PEC2 Análisis de datos ómicos

Esteban Hernández Maldonado

10 de junio de 2020

Contents

Todo el material de esta práctica está disponible en el repositorio de Github ubicado en https://github.com/ehm2411/Hernandez_Esteban_ADO_PEC2

1 Resumen

Estudio de ultrasecuenciación de 30 muestras de tiroides.

2 Objetivos

Ilustrar el proceso de análisis de datos de ultrasecuenciación mediante la realización de un análisis del tiroides en donde ser comparan tres tipos de infiltración medido en 30 muestras de un total de 292 pertenencientes a 3 grupos: NIT (Not infiltrated tissues) con 236 muestras, SFI (Small focal intiltrates) con 42 muestras y ELI (Extensive lymphoid infiltrates) con 14 muestras.

3 Materiales y métodos

3.1 Naturaleza de los datos, tipo de experimiento

Se dispone de un fichero de puntaje y uno de descripción de las diferentes muestras. Se hace una elección aleatoria de 10 muestras por cada grupo.

3.2 Métodos y herramientas utilizadas

Se ha seguido el **workflow** habitual del paquete DESeq2 para los estudios de ultrasecuenciación.

Herramientas:
Bioconductor/R
RStudio Markdown paquete DESeq2

4 Resultados

4.1 Definición de los datos y preprocesado (filtraje y normalización)

El fichero de puntajes original tiene información de 292 muestras y 56202 genes. Se ha hecho una selección aleatoria de 10 muestras por cada grupo y se han descartado el resto. Para contruir el objeto *dds* de clase DESeq2 mediante la función *DESeqDataSetFromMatrix* se han considerado todas las filas (genes).

Es importante recalcar que se ha tenido especial cuidado en asegurarnos de que los nombres de las filas del fichero coldata coindicen con el nombre de las columnas del fichero de puntaje. De no ser así, la utilización del fichero de tipo DESeq2 daría resultados falsos.

Una vez creado el objeto dds se procede a hacer un filtrado, como vemos a continuación, seleccionando los genes que tienen almenos un puntaje.

```
keep <- rowSums(counts(dds)) > 1
dds <- dds[keep,]
```

Se ha pasado por tanto de 56202 genes a 43573.

Ahora se tienen que normalizar los datos estabilizando la varianza con la función *vst*. Se descarta el uso de *rlog* debido al gran número de genes que tiene el fichero. Esta normalización hará que métodos exploratorios como el clustering y PCA funcionen mejor.

```
vsd <- vst(dds, blind = FALSE)
head(assay(vsd)[,1:2], 3)  # datos de solo dos muestras
```

```
##                    GTEX.QEL4.0726.SM.3GIJ5 GTEX.132AR.1126.SM.5P9GA
## ENSG00000223972.4                 5.427589                 4.810478
## ENSG00000227232.4                 9.493836                 9.541032
## ENSG00000243485.2                 5.248491                 5.047739
```

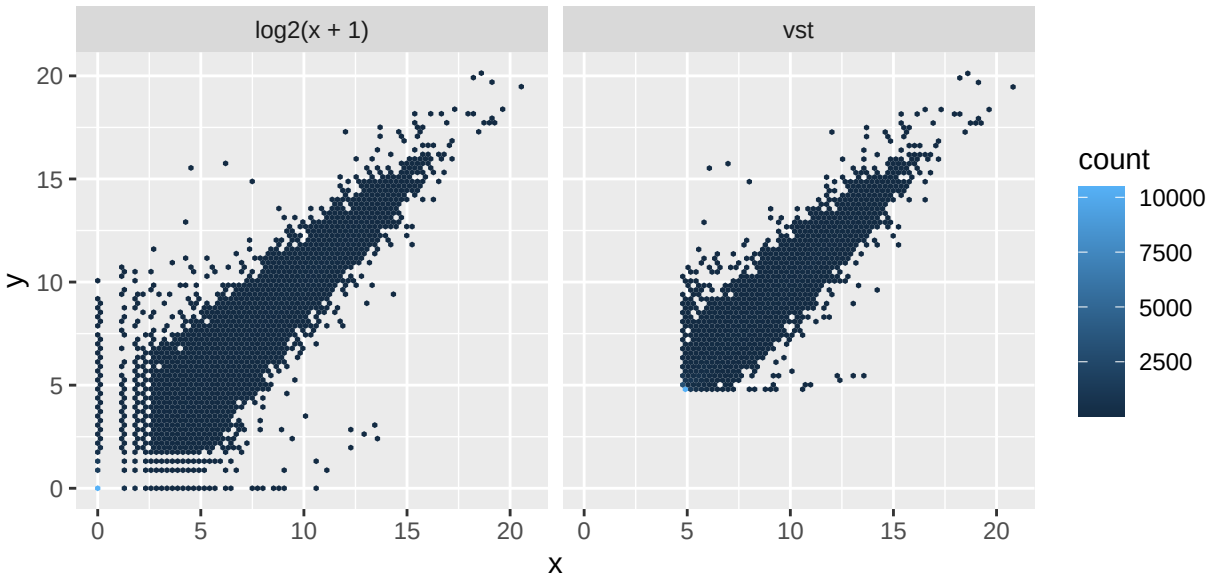
```
colData(vsd)[,c(1:2,7:8,10)]
```

```
## DataFrame with 30 rows and 5 columns
```

```
##                          Experiment SRA_Sample      sex    Group       alias
##                            <factor>   <factor> <factor> <factor> <character>
## GTEX.QEL4.0726.SM.3GIJ5   SRX222777  SRS389744     male      NIT        NIT1
## GTEX.132AR.1126.SM.5P9GA  SRX604580  SRS639146   female      NIT        NIT2
## GTEX.ZDYS.0626.SM.5J2N5   SRX605489  SRS639266     male      NIT        NIT3
## GTEX.ZTPG.0826.SM.5DUVC   SRX576876  SRS629440   female      NIT        NIT4
## GTEX.ZTX8.0626.SM.59HKC   SRX625353  SRS646077     male      NIT        NIT5
## ...                             ...        ...      ...      ...         ...
## GTEX.11XUK.0226.SM.5EQLW  SRX619829  SRS644736   female      ELI        ELI6
## GTEX.ZYY3.1926.SM.5GZXS   SRX568364  SRS627095   female      ELI        ELI7
## GTEX.14ABY.0926.SM.5Q5DY  SRX575932  SRS629299     male      ELI        ELI8
## GTEX.13QJC.0826.SM.5RQKC  SRX601511  SRS638114   female      ELI        ELI9
## GTEX.YJ89.0726.SM.5P9F7   SRX583148  SRS631283     male      ELI       ELI10
```

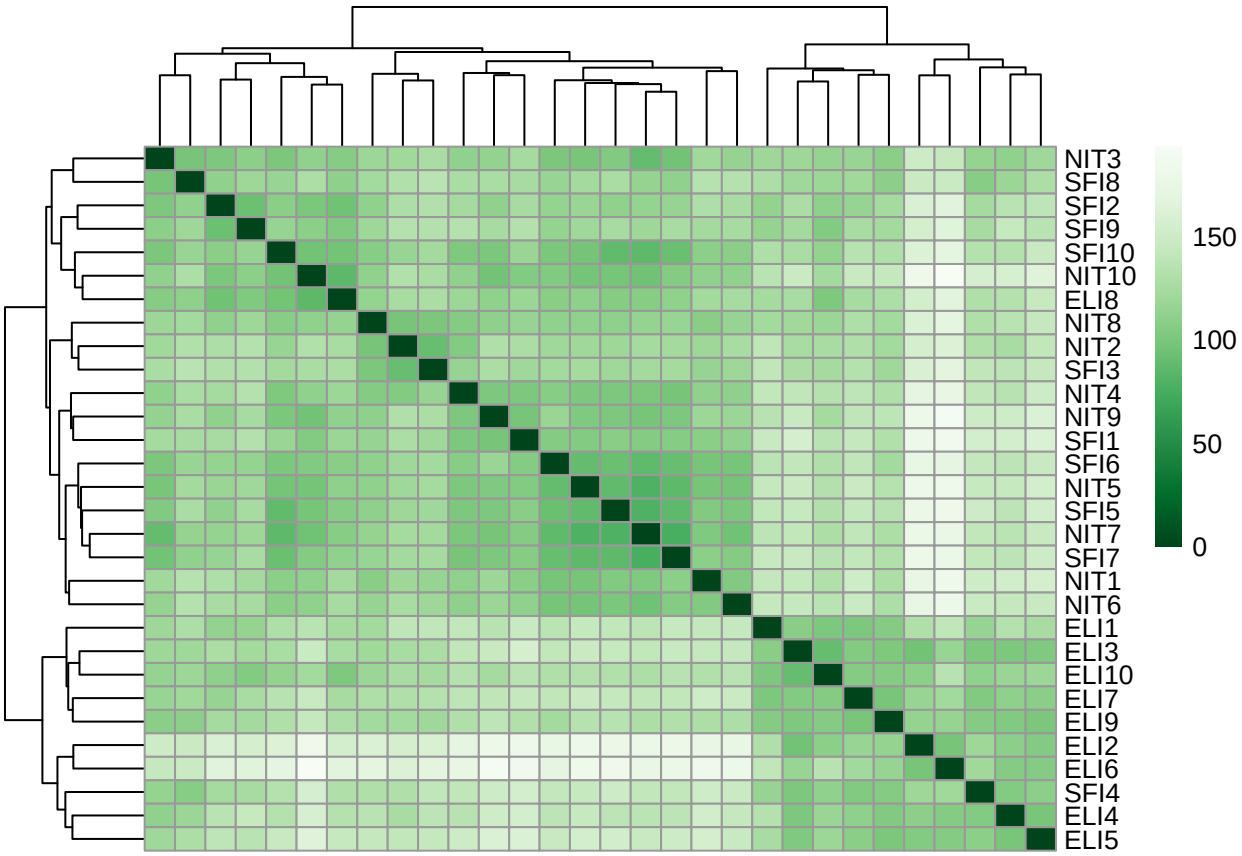
Si visualizamos los datos transformados por la función *vst* podemos ver un ligera mejora en la dispersión de los genes con puntaje muy bajo, ubicados en la esquina inferior izquierda. No obstante no se consigue una reducción sustancial.

```
library("dplyr")
library("ggplot2")

dds <- estimateSizeFactors(dds)

df <- bind_rows(
  as_data_frame(log2(counts(dds, normalized=TRUE)[, 1:2]+1)) %>%
         mutate(transformation = "log2(x + 1)"),
  as_data_frame(assay(vsd)[, 1:2]) %>% mutate(transformation = "vst"))

colnames(df)[1:2] <- c("x", "y")

ggplot(df, aes(x = x, y = y)) + geom_hex(bins = 80) +
  coord_fixed() + facet_grid( . ~ transformation)
```



