project.org/index.html
[2] https://www.rstudio.com/
[3] https://www.bioconductor.org/

`targets.csv` las muestras según el grupo (ELI, SFI o NIT) y después con la función `sample` se aleatorizaron diez muestras de cada uno., las cuales se unieron en un mismo dataframe posteriormente. El siguiente paso fue seleccionar los contajes correspondientes a las muestras donde se utilizó el paquete `dplyr` quedándonos con las treinta columnas de `counts.csv` que nos interesaban. Pero previamente se cambiaron los guiones por puntos para que coincidan con los nombres de las muestras en el dataframe de counts .

Y finalmente se guardan los nuevos archivos en la carpeta `data` previamente creada.(script completo en el apéndice)

```
> setwd(".")
> dir.create("data")
> dir.create("results")
> dir.create("figuras")
```

Creación objeto *DESeqDataSet*

Se ha utilizado el paquete `DESeq2` de Bioconductor para realizar el análisis de expresión diferencial que se muestra a continuación. Pero para poder utilizar las funciones de este paquete previamente se debe creas un objeto `DESeqDataSet` con la función `DESeqDataSetFromMatrix`.

Filtraje

Para reducir la información del objeto y agilizar el analisis se realiza un filtraje para retirar las filas sin nigún conteo. El número de genes obtenidos tras este filtrado es 43108.

Visualizamos la distribución de los datos obtenidos mediante un boxplot.

Normalización

Como se ha podido observar los datos no presentan homocedasticidad por lo que se procede ha normalizar los mismos mediante la función `vst` que normaliza los datos siguiendo el concepto de de transformación estabilizadora de varianzas.Aunque menos sensible a los contajes ellevados es más rápida en el análisis.

```
                    NIT.1    NIT.2    NIT.3    NIT.4    NIT.5    NIT.6
ENSG00000223972.4 5.129926 4.337691 4.900096 4.859809 5.245206 4.675556
ENSG00000227232.4 9.068004 9.572418 9.278619 9.376187 9.582470 9.168862
ENSG00000243485.2 4.966908 4.658415 4.774423 4.764768 4.337691 4.337691
ENSG00000237613.2 4.737523 4.658415 4.694732 4.859809 4.628933 4.675556
ENSG00000268020.2 4.737523 4.337691 4.337691 4.337691 4.337691 4.814425
ENSG00000240361.1 4.826609 4.337691 4.337691 4.764768 4.628933 4.675556
                    NIT.7    NIT.8    NIT.9   NIT.10    SFI.1     SFI.2
ENSG00000223972.4 5.027751 5.180870 4.686134 4.337691 5.404307  4.652978
ENSG00000227232.4 9.417608 8.850558 9.840866 8.806092 8.739928 10.065610
ENSG00000243485.2 4.874222 4.337691 4.938332 4.848216 5.057582  4.782695
ENSG00000237613.2 4.337691 4.337691 4.938332 4.848216 4.849343  4.337691
ENSG00000268020.2 4.648644 4.719051 4.337691 4.337691 4.849343  4.782695
ENSG00000240361.1 4.776601 4.875468 4.337691 4.699624 4.337691  4.782695
                     SFI.3    SFI.4    SFI.5    SFI.6    SFI.7    SFI.8
ENSG00000223972.4  4.768306 5.123096 4.802294 4.337691 5.261688 4.963035
ENSG00000227232.4 10.064550 9.724159 9.855247 9.506836 9.183194 9.147431
ENSG00000243485.2  4.586921 4.692395 4.666922 4.337691 4.650369 4.337691
ENSG00000237613.2  4.586921 4.692395 4.991961 4.610393 4.650369 4.594651
```

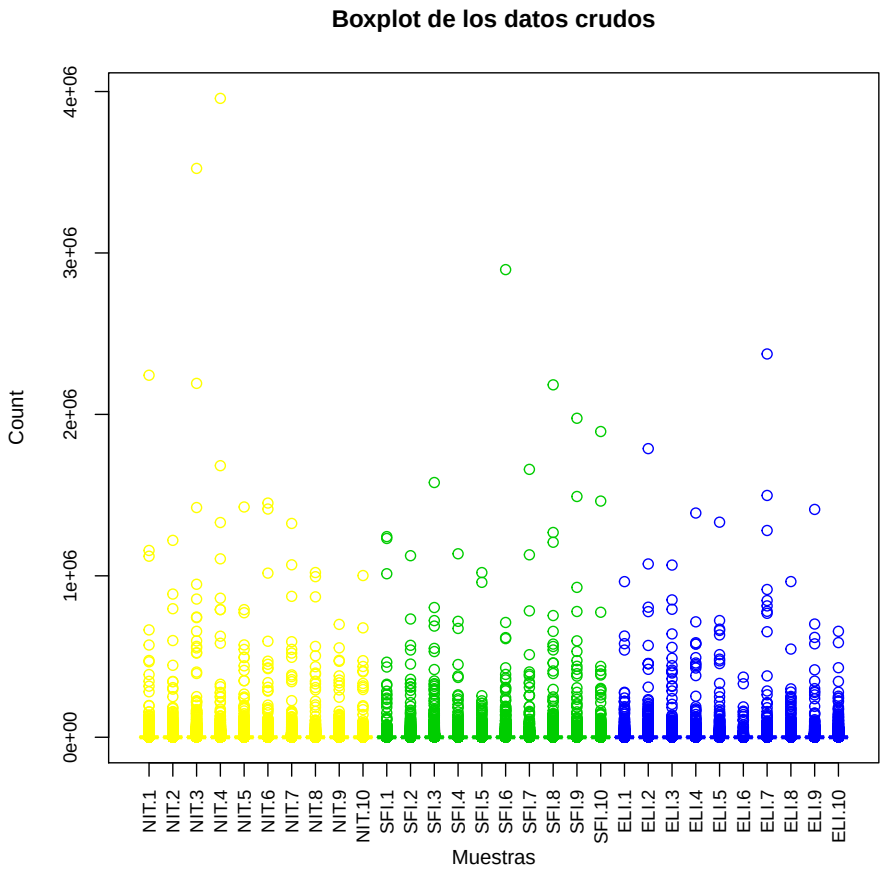


Figure 1: Boxplot de los datos crudos

```
ENSG00000268020.2  4.337691 4.337691 4.666922 4.337691 4.650369 4.337691
ENSG00000240361.1  4.337691 4.692395 4.666922 4.337691 4.337691 4.849600
                     SFI.9   SFI.10    ELI.1    ELI.2    ELI.3    ELI.4
ENSG00000223972.4 4.915214 4.907618 4.825180 4.845207 4.337691 4.337691
ENSG00000227232.4 8.982732 8.883243 9.056941 9.777117 8.903476 9.821670
ENSG00000243485.2 4.672602 4.994390 4.337691 4.592429 4.337691 4.665500
ENSG00000237613.2 4.672602 4.668160 4.337691 4.337691 4.665497 4.337691
ENSG00000268020.2 4.672602 4.337691 4.825180 4.697479 4.337691 4.337691
ENSG00000240361.1 4.810276 4.668160 4.683214 4.592429 4.337691 4.665500
                     ELI.5    ELI.6    ELI.7     ELI.8    ELI.9   ELI.10
ENSG00000223972.4  5.144354 5.313473 4.665555  4.863405 4.337691 4.337691
ENSG00000227232.4 10.135213 8.872573 9.068226 10.249573 9.905554 9.596292
ENSG00000243485.2  4.670578 4.908136 4.665555  4.642328 4.673741 4.611906
ENSG00000237613.2  4.807431 5.139335 4.337691  4.337691 4.673741 4.337691
ENSG00000268020.2  4.337691 4.908136 4.665555  4.337691 4.337691 4.337691
ENSG00000240361.1  4.670578 4.337691 4.665555  4.642328 4.337691 4.611906
```

Volvemos a genera un boxplot para visualizar de nuevo nuestros datos. Esta vez podemos observar que se ha ganado en hocedasticidad.

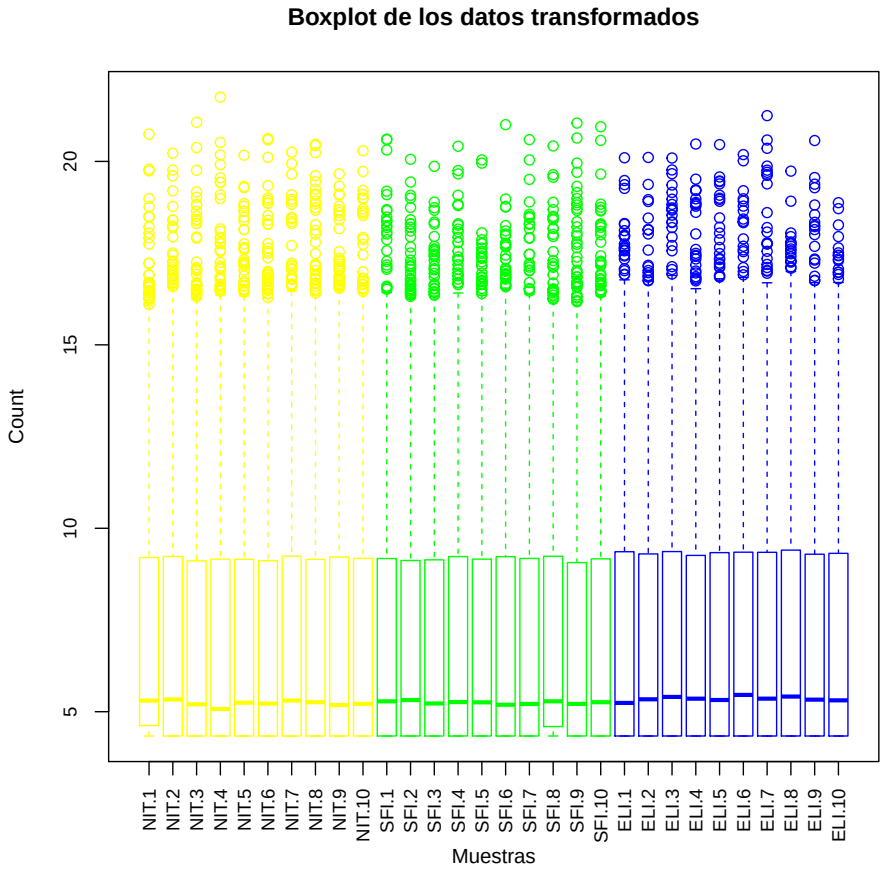


Figure 2: Boxplot de los datos transformados mediante la función 'vst'

Mediante el gráfico siguiente donde se muestra la desviación estándar en función de la media se puede corroborar que la desviación estándar es suficientemente baja y aceptar la transformación de los datos

realizada, a pesar de que haya outliners.

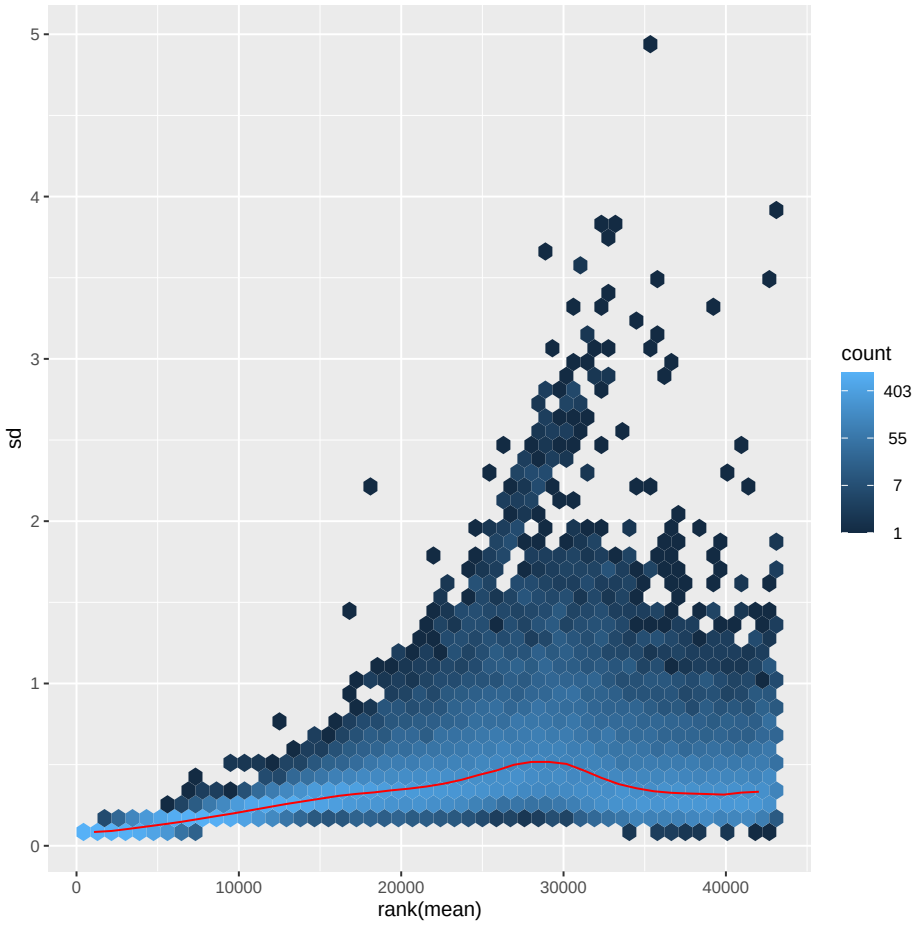


Figure 3: Gráfico de la desviación estandar en función de de la media

Clusterización de los datos

Se calcula la distancia euclídea entre las diferentes muestras con los datos transformados para un análisis de visualización. Con estas distancias se puede generar un mapa de calor que permite analizar si el diseño del experimento ha sido correcto.

En el mapa de calor se puede observar que las muestras NIT y las muestras ELI no estan relacionadas mientras que algunas muestras SFI estan agrupadas junto las NIT y dos de ellas (SFI.3 y SFI.8) se agrupan junto a las muestras ELI.

El gràfico de componentes principales corrobora los agrupamientos mencionados, NIT por un lado y ELI por otro con muestras de SFI repartida en ambos grupos. En este gráfico tambien podemos observar que el eje x explica el 59% de la variabilidad.