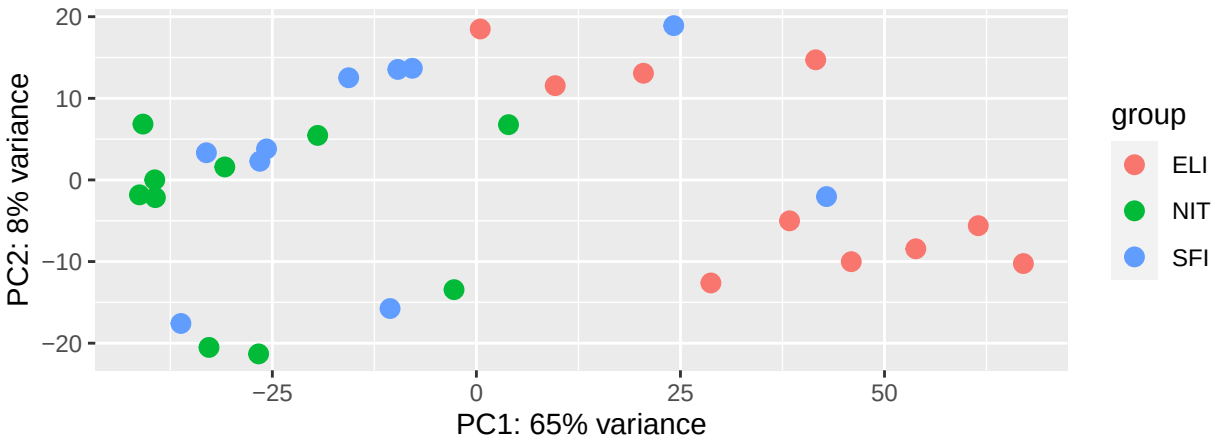
Se calcula ahora las distancias entre muestras para poder crear los Heatmaps.

```
sampleDists <- dist(t(assay(vsd)))

library("pheatmap")
library("RColorBrewer")

sampleDistMatrix <- as.matrix( sampleDists )
rownames(sampleDistMatrix) <- vsd$alias
colnames(sampleDistMatrix) <- NULL
colors <- colorRampPalette( rev(brewer.pal(9, "Greens")) )(255)
pheatmap(sampleDistMatrix,
         clustering_distance_rows = sampleDists,
         clustering_distance_cols = sampleDists,
         col = colors)
```



Y por último se crea el gráfico PCA.

```
plotPCA(vsd, intgroup  = c("Group"))
```



4.2 Identificación de los genes diferencialmente expresados.

Aplicamos la función DESeq para detectar los genes diferencialmente expresados y calculamos el conjunto de resultados para cada comparación de dos grupos: NIT-SFI, NIT-ELI y SFI-ELI.

```
dds <- DESeq(dds, parallel =TRUE)
```

```
resNITSFI <- results(dds, contrast=c("Group","NIT","SFI"))
round_df(as.data.frame(resNITSFI)[1:5,c(1:4,6)],6)
```

```
##                     baseMean log2FoldChange    lfcSE      stat     padj
## ENSG00000223972.4   2.558911      -0.130386 0.722859 -0.180376 0.999856
## ENSG00000227232.4 650.020574       0.205602 0.209761  0.980174 0.992577
## ENSG00000243485.2   1.395690      -0.272321 0.683942 -0.398163       NA
## ENSG00000237613.2   1.545823       0.638594 0.787246  0.811175       NA
## ENSG00000268020.2   0.877952       0.379479 1.109062  0.342162       NA
```

```
resNITELI <- results(dds, contrast=c("Group","NIT","ELI"))
round_df(as.data.frame(resNITELI)[1:5,c(1:4,6)],6)
```

```
##                     baseMean log2FoldChange    lfcSE      stat     padj
## ENSG00000223972.4   2.558911       0.307259 0.746107  0.411816 0.837309
## ENSG00000227232.4 650.020574      -0.026109 0.209767 -0.124466 0.956006
```

```
## ENSG00000243485.2   1.395690       0.738958 0.761595  0.970276 0.569131
## ENSG00000237613.2   1.545823       2.023002 0.908008  2.227956 0.113529
## ENSG00000268020.2   0.877952       0.719051 1.134752  0.633663       NA
```

```
resSFIELI <- results(dds, contrast=c("Group","SFI","ELI"))
round_df(as.data.frame(resSFIELI)[1:5,c(1:4,6)],6)
```

```
##                     baseMean log2FoldChange    lfcSE      stat     padj
## ENSG00000223972.4   2.558911       0.437645 0.734491  0.595848 0.775306
## ENSG00000227232.4 650.020574      -0.231711 0.209758 -1.104659 0.541955
## ENSG00000243485.2   1.395690       1.011279 0.736588  1.372923       NA
## ENSG00000237613.2   1.545823       1.384408 0.929490  1.489428 0.371277
## ENSG00000268020.2   0.877952       0.339572 1.151384  0.294925       NA
```

```
table(resNITSFI$padj < 0.05)
```

```
##
## FALSE  TRUE
## 28332    35
```

```
table(resNITELI$padj < 0.05)
```

```
##
## FALSE  TRUE
## 26007  4894
```

```
table(resSFIELI$padj < 0.05)
```

```
##
## FALSE  TRUE
## 25762  3450
```

```
# genes con un p valor ajustado < 0.1
sum(resNITSFI$padj < 0.1, na.rm=TRUE)
```

```
## [1] 94
```

```
sum(resNITELI$padj < 0.1, na.rm=TRUE)
```

```
## [1] 6635
```

```
sum(resSFIELI$padj < 0.1, na.rm=TRUE)
```

```
## [1] 4765
```

```
resSigNITSFI <- subset(resNITSFI, padj < 0.1)
resSigNITELI <- subset(resNITELI, padj < 0.1)
resSigSFIELI <- subset(resSFIELI, padj < 0.1)

# genes significantes con down-regulation más fuerte
sigNITSFI <- as.data.frame(head(resSigNITSFI[ order(resSigNITSFI$log2FoldChange,
```

```
                                                    decreasing = TRUE), c(1:4,6)]))
sigNITELI <- as.data.frame(head(resSigNITELI[ order(resSigNITELI$log2FoldChange,
                                                    decreasing = TRUE), c(1:4,6)]))
sigSFIELI <- as.data.frame(head(resSigSFIELI[ order(resSigSFIELI$log2FoldChange,
                                                    decreasing = TRUE), c(1:4,6)]))

round_df(sigNITSFI,6)
```

```
##                     baseMean log2FoldChange    lfcSE      stat     padj
## ENSG00000143552.5   85.61652       3.156954 0.715380  4.412976 0.020659
## ENSG00000261857.2  117.11325       2.624286 0.624706  4.200835 0.037718
## ENSG00000173432.6  155.94748       2.491968 0.618102  4.031648 0.049507
## ENSG00000234617.1  130.72333       0.821907 0.222332  3.696758 0.086035
## ENSG00000128578.5  113.87855      -0.973630 0.255384 -3.812419 0.067093
## ENSG00000029534.15 133.40003      -1.153664 0.318572 -3.621363 0.095025
```

```
round_df(sigNITELI,6)
```

```
##                    baseMean log2FoldChange    lfcSE     stat     padj
## ENSG00000253301.1  5.763226       4.875239 1.425621 3.419731 0.007198
## ENSG00000243584.1  4.642794       4.838893 1.027421 4.709749 0.000070
## ENSG00000170419.6 23.096407       4.641573 1.146549 4.048299 0.000938
## ENSG00000236848.2 14.964925       4.157532 1.052434 3.950396 0.001310
## ENSG00000215873.2  1.495046       4.084119 1.114586 3.664248 0.003411
## ENSG00000198414.5  1.256288       3.831388 1.368398 2.799908 0.036279
```

```
round_df(sigSFIELI,6)
```