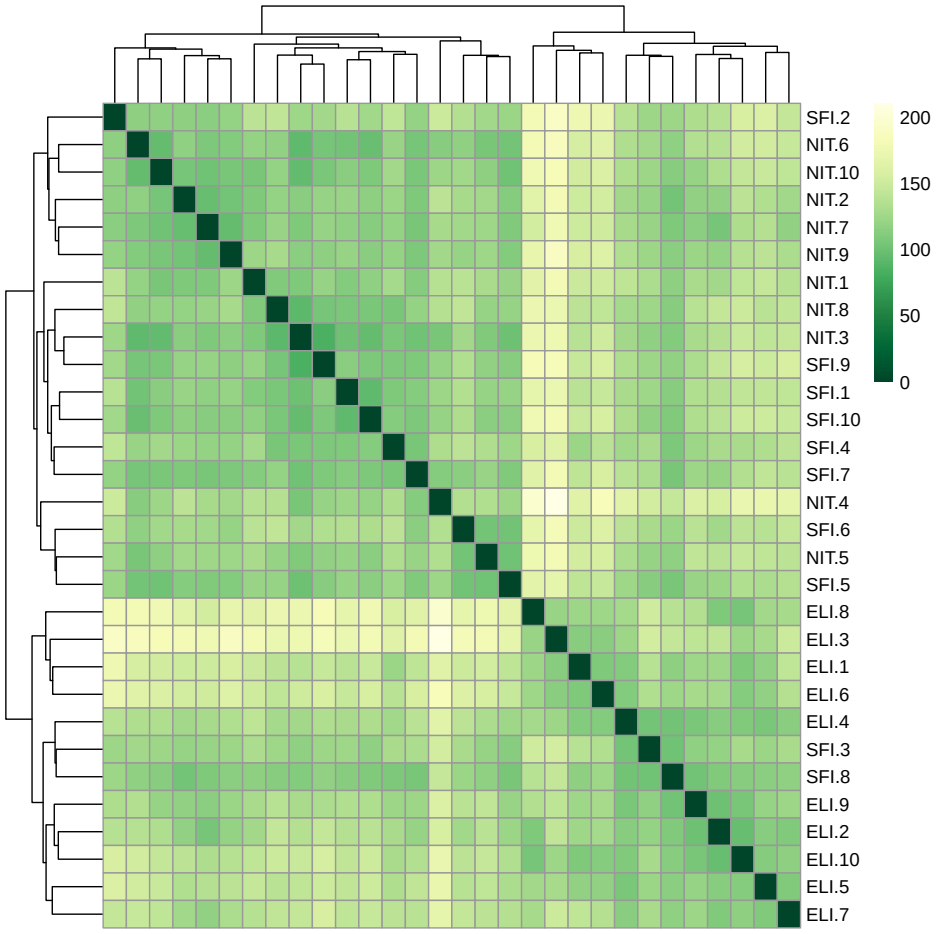


Figure 4: Heatmap de la similitudes entre las muestras

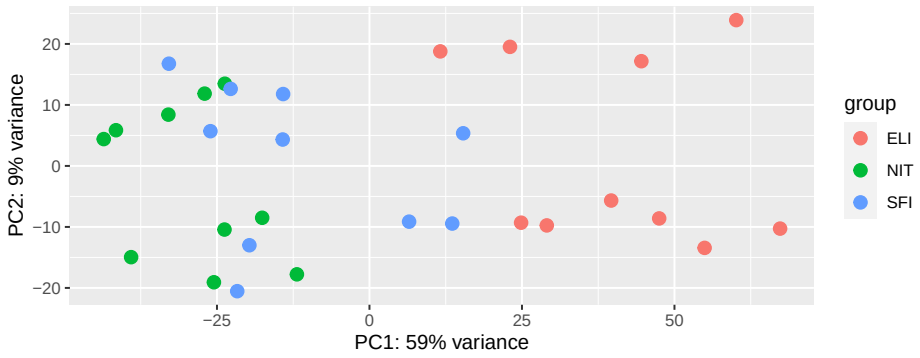


Figure 5: PCA de la similitudes entre las muestras

RESULTADOS y DISCUSIÓN

Análisis de expresión diferencial

En este paso se quiere determinar cuantitativamente la diferencia de experesión genica en las tres comparaciónes mediante la función `results`.

De los genes resultantes se filtraron quedandonos solo con los que tenian un p-valor < 0.01 y así resumir la cantidad de genes obtenidos. posteriormente se ordenaron y se obtuvieron los quince que se encontraban mas sobre expresados y los quince mas subextresados.

```
out of 43106 with nonzero total read count
adjusted p-value < 0.1
LFC > 0 (up)       : 33, 0.077%
LFC < 0 (down)     : 146, 0.34%
outliers [1]       : 0, 0%
low counts [2]     : 15045, 35%
(mean count < 2)
[1] see 'cooksCutoff' argument of ?results
[2] see 'independentFiltering' argument of ?results
```

```
out of 43106 with nonzero total read count
adjusted p-value < 0.1
LFC > 0 (up)       : 4092, 9.5%
LFC < 0 (down)     : 2457, 5.7%
outliers [1]       : 0, 0%
low counts [2]     : 12538, 29%
(mean count < 1)
[1] see 'cooksCutoff' argument of ?results
[2] see 'independentFiltering' argument of ?results
```

```
out of 43106 with nonzero total read count
adjusted p-value < 0.1
LFC > 0 (up)       : 2608, 6.1%
LFC < 0 (down)     : 4220, 9.8%
outliers [1]       : 0, 0%
low counts [2]     : 11702, 27%
(mean count < 1)
[1] see 'cooksCutoff' argument of ?results
[2] see 'independentFiltering' argument of ?results
```

Visualización de los genes mas diferenciados en su expresión

Para visualizar los agrupamientos sentre ciertos genes y ciertas muestras se realizo un mapa de calor y en esta se puede observar que se producen agrupamientos entre los genes sobre expresados y los subexpresados.

Se representan los resultados en un diagrama de Venn mediante la función`venn.diagram` del paquete `VennDiagram`para comprobar si los genes diferencialmente expresados que se obtienen mediante la función `results` son compartidos por más de una de las comparaciones.Se observa como la mayoría de genes que