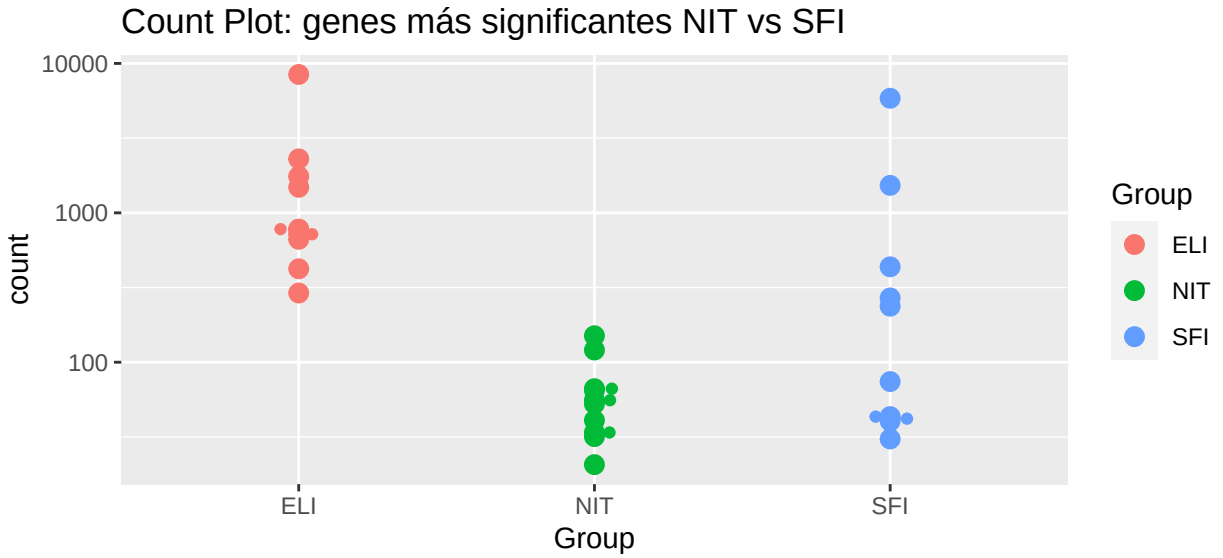
```
##                     baseMean log2FoldChange    lfcSE     stat     padj
## ENSG00000110680.8 1724.656655       8.177705 1.300593 6.287673 0.000000
## ENSG00000157005.3   14.553397       4.701818 1.299887 3.617097 0.005565
## ENSG00000253301.1    5.763226       4.510044 1.426515 3.161581 0.019456
## ENSG00000145808.4    2.268070       4.419819 1.256451 3.517701 0.007369
## ENSG00000100604.8   61.927148       4.288142 0.882679 4.858100 0.000059
## ENSG00000128564.5   34.609135       3.919798 0.794097 4.936170 0.000042
```

Visualizamos los resultados obtenidos.

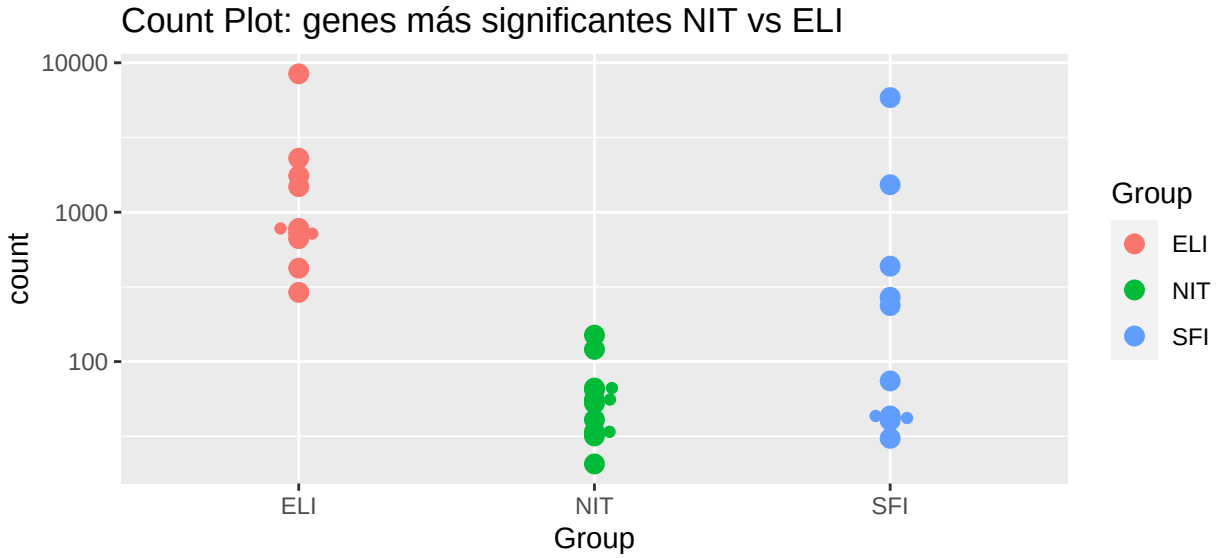
Count Plot

```
topGene1 <- rownames(resNITSFI)[which.min(resNITSFI$padj)]
topGene2 <- rownames(resNITELI)[which.min(resNITELI$padj)]
topGene3 <- rownames(resSFIELI)[which.min(resSFIELI$padj)]

library("ggbeeswarm")
library("gridExtra")
geneCounts1 <- plotCounts(dds, gene = topGene1, intgroup = c("Group"),returnData = TRUE)
```

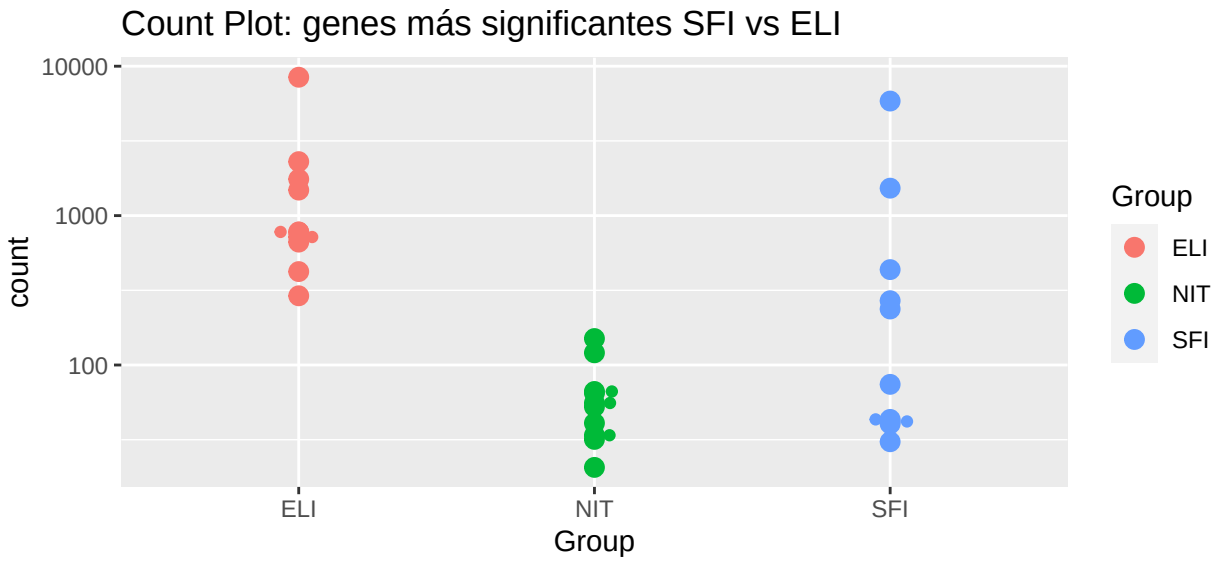
```
ggplot(geneCounts1, aes(x = Group, y = count, color = Group, group= Group)) +
  scale_y_log10() + geom_point(size = 3) + geom_beeswarm(cex = 2) +
  ggtitle("Count Plot: genes más significantes NIT vs SFI")
```




```
geneCounts2 <- plotCounts(dds, gene = topGene1, intgroup = c("Group"),returnData = TRUE)
ggplot(geneCounts1, aes(x = Group, y = count, color = Group, group= Group)) +
  scale_y_log10() + geom_point(size = 3) + geom_beeswarm(cex = 2) +
  ggtitle("Count Plot: genes más significantes NIT vs ELI")
```




```
geneCounts3 <- plotCounts(dds, gene = topGene1, intgroup = c("Group"),returnData = TRUE)
ggplot(geneCounts1, aes(x = Group, y = count, color = Group, group= Group)) +
  scale_y_log10() + geom_point(size = 3) + geom_beeswarm(cex = 2) +
  ggtitle("Count Plot: genes más significantes SFI vs ELI")
```

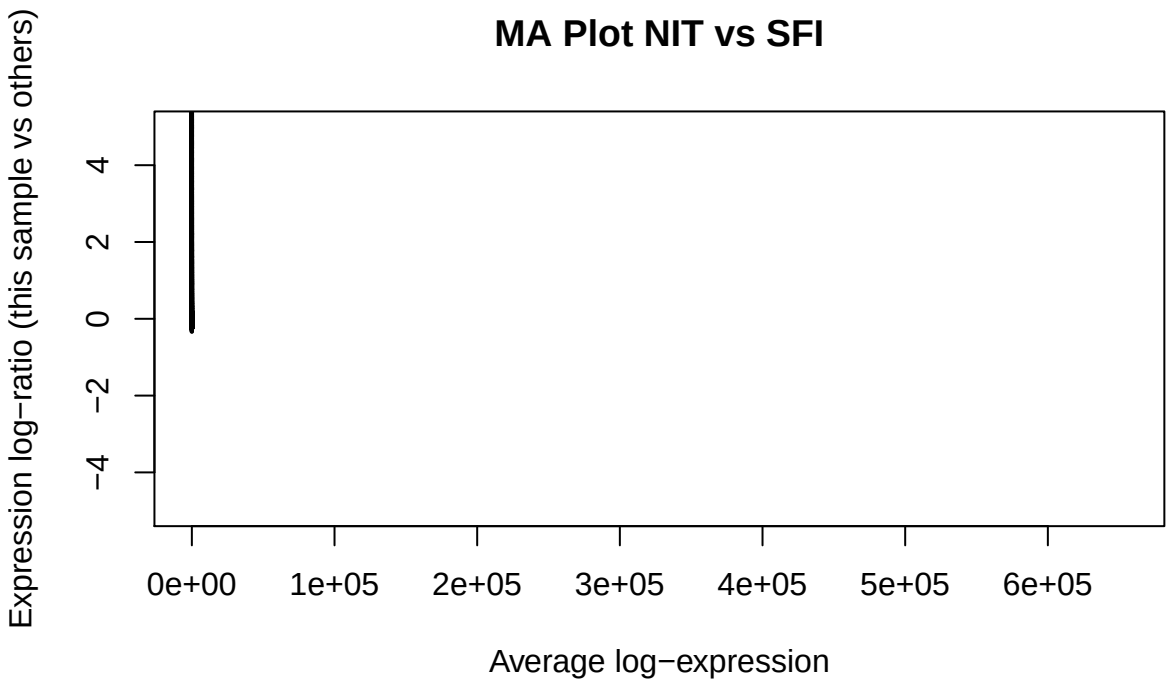


MA-Plot

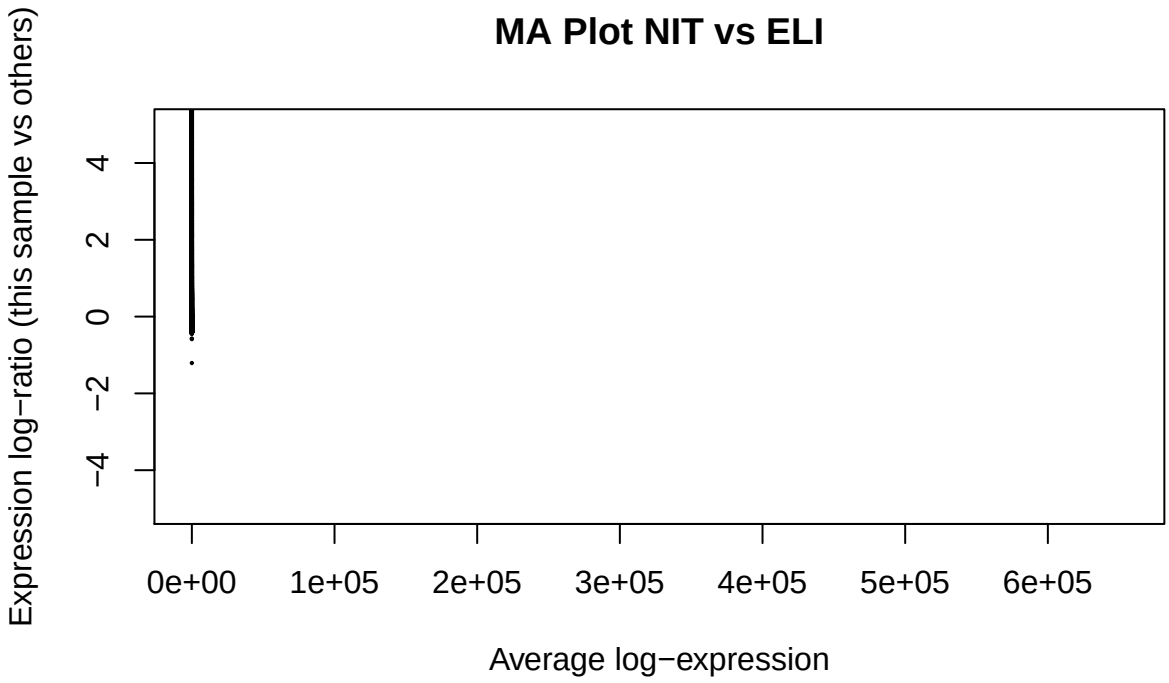
```
library("apeglm")
resultsNames(dds)
```

```
## [1] "Intercept"        "Group_NIT_vs_ELI" "Group_SFI_vs_ELI"
```

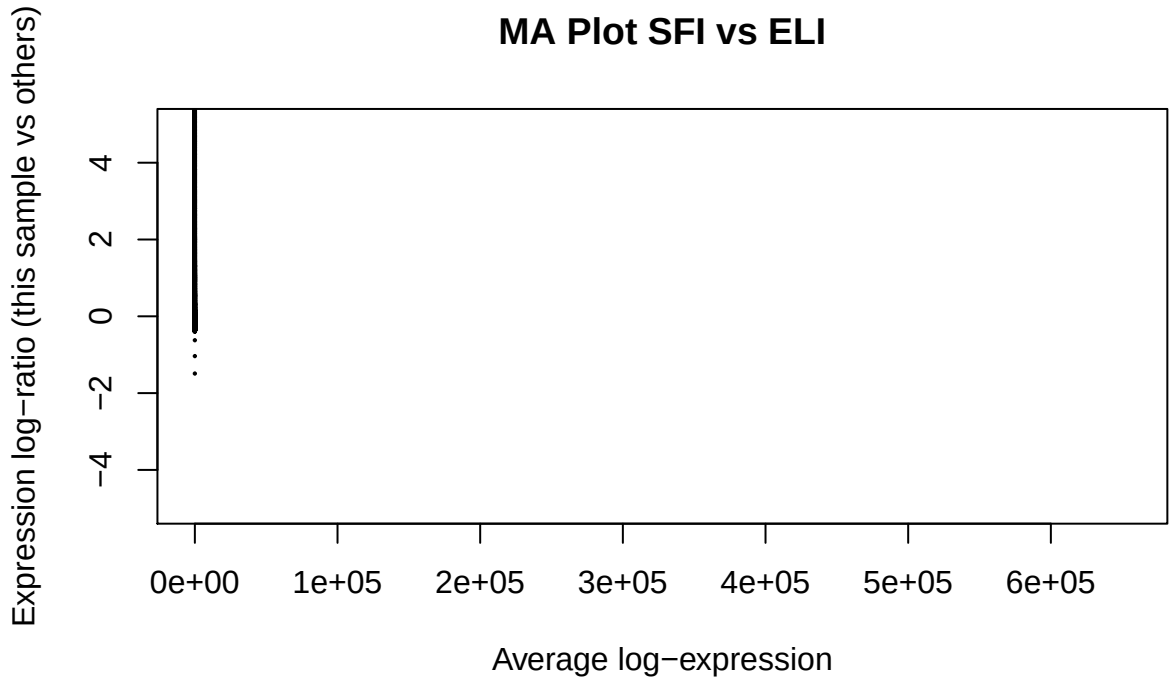
```
#resNITSFI <- lfcShrink(dds, coef="Group_NIT_vs_ELI", type="normal")
resNITSFI <- lfcShrink(dds,3,type="normal")
resNITELI <- lfcShrink(dds, coef="Group_NIT_vs_ELI", type="apeglm")
resSFIELI <- lfcShrink(dds, coef="Group_SFI_vs_ELI", type="apeglm")

plotMA(resNITSFI,ylim=c(-5,5),main="MA Plot NIT vs SFI")
```




```
plotMA(resNITELI,ylim=c(-5,5),main="MA Plot NIT vs ELI")
```

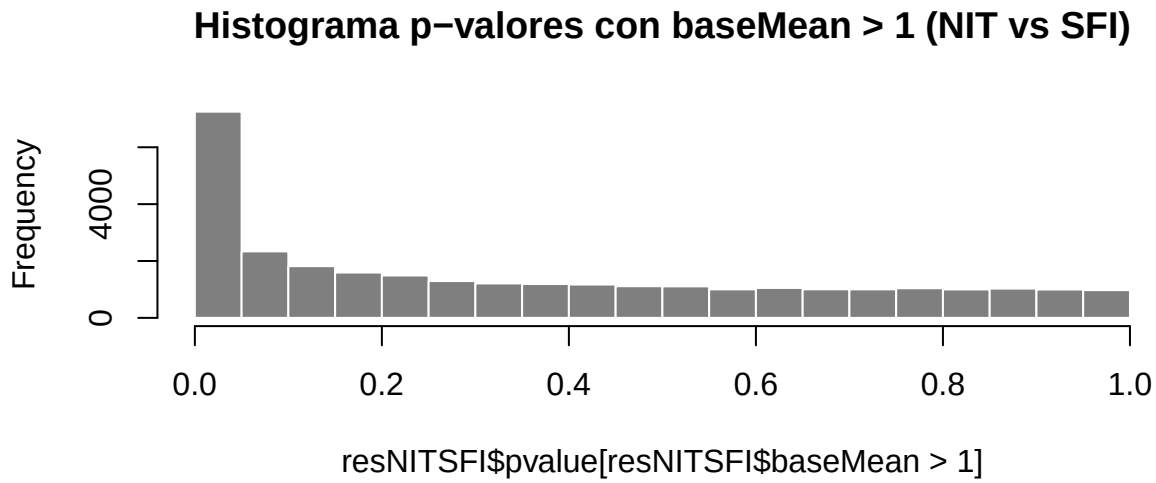


```
plotMA(resSFIELI,ylim=c(-5,5),main="MA Plot SFI vs ELI")
```