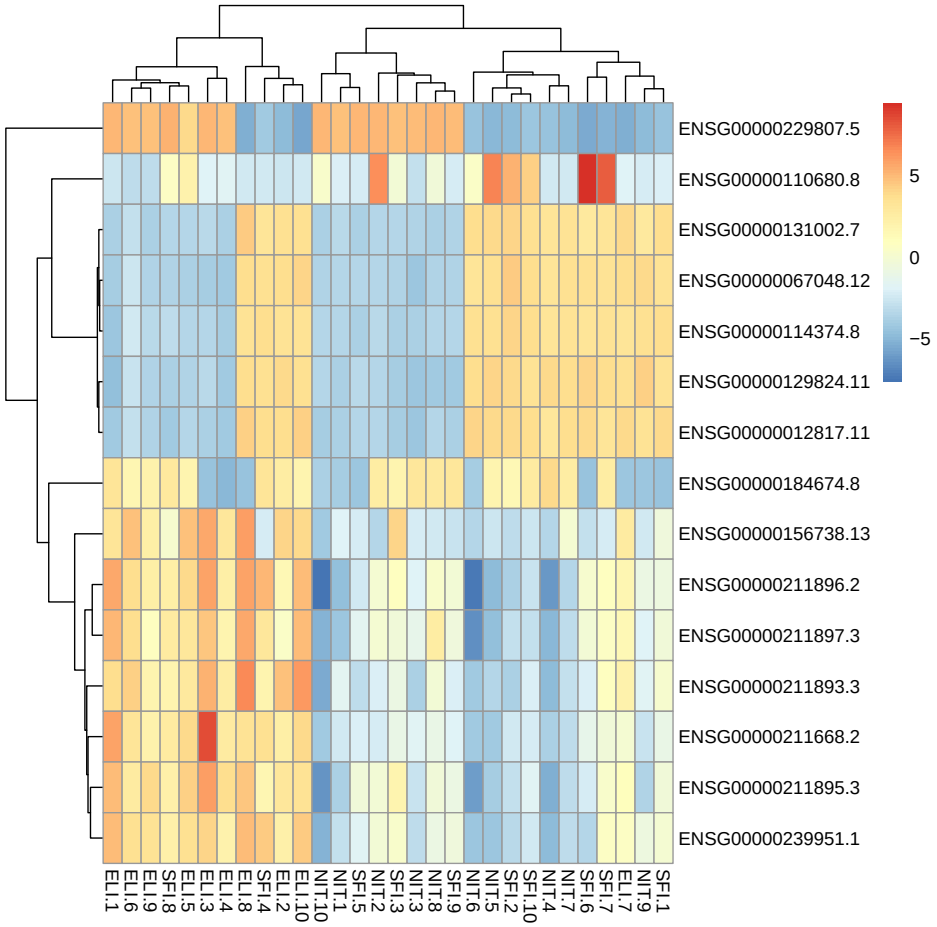


Figure 6: Heatmap de los genes mas diferenciados

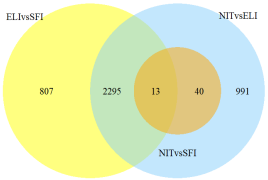


Figure 7: Diagrama de Venn de los genes diferenciados

están diferencialmente expresados en la comparación entre ELI y SFI también lo están entre ELI y NIT.Y que todos los que se exprresan diferencialmente en NIT y SFI estan incluidos en los de NIT y ELI.

En el gráfico MA se representa la media de los coeficientes en eje x (A) mientras que en el eje y (M) se representa la resta de los valores logarítmicos, lo que muestra la distribución de los coeficiente estimados en el modelo en todos los genes. Se observa que en la comparación SFI y ELI los genes estan sobre expresados en cambio en las otras dos comparaciones es la contrario estan subexpresados.

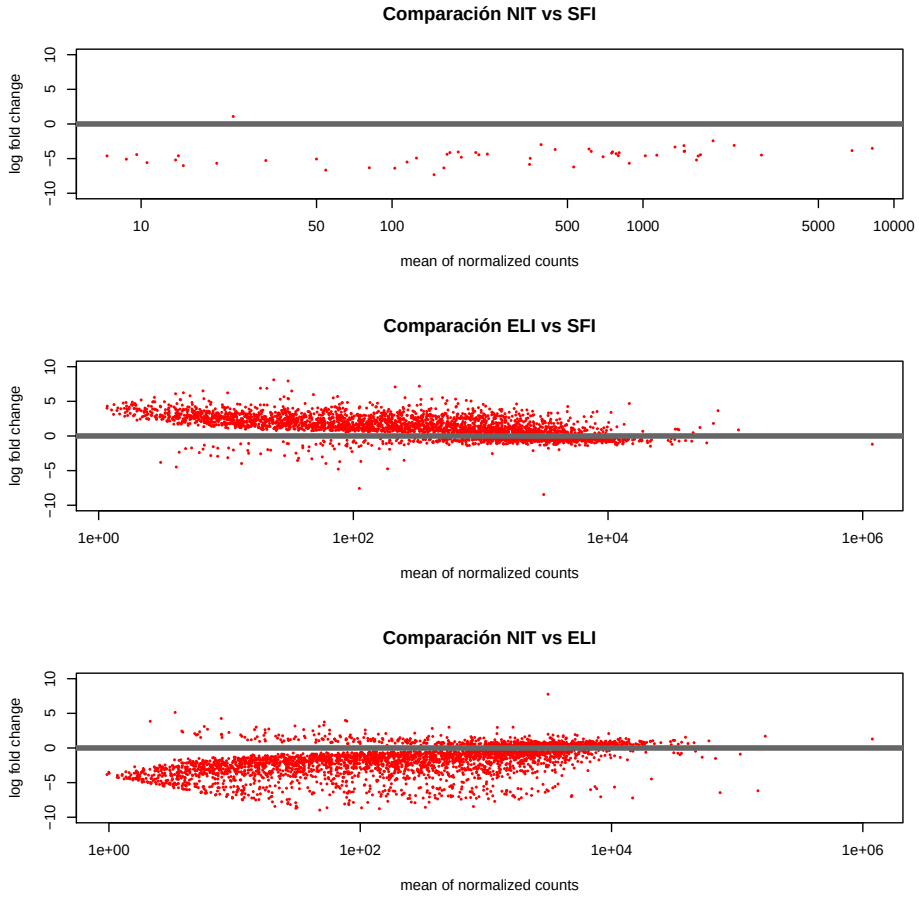


Figure 8: MAplot de las diferentes comparaciones

Anotación

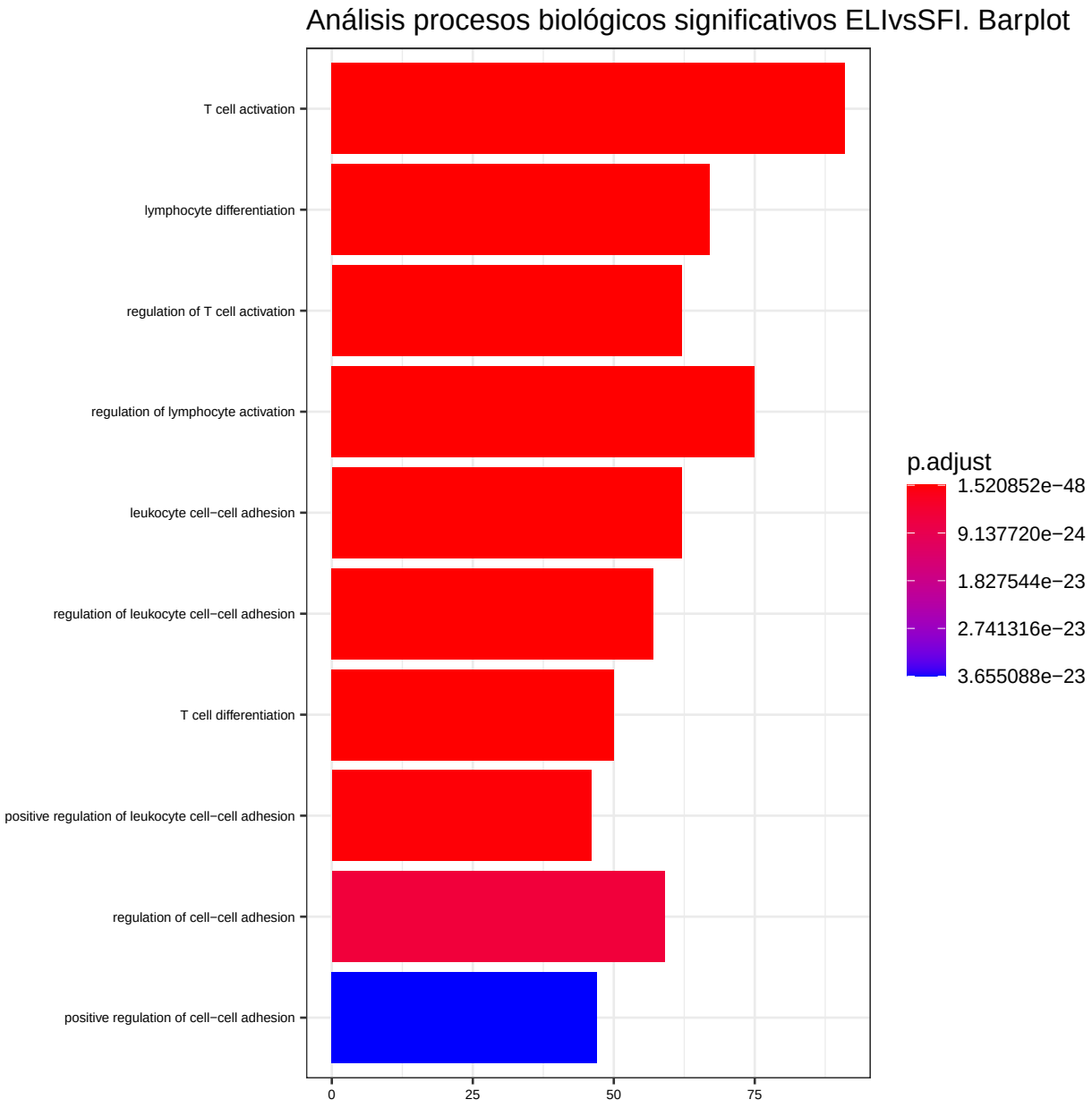
Para el análisis se han utilizado los identificadores de genes los d'ENSEMBL pero nos interesa añadir tambien los identificadores de SYMBIOL y d'ENTREZ. Para añadir esta información se utiliza las anotaciones de los paquetes `AnnotationDbi` y `org.Hs.eg.db` que se anotan en las tres tablas de resultados obtenidas de las tres comparaciones.