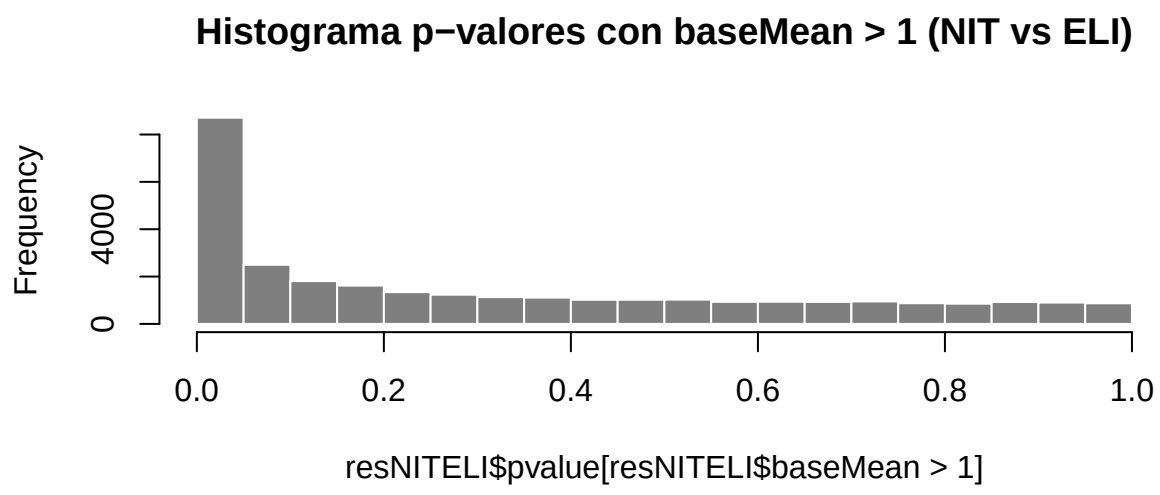


Histogramas

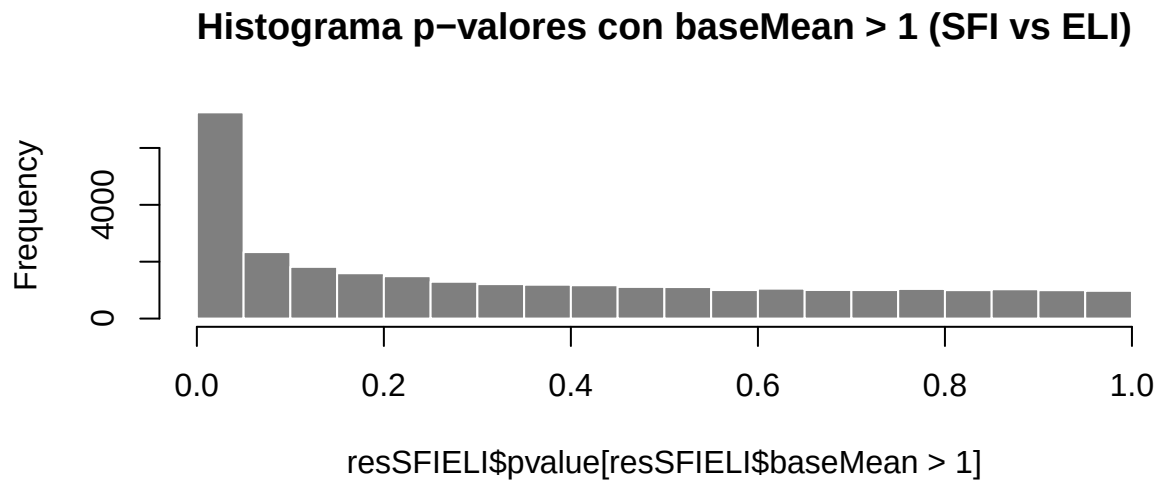
```
hist(resNITSFI$pvalue[resNITSFI$baseMean > 1], breaks = 0:20/20,
     col = "grey50", border = "white",main="Histograma p-valores con baseMean > 1 (NIT vs SFI)")
```




```
hist(resNITELI$pvalue[resNITELI$baseMean > 1], breaks = 0:20/20,
     col = "grey50", border = "white",main="Histograma p-valores con baseMean > 1 (NIT vs ELI)")
```

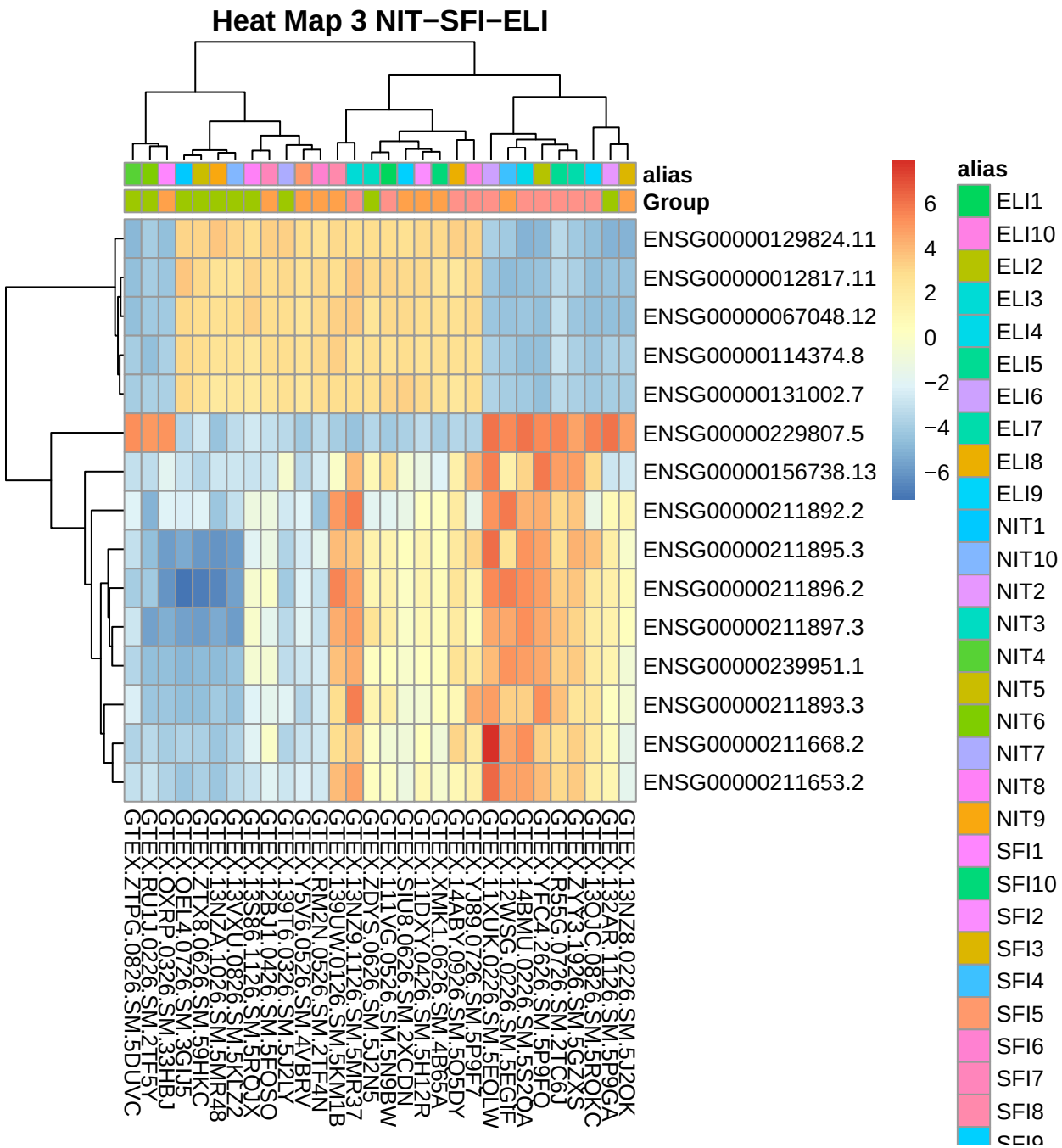



```
hist(resSFIELI$pvalue[resSFIELI$baseMean > 1], breaks = 0:20/20,
     col = "grey50", border = "white",main="Histograma p-valores con baseMean > 1 (SFI vs ELI)")
```