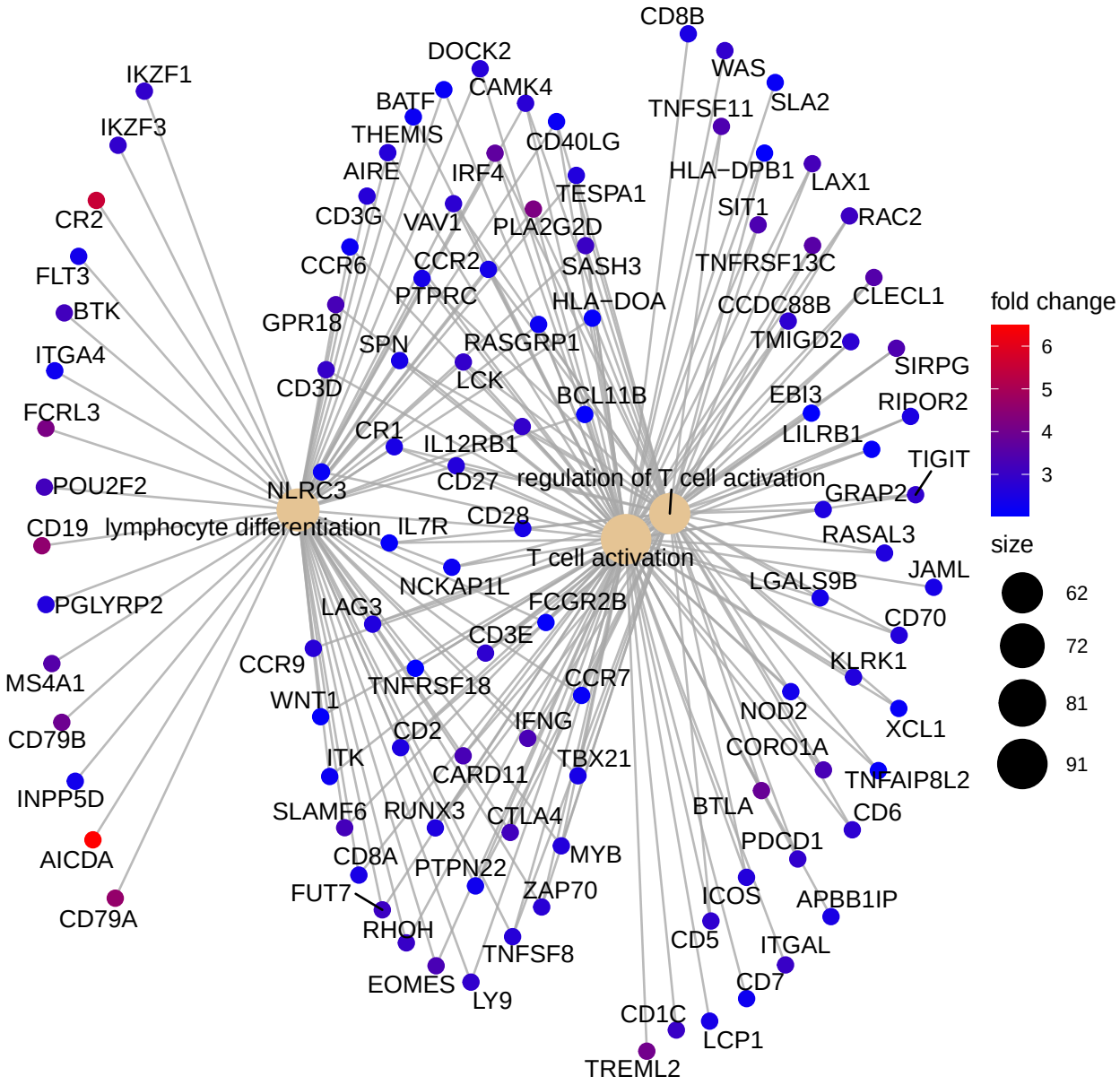
Análisis significación biológica

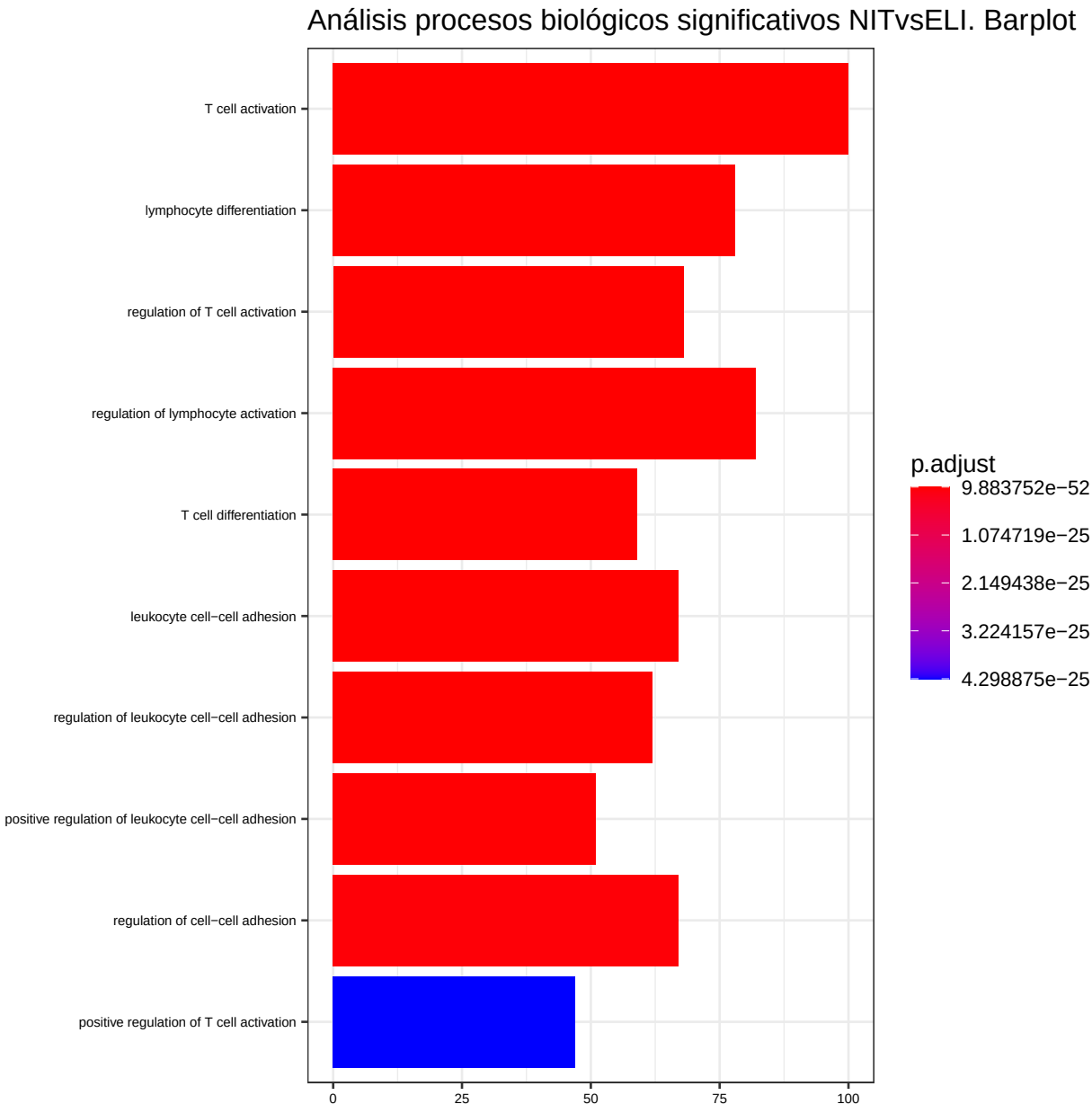
Se realiza un análisi de significación biologica con Over-Representation Analysis para poder ver que procesos biologicos estan involucrados.Este metodo nos permite comprovar los genes involucrados en determinados procesos biológicos. Se han seleccionado los que tienen un p-valor < 0.01 y un lofFC superior a 2 i asi reducir el numero de genes. Los resultados se han representado con las funciones `barploty` `cnetplot`.

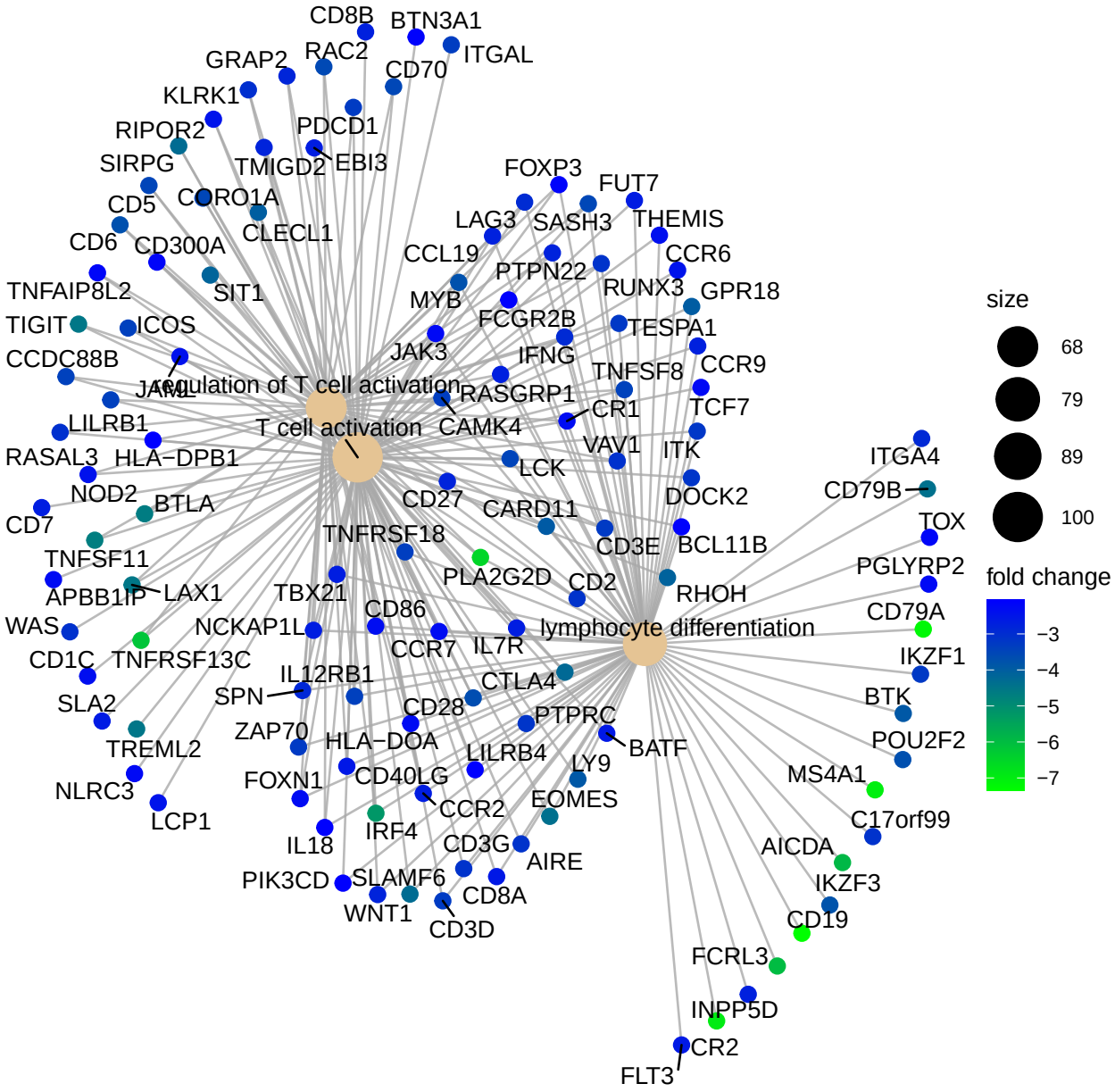
```
NITvsSFI ELIvsSFI NITvsELI
       6     2492     2459

NITvsSFI ELIvsSFI NITvsELI
       5      690      775
```

Se puede observar en los resultados del barplot que los genes que se expresan diferencialmente estan involucradis en procesos biologicos de defensa, donde se activan los linfocitos, celulas T y mecanismos del sistema immune.




CD8B
DOCK2
WAS
IKZF1
CAMK4
SLA2
BATF
TNFSF11
IKZF3
THEMIS
CD40LG
HLA−DPB1
LAX1
AIRE
IRF4
TESPA1
SIT1
CR2
CD3G
VAV1
PLA2G2D
RAC2
FLT3
CCR6
CCR2
SASH3
TNFRSF13C
BTK
PTPRC
HLA−DOA
CCDC88B
CLECL1
GPR18
SPN
RASGRP1
TMIGD2
ITGA4
SIRPG
CD3D
LCK
BCL11B
EBI3
RIPOR2
FCRL3
CR1
IL12RB1
LILRB1
TIGIT
POU2F2
NLRC3
CD27
regulation of T cell activation
GRAP2
CD19
lymphocyte differentiation
IL7R
CD28
RASAL3
T cell activation
JAML
NCKAP1L
LGALS9B
PGLYRP2
LAG3
FCGR2B
CD70
CD3E
CCR9
KLRK1
MS4A1
TNFRSF18
CCR7
WNT1
NOD2
XCL1
CD79B
CD2
IFNG
CORO1A
ITK
TBX21
CARD11
TNFAIP8L2
INPP5D
BTLA
SLAMF6
RUNX3
CTLA4
CD6
AICDA
CD8A
MYB
PDCD1
PTPN22
FUT7
ZAP70
ICOS
APBB1IP
CD79A
RHOH
CD5
TNFSF8
ITGAL
EOMES
LY9
CD7
CD1C
LCP1
TREML2
fold change
6
5
4
3
size
62
72
81
91


Análisis procesos biológicos significativos NITvsELI. Barplot
T cell activation
lymphocyte differentiation
regulation of T cell activation
regulation of lymphocyte activation
T cell differentiation
leukocyte cell−cell adhesion
regulation of leukocyte cell−cell adhesion
positive regulation of leukocyte cell−cell adhesion
regulation of cell−cell adhesion
positive regulation of T cell activation
0
25
50
75
100
p.adjust
9.883752e−52
1.074719e−25
2.149438e−25
3.224157e−25
4.298875e−25



Resumen de resultados

```
> Result_Files <- dir("./results/")
> knitr::kable(
+   Result_Files, booktabs = TRUE,
+   caption = "Listado de ficheros de resultados",
+   col.names="Ficheros"
+ )
```

Table 1: Listado de ficheros de resultados

Ficheros
EnrichGO_ELIvsSFI.csv
EnrichGO_NITvsELI.csv
EnrichGO_NITvsSFI.csv
ordRES.csv
ordRNE.csv
ordRNS.csv