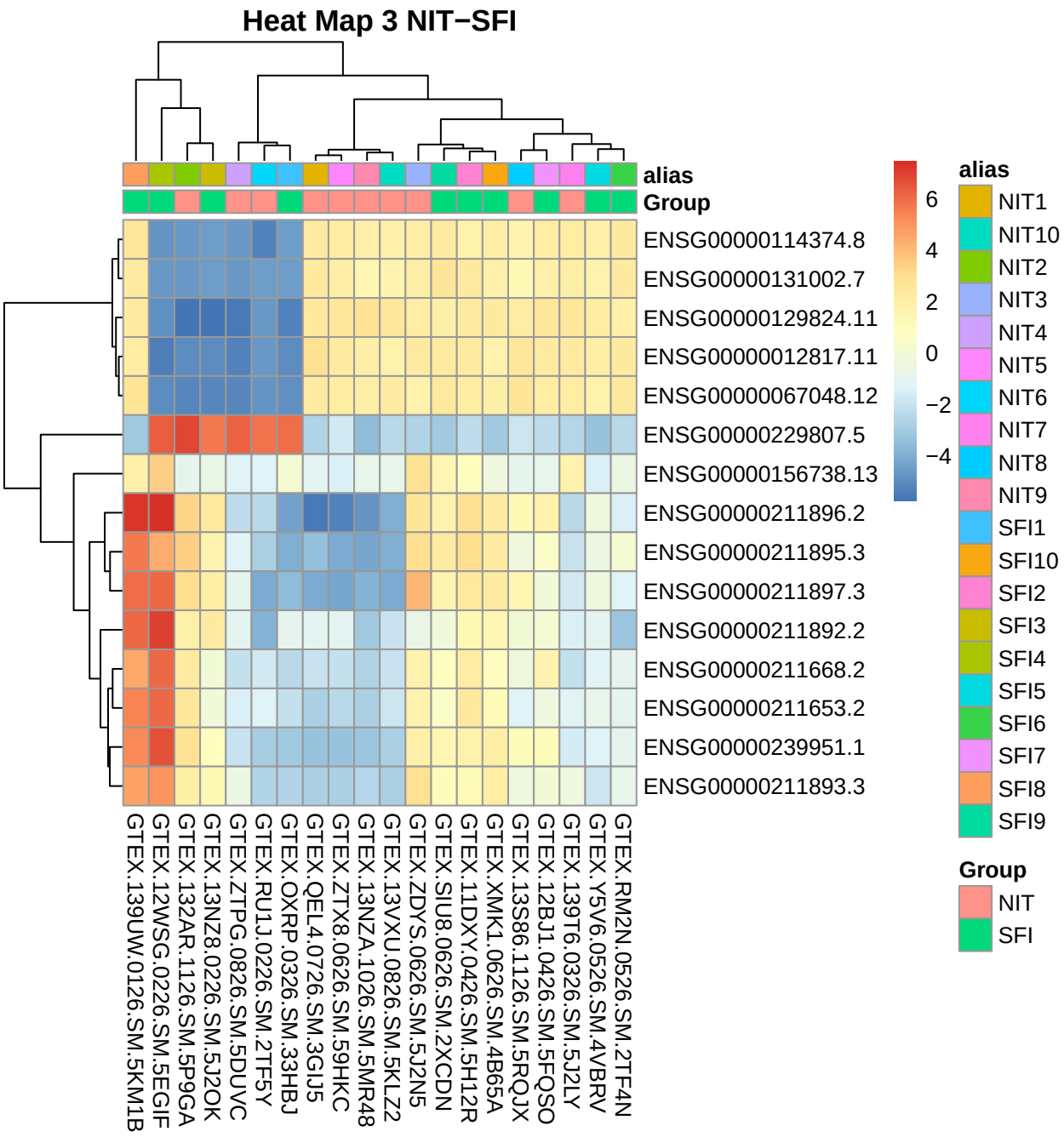


Gene clustering

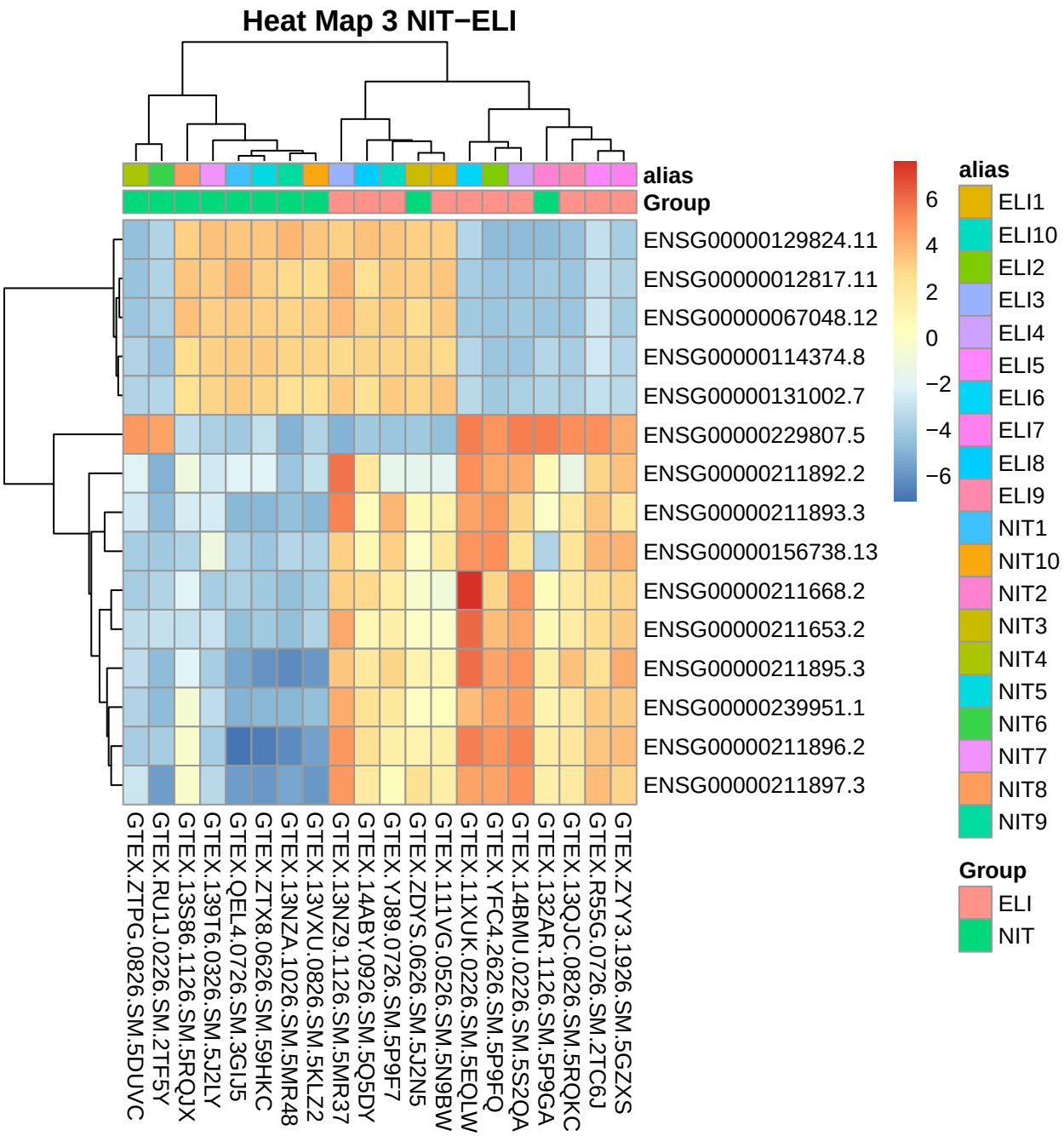
```
library("genefilter")
topVarGenes <- head(order(rowVars(assay(vsd)), decreasing = TRUE), 15)

# global
mat  <- assay(vsd)[topVarGenes, ]
mat  <- mat - rowMeans(mat)
anno <- as.data.frame(colData(vsd)[, c("Group","alias")])
pheatmap(mat, annotation_col = anno,main="Heat Map 3 NIT-SFI-ELI")
```



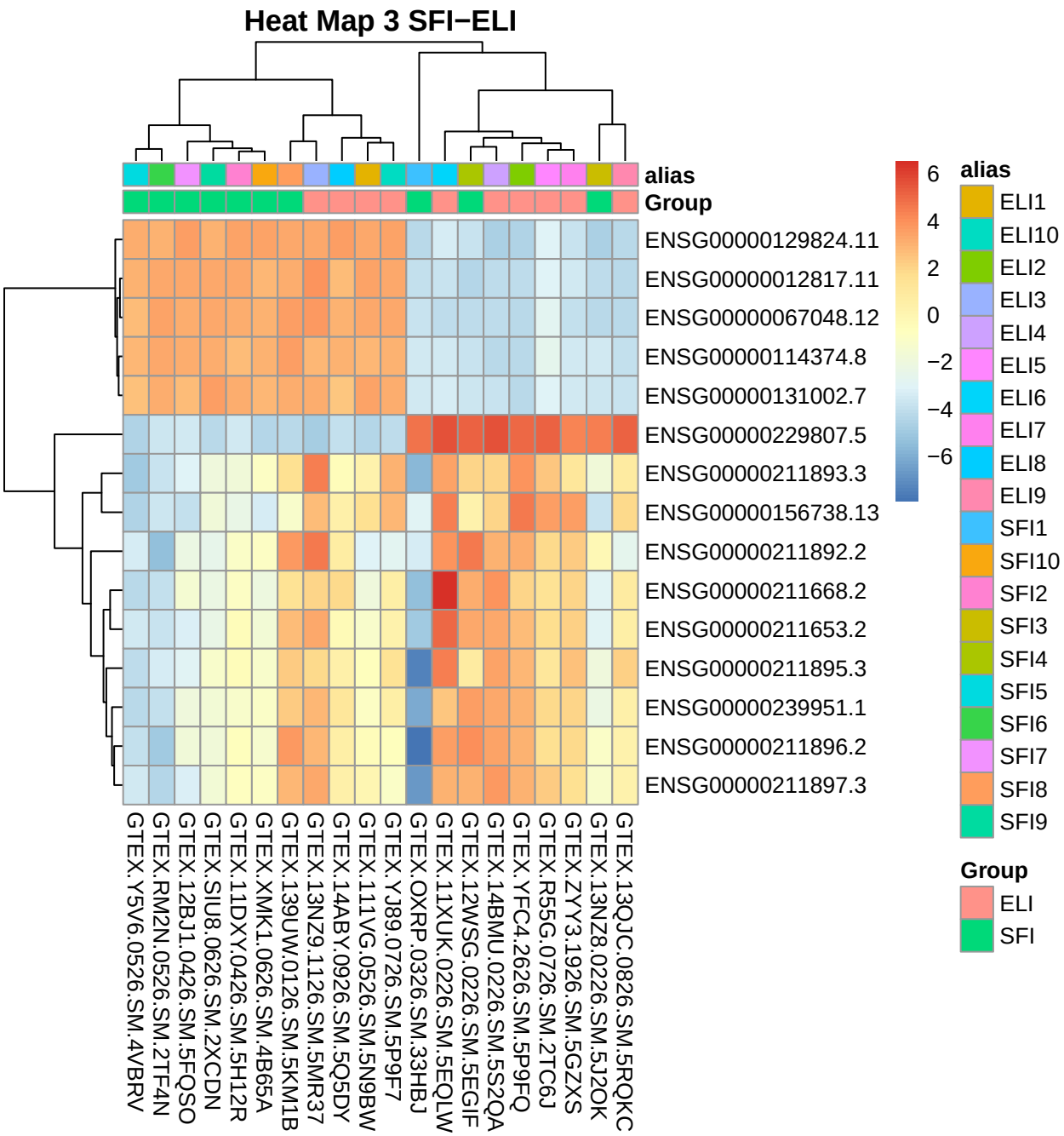

```
# NIT-SFI
matNS  <- assay(vsd)[topVarGenes, 1:20]
matNS  <- matNS - rowMeans(matNS)
annoNS <- as.data.frame(colData(vsd)[1:20, c("Group","alias")])
pheatmap(matNS, annotation_col = annoNS,main="Heat Map 3 NIT-SFI")
```




```
# NIT-ELI
matNE  <- assay(vsd)[topVarGenes, c(1:10,21:30)]
matNE  <- matNE - rowMeans(matNE)
annoNE <- as.data.frame(colData(vsd)[c(1:10,21:30), c("Group","alias")])
pheatmap(matNE, annotation_col = annoNE,main="Heat Map 3 NIT-ELI")
```




```
# SFI-ELI
matSE  <- assay(vsd)[topVarGenes, 11:30]
matSE  <- matSE - rowMeans(matSE)
annoSE <- as.data.frame(colData(vsd)[11:30, c("Group","alias")])
pheatmap(matSE, annotation_col = annoSE,main="Heat Map 3 SFI-ELI")
```



4.3 Anotación de resultados

Se buscan la coincidencias con los genes de la base de datos *org.Hs.eg.db*. Y los guardamos en el fichero *results.csv* para una posterior consulta.

```
library("org.Hs.eg.db")
library("AnnotationDbi")
res <- resNITSFI
res$symbol <- mapIds(org.Hs.eg.db,
```

```
                    keys=gsub("\\..*", "",row.names(res), fixed = FALSE),
                    column="SYMBOL",
                    keytype="ENSEMBL",
                    multiVals="first")

res$entrez <- mapIds(org.Hs.eg.db,
                    keys=gsub("\\..*", "",row.names(res), fixed = FALSE),
                    column="ENTREZID",
                    keytype="ENSEMBL",
                    multiVals="first")


resOrdered <- res[order(res$pvalue),]
head(resOrdered)
```

```
## log2 fold change (MAP): Group SFI vs ELI
## Wald test p-value: Group SFI vs ELI
## DataFrame with 6 rows and 8 columns
##                           baseMean    log2FoldChange             lfcSE
##                          <numeric>         <numeric>         <numeric>
## ENSG00000175857.4  170.051335993014 -3.04565170835467 0.326829434035722
## ENSG00000269404.2  312.440114764659 -3.24064728602351 0.358762704021958
## ENSG00000247982.2  1531.01851888727  -2.6856494106733  0.29549760897354
## ENSG00000167483.13 761.902767558693 -3.08649574105445 0.364988653591468
## ENSG00000068831.14 3445.52660742823 -2.51975741290058 0.286498097469292
## ENSG00000104921.10 305.845849726126 -2.98065706016091 0.370459730549136
##                               stat               pvalue                 padj
##                          <numeric>            <numeric>            <numeric>
## ENSG00000175857.4  -9.61956104104329 6.61126905855215e-22 1.25485590265287e-17
## ENSG00000269404.2  -9.59257482306124 8.59137274170118e-22 1.25485590265287e-17
## ENSG00000247982.2  -9.33830034168489 9.78928928783456e-21 9.53215728920744e-17
## ENSG00000167483.13 -9.12921170527036 6.89998725268441e-20 5.03906069063542e-16
## ENSG00000068831.14 -9.02412201966414 1.81142036436817e-19 1.05830423367846e-15
## ENSG00000104921.10 -8.85055163304124 8.70877611503033e-19  4.2400127978711e-15
##                         symbol      entrez
##                    <character> <character>
## ENSG00000175857.4         GAPT      202309
## ENSG00000269404.2         SPIB        6689
## ENSG00000247982.2    LINC00926      283663
## ENSG00000167483.13      NIBAN3      199786
## ENSG00000068831.14     RASGRP2       10235
## ENSG00000104921.10       FCER2        2208
```

```r
# genes en el top 100
resOrderedDF <- as.data.frame(resOrdered)[1:100, ]
write.csv(resOrderedDF, file = "results.csv")
```

4.4 Gene enrichment

```r
# ordenar los genes en orden decreciente
#gene_list = sort(na.omit(res$entrez), decreasing = TRUE)
gene_list = sort(rownames(resSigNITELI), decreasing = TRUE)
gse <- gseGO(geneList=gene_list,
             ont ="ALL",
             keyType = "ENSEMBL",
             nPerm = 10000,
             minGSSize = 3,
             maxGSSize = 800,
             pvalueCutoff = 0.05,
             verbose = TRUE,
             OrgDb = org.Hs.eg.db,
             pAdjustMethod = "none")


emapplot(gse, showCategory = 10)
```

5 Discusión

Es posible que se tuviera que replantear la selección de 10 muestras de cafda grupo. El grupo NIT qedaria infrarepresentado porque seleccionaríamos aproximadamente un 5% del total miestras que los otros dos grupos estaría sobrerepresentados con un 25% aprox. para el grupo SFI y con un 71% aprox. para el grupo ELI.