APÉNDICE

```
> library(knitr)
> knitr::opts_chunk$set(echo = TRUE, message = FALSE, warning = FALSE,
+                       comment = NA, prompt = TRUE, tidy = FALSE,
+                       fig.width = 7, fig.height = 7, fig_caption = TRUE,
+                       cache=FALSE)
>
> # Librerias necesarias
> library("GenomicAlignments")
> library("BiocParallel")
> library("DESeq2")
> library("readxl")
> library("ggbeeswarm")
> library("genefilter")
> library("AnnotationDbi")
> library("org.Hs.eg.db")
> library("dplyr")
> library("ggplot2")
> library("vsn")
> library("pheatmap")
> library("RColorBrewer")
> library("VennDiagram")
> library("clusterProfiler")
>
> targets <- read.csv("./data/targets.csv")
> counts <- read.csv("./data/counts.csv", sep = ";")
>
> # Se comprueba si hay o no NA
> sum(is.na(targets))
> sum(is.na(counts))
>
> #Separamos por grupos el dataframe
> nit <- subset(targets, Group == "NIT")
> sfi <- subset(targets, Group == "SFI")
> eli <- subset(targets, Group == "ELI")
>
> #Selección al azar de 10 muestras de cada grupo y creación de un nuevo dataframe con la selección
> set.seed(2704)
> sel_nit <-nit [sample(nrow(nit), size = 10, replace = FALSE),]
> sel_sfi <- sfi[sample(nrow(sfi), size = 10, replace = FALSE),]
```

```
> sel_eli <- eli[sample(nrow(eli), size = 10, replace = FALSE),]
> sel_targets <- rbind(sel_nit, sel_sfi, sel_eli, deparse.level = 0)
>
> #Se cambian los guiones por puntos para que coincidan con los nombres de las muestras en el dataframe
>
> colnames(counts) <- gsub("\\.", "-", colnames(counts))
>
> nom_mostres <- sel_targets[,3]
>
> # en el dataframe de counts
> sel_counts <- dplyr::select(counts, nom_mostres)
> row.names(sel_counts) <- counts$X
>
> #Se definen nombres cortos a las columnas
>
> short_name <- c(paste("NIT", c(1:10), sep = "."), paste("SFI", c(1:10), sep = "."), paste("ELI", c(1:
> sel_targets <- cbind(sel_targets, short_name)
> #Se introducen en el dataframe sel_counts
> colnames(sel_counts) <- sel_targets$short_name
>
> #Se guardan los nuevos archivos:
>
>
> write.csv(sel_targets, file = "./data/selected_targets.csv")
> write.csv(sel_counts, file = "./data/selected_counts.csv")
>
> dds <- DESeqDataSetFromMatrix(countData = sel_counts, colData = sel_targets, design = ~ Group)
> dds
>
> # Filtraje
>
> dds <- dds[rowSums(counts(dds)) > 1, ]
>
> nrow(dds)
>
>
> boxplot(counts(dds), border = c(rep("yellow", 10), rep("green3", 10), rep("blue", 10)), ylab = "Count
>
> # Normalización
>
> vsd <- vst(dds, blind = FALSE)
> head(assay(vsd))
>
> boxplot(assay(vsd), border = c(rep("yellow", 10), rep("green", 10), rep("blue", 10)), ylab = "Count",
>
> meanSdPlot(assay(vsd))
>
> # Calculo de las distinas entre las muestras:
> distancia <- dist(t(assay(vsd)))
> dist_matrix <- as.matrix(distancia)
>
> colnames(dist_matrix) <- NULL
> colors <- colorRampPalette(rev(brewer.pal(9,"YlGn")))(200)
```

```
> pheatmap(dist_matrix, clustering_distance_rows = distancia,
+          clustering_distance_cols = distancia, col = colors)
>
> # Gráfico PCA :
> plotPCA(vsd, intgroup = c("Group"))
>
>
>
> #  Análisis de expresión diferencial
>
> dds <- DESeq(dds, parallel = TRUE)
>
>
>
> # Contraste entre las comparaciones de los grupos:
> resNitSfi <- results(dds, contrast = c("Group", "NIT", "SFI"))
> resEliSfi <- results(dds, contrast = c("Group", "ELI", "SFI"))
> resNitEli <- results(dds, contrast = c("Group", "NIT", "ELI"))
>
>
>
> # Resumen de las comparaciones
> summary(resNitSfi)
> summary(resEliSfi)
> summary(resNitEli)
>
> # Buscamos los 15 genes más sobreexpresados y los 15 genes más subexpresados en cada comparación
> adjRNS <- subset(resNitSfi, padj < 0.01)
> adjRES <- subset(resEliSfi, padj < 0.01)
> adjRNE <- subset(resNitEli, padj < 0.01)
> #Sobreesxpresados
> head(adjRNS[order(adjRNS$log2FoldChange),],15)
> head(adjRES[order(adjRES$log2FoldChange),],15)
> head(adjRNE[order(adjRNE$log2FoldChange),],15)
>
> #Subexpresados
> head(adjRNS[order(adjRNS$log2FoldChange, decreasing = T),],15)
> head(adjRES[order(adjRES$log2FoldChange, decreasing = T),],15)
> head(adjRNE[order(adjRNE$log2FoldChange, decreasing = T),],15)
>
> # Visualización de los genes mas diferenciados en su expresión
>
>
> dif_genes <- head(order(rowVars(assay(vsd)), decreasing = TRUE), 20)
> mdif_genes <- assay(vsd)[dif_genes, ]
> mdif_genes <- mdif_genes - rowMeans(mdif_genes)
> anno <- as.data.frame(colData(vsd)[,c("Group")])
> pheatmap(mdif_genes, annotacion_col = anno)
>
>
> genesRNS <- rownames(as.data.frame(adjRNS))
> genesRES <- rownames(as.data.frame(adjRES))
> genesRNE <- rownames(as.data.frame(adjRNE))
```

```
> venn.diagram(list(genesRNS, genesRES, genesRNE), category.names = c("NITvsSFI", "ELIvsSFI", "NITvsELI
>
>
> par(mfrow = c(3, 1))
> plotMA(adjRNS, alpha = 0.1, main = "Comparación NIT vs SFI",
+   xlab = "mean of normalized counts", ylim= c(-10,10), colNonSig = "gray60",
+   colSig = "red", colLine = "grey40",
+   MLE = FALSE)
> plotMA(adjRES, alpha = 0.1, main = "Comparación ELI vs SFI",
+   xlab = "mean of normalized counts", ylim= c(-10,10), colNonSig = "gray60",
+   colSig = "red", colLine = "grey40",
+   MLE = FALSE)
> plotMA(adjRNE, alpha = 0.1, main = "Comparación NIT vs ELI",
+   xlab = "mean of normalized counts", ylim= c(-10,10), colNonSig = "gray60",
+   colSig = "red", colLine = "grey40",
+   MLE = FALSE)
>
> # Anatoación
>
> adjRNS$symbol <- mapIds(org.Hs.eg.db, keys = substr(rownames(adjRNS), 1, 15), column = "SYMBOL", keyty
> adjRNS$entrez <- mapIds(org.Hs.eg.db, keys = substr(rownames(adjRNS), 1, 15), column = "ENTREZID", key
> adjRES$symbol <- mapIds(org.Hs.eg.db, keys = substr(rownames(adjRES), 1, 15), column = "SYMBOL", keyty
> adjRES$entrez <- mapIds(org.Hs.eg.db, keys = substr(rownames(adjRES), 1, 15), column = "ENTREZID", key
> adjRNE$symbol <- mapIds(org.Hs.eg.db, keys = substr(rownames(adjRNE), 1, 15), column = "SYMBOL", keyty
> adjRNE$entrez <- mapIds(org.Hs.eg.db, keys = substr(rownames(adjRNE), 1, 15), column = "ENTREZID", key
>
> # Se guardan ordenados
> ordRNS <- as.data.frame (adjRNS[order(adjRNS$pvalue),])
> ordRES <- as.data.frame (adjRES[order(adjRES$pvalue),])
> ordRNE <- as.data.frame (adjRNE[order(adjRNE$pvalue),])
> write.csv(ordRNS, file = "./results/ordRNS.csv")
> write.csv(ordRES, file = "./results/ordRES.csv")
> write.csv(ordRNE, file = "./results/ordRNE.csv")
>
>
> # Análisis significación biológica
>
>
> listOfTables <- list(adjRNS, adjRES, adjRNE)
> list_names <- list("NITvsSFI", "ELIvsSFI", "NITvsELI")
> selected_gens <- list()
>
> for (i in 1:length(listOfTables)) {
+   topTap <- listOfTables[[i]]
+   df <- as.data.frame(topTap)[order(topTap$log2FoldChange, decreasing = TRUE), c(2, 7)]
+   df <- na.omit(df)
+   list_gens <- df[, 1]
+   names(list_gens) <- df[, 2]
+   selected_gens[[i]] <- list_gens
+   names(selected_gens)[i] <- list_names[i]
+ }
> sapply(selected_gens, length)
>
```

```
>
> sig_gens <- list()
>
> for (i in 1:length(listOfTables)) {
+   topTap <- listOfTables[[i]]
+   df <- as.data.frame(topTap)[order(topTap$log2FoldChange, decreasing = TRUE), c(2, 7)]
+   df <- na.omit(df)
+    sigTab <- subset(topTap, topTap$padj < 0.01 & abs(topTap$log2FoldChange) > 2)
+   entrezid <- sigTab$entrez
+   sig_gens[[i]] <- as.character(entrezid[!is.na(entrezid)])
+   names(sig_gens)[i] <- list_names[i]
+ }
>
> sapply(sig_gens, length)
>
>
>
>
> mapped_genes2GO <- mappedkeys(org.Hs.egGO)
> mapped_genes2KEGG <- mappedkeys(org.Hs.egPATH)
> mapped_genes <- union(mapped_genes2GO, mapped_genes2KEGG)
>
>
>
> universe <- mapped_genes
> for (i in 1:length(sig_gens)){
+   gens <- sig_gens[[i]]
+   comparison <- names(sig_gens[i])
+   eGO <- enrichGO(gene = gens, universe = universe, OrgDb = org.Hs.eg.db, ont = "BP", pAdjustMethod =
+
+
+   if (nrow(eGO) != 0) {
+     write.csv(as.data.frame(eGO), file = paste0("./results/EnrichGO_", comparison, ".csv"), row.names
+
+
+     print(barplot(eGO, showCategory = 10, font.size = 6, title = paste0("Análisis procesos biológicos
+
+
+     print(cnetplot(eGO, categorySize = "geneNum",font.size=6, showCategory = 3, foldChange = selected
+   }
+ }
```

REFERENCIAS

- Analyzing RNA-seq data with DESeq2,Michael I. Love, Simon Anders, and Wolfgang Huber, https://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html#theory
- RNA-seq workflow: gene-level exploratory analysis and differential expression, Michael I. Love1,, Simon Anders, Vladislav Kim and Wolfgang Huber, https://www.bioconductor.org/packages/devel/workflows/vignettes/rnaseqGene/inst/doc/rnaseqGene.html}
- MA-Plot From Base Means And Log Fold Changes, https://www.rdocumentation.org/packages/DESeq2/versions/1.12.3/topics/plotMA}