6 Apéndice

6.1 Código R

```r
#paquetes "nativos" de R
if(!require(magrittr)) install.packages("magrittr", dep=TRUE)
if(!require(dplyr)) install.packages("dplyr", dep=TRUE)
if(!require(ggplot2)) install.packages("ggplot2", dep=TRUE)
if(!require(pheatmap)) install.packages("pheatmap", dep=TRUE)
if(!require(RColorBrewer)) install.packages("RColorBrewer", dep=TRUE)
if(!require(ggbeeswarm)) install.packages("ggbeeswarm", dep=TRUE)
if(!require(edgeR)) install.packages("edgeR", dep=TRUE)
```

```
# paquetes de Bioconductor
if(!require(BiocManager)) install.packages("BiocManager")
if(!require(Rsamtools)) BiocManager::install("Rsamtools")
if(!require(GenomicFeatures)) BiocManager::install("GenomicFeatures")
if(!require(DESeq2)) BiocManager::install("DESeq2")
if(!require(apeglm)) BiocManager::install("apeglm")
if(!require(BiocParallel)) BiocManager::install("BiocParallel")
if(!require(genefilter)) BiocManager::install("genefilter")
if(!require(AnnotationDbi)) BiocManager::install("AnnotationDbi")
if(!require(ReportingTools)) BiocManager::install("ReportingTools")
if(!require(RUVSeq)) BiocManager::install("RUVSeq")
if(!require(sva)) BiocManager::install("sva")
if(!require(Gviz)) BiocManager::install("Gviz")

round_df <- function(x, digits) {
    # round all numeric variables
    # x: data frame
    # digits: number of digits to round
    numeric_columns <- sapply(x, mode) == 'numeric'
    x[numeric_columns] <-  round(x[numeric_columns], digits)
    x

# lectura de los ficheros
coldata0 <- read.csv("./input/targets.csv", header=TRUE, sep=",")
cts0  <- read.csv("./input/counts.csv",  header=TRUE, sep=";",row.names = 1)

# comprobación de las dimensiones
dim(coldata0)
dim(cts0)

coldata0$Sample_Name <- gsub("[-]",".",coldata0$Sample_Name)
rownames(coldata0) <- coldata0$Sample_Name
all(rownames(coldata0) %in% colnames(cts0))
all(rownames(coldata0) == colnames(cts0))

# seleccionar aleatóriamente 10 muestras de cada grupo (NIT, SFI y ELI) del fichero "targets"
set.seed(12345)
sample_NIT <- sample(coldata0[which(coldata0$Group=="NIT"),"Sample_Name"],10)
sample_SFI <- sample(coldata0[which(coldata0$Group=="SFI"),"Sample_Name"],10)
sample_ELI <- sample(coldata0[which(coldata0$Group=="ELI"),"Sample_Name"],10)


cts <- cts0[,c(sample_NIT,sample_SFI,sample_ELI)]
```

```r
coldata <- coldata0[c(sample_NIT,sample_SFI,sample_ELI),]
coldata$alias <- c(paste0("NIT",1:10),paste0("SFI",1:10),paste0("ELI",1:10))

all(rownames(coldata) %in% colnames(cts))
all(rownames(coldata) == colnames(cts))


library("DESeq2")
dds <- DESeqDataSetFromMatrix(countData = cts,
                              colData = coldata,
                              design = ~ Group)
nrow(dds)


keep <- rowSums(counts(dds)) > 1
dds <- dds[keep,]

vsd <- vst(dds, blind = FALSE)
head(assay(vsd)[,1:2], 3)  # datos de solo dos muestras
colData(vsd)[,c(1:2,7:8,10)]


library("dplyr")
library("ggplot2")

dds <- estimateSizeFactors(dds)

df <- bind_rows(
  as_data_frame(log2(counts(dds, normalized=TRUE)[, 1:2]+1)) %>%
         mutate(transformation = "log2(x + 1)"),
  as_data_frame(assay(vsd)[, 1:2]) %>% mutate(transformation = "vst"))

colnames(df)[1:2] <- c("x", "y")

ggplot(df, aes(x = x, y = y)) + geom_hex(bins = 80) +
  coord_fixed() + facet_grid( . ~ transformation)


sampleDists <- dist(t(assay(vsd)))

library("pheatmap")
library("RColorBrewer")
```

```
sampleDistMatrix <- as.matrix( sampleDists )
rownames(sampleDistMatrix) <- vsd$alias
colnames(sampleDistMatrix) <- NULL
colors <- colorRampPalette( rev(brewer.pal(9, "Greens")) )(255)
pheatmap(sampleDistMatrix,
         clustering_distance_rows = sampleDists,
         clustering_distance_cols = sampleDists,
         col = colors)

plotPCA(vsd, intgroup  = c("Group"))

dds <- DESeq(dds, parallel =TRUE)

resNITSFI <- results(dds, contrast=c("Group","NIT","SFI"))
round_df(as.data.frame(resNITSFI)[1:5,c(1:4,6)],6)

resNITELI <- results(dds, contrast=c("Group","NIT","ELI"))
round_df(as.data.frame(resNITELI)[1:5,c(1:4,6)],6)

resSFIELI <- results(dds, contrast=c("Group","SFI","ELI"))
round_df(as.data.frame(resSFIELI)[1:5,c(1:4,6)],6)

table(resNITSFI$padj < 0.05)
table(resNITELI$padj < 0.05)
table(resSFIELI$padj < 0.05)

# genes con un p valor ajustado < 0.1
sum(resNITSFI$padj < 0.1, na.rm=TRUE)
sum(resNITELI$padj < 0.1, na.rm=TRUE)
sum(resSFIELI$padj < 0.1, na.rm=TRUE)

resSigNITSFI <- subset(resNITSFI, padj < 0.1)
resSigNITELI <- subset(resNITELI, padj < 0.1)
resSigSFIELI <- subset(resSFIELI, padj < 0.1)

# genes significantes con down-regulation más fuerte
sigNITSFI <- as.data.frame(head(resSigNITSFI[ order(resSigNITSFI$log2FoldChange,
                                                    decreasing = TRUE), c(1:4,6)]))
sigNITELI <- as.data.frame(head(resSigNITELI[ order(resSigNITELI$log2FoldChange,
                                                    decreasing = TRUE), c(1:4,6)]))
sigSFIELI <- as.data.frame(head(resSigSFIELI[ order(resSigSFIELI$log2FoldChange,
                                                    decreasing = TRUE), c(1:4,6)]))
```

```r
round_df(sigNITSFI,6)
round_df(sigNITELI,6)
round_df(sigSFIELI,6)


topGene1 <- rownames(resNITSFI)[which.min(resNITSFI$padj)]
topGene2 <- rownames(resNITELI)[which.min(resNITELI$padj)]
topGene3 <- rownames(resSFIELI)[which.min(resSFIELI$padj)]

library("ggbeeswarm")
library("gridExtra")
geneCounts1 <- plotCounts(dds, gene = topGene1, intgroup = c("Group"),returnData = TRUE)
ggplot(geneCounts1, aes(x = Group, y = count, color = Group, group= Group)) +
  scale_y_log10() + geom_point(size = 3) + geom_beeswarm(cex = 2) +
  ggtitle("Count Plot: genes más significantes NIT vs SFI")

geneCounts2 <- plotCounts(dds, gene = topGene1, intgroup = c("Group"),returnData = TRUE)
ggplot(geneCounts1, aes(x = Group, y = count, color = Group, group= Group)) +
  scale_y_log10() + geom_point(size = 3) + geom_beeswarm(cex = 2) +
  ggtitle("Count Plot: genes más significantes NIT vs ELI")

geneCounts3 <- plotCounts(dds, gene = topGene1, intgroup = c("Group"),returnData = TRUE)
ggplot(geneCounts1, aes(x = Group, y = count, color = Group, group= Group)) +
  scale_y_log10() + geom_point(size = 3) + geom_beeswarm(cex = 2) +
  ggtitle("Count Plot: genes más significantes SFI vs ELI")

library("apeglm")
resultsNames(dds)
#resNITSFI <- lfcShrink(dds, coef="Group_NIT_vs_ELI", type="normal")
resNITSFI <- lfcShrink(dds,3,type="normal")
resNITELI <- lfcShrink(dds, coef="Group_NIT_vs_ELI", type="apeglm")
resSFIELI <- lfcShrink(dds, coef="Group_SFI_vs_ELI", type="apeglm")

plotMA(resNITSFI,ylim=c(-5,5),main="MA Plot NIT vs SFI")
plotMA(resNITELI,ylim=c(-5,5),main="MA Plot NIT vs ELI")
plotMA(resSFIELI,ylim=c(-5,5),main="MA Plot SFI vs ELI")

hist(resNITSFI$pvalue[resNITSFI$baseMean > 1], breaks = 0:20/20,
     col = "grey50", border = "white",main="Histograma p-valores con baseMean > 1 (NIT vs SFI)")

hist(resNITELI$pvalue[resNITELI$baseMean > 1], breaks = 0:20/20,
     col = "grey50", border = "white",main="Histograma p-valores con baseMean > 1 (NIT vs ELI)")
```

```
hist(resSFIELI$pvalue[resSFIELI$baseMean > 1], breaks = 0:20/20,
     col = "grey50", border = "white",main="Histograma p-valores con baseMean > 1 (SFI vs ELI)")

library("genefilter")
topVarGenes <- head(order(rowVars(assay(vsd)), decreasing = TRUE), 15)

# global
mat  <- assay(vsd)[topVarGenes, ]
mat  <- mat - rowMeans(mat)
anno <- as.data.frame(colData(vsd)[, c("Group","alias")])
pheatmap(mat, annotation_col = anno,main="Heat Map 3 NIT-SFI-ELI")
# NIT-SFI
matNS  <- assay(vsd)[topVarGenes, 1:20]
matNS  <- matNS - rowMeans(matNS)
annoNS <- as.data.frame(colData(vsd)[1:20, c("Group","alias")])
pheatmap(matNS, annotation_col = annoNS,main="Heat Map 3 NIT-SFI")
# NIT-ELI
matNE  <- assay(vsd)[topVarGenes, c(1:10,21:30)]
matNE  <- matNE - rowMeans(matNE)
annoNE <- as.data.frame(colData(vsd)[c(1:10,21:30), c("Group","alias")])
pheatmap(matNE, annotation_col = annoNE,main="Heat Map 3 NIT-ELI")
# SFI-ELI
matSE  <- assay(vsd)[topVarGenes, 11:30]
matSE  <- matSE - rowMeans(matSE)
annoSE <- as.data.frame(colData(vsd)[11:30, c("Group","alias")])
pheatmap(matSE, annotation_col = annoSE,main="Heat Map 3 SFI-ELI")

library("org.Hs.eg.db")
library("AnnotationDbi")

res$symbol <- mapIds(org.Hs.eg.db,
                     keys=gsub("\\..*", "",row.names(res), fixed = FALSE),
                     column="SYMBOL",
                     keytype="ENSEMBL",
                     multiVals="first")

res$entrez <- mapIds(org.Hs.eg.db,
                     keys=gsub("\\..*", "",row.names(res), fixed = FALSE),
                     column="ENTREZID",
                     keytype="ENSEMBL",
                     multiVals="first")
```

```r
resOrdered <- res[order(res$pvalue),]
head(resOrdered)

# genes en el top 100
resOrderedDF <- as.data.frame(resOrdered)[1:100, ]
write.csv(resOrderedDF, file = "results.csv")
```