DE_Cyt_vs_Nuc

Núria Rivera Brugués

2023-10-02

```
counts_kidney_human<-read.delim("./GSE116008_Fractionation_counts_polyA_hek293t.txt")
rownames(counts_kidney_human)<-counts_kidney_human[,1]
counts_kidney_human<-counts_kidney_human[,-1]

colnames(counts_kidney_human)<-c("C1","C2","N1","N2")

targets_kidney_human<-read.delim("./SraRunTable_kidney.txt", header=T, sep=",")
targets_kidney_human<-targets_kidney_human[,c(1,6,8,15,27,28,29)]
info<-targets_kidney_human
info$Sample<-c("N1", "N2", "C1", "C2")
info$Fraction<-c("Nuc", "Nuc", "Cyto", "Cyto")
info<-info[c(3,4,1,2),]
write.csv(info, "./info_kidney.csv")
```

```
# redueixo el dataframe (nomes Cyt vs Nuc del neurones corticals primaries)
info$Subgroup<-c(1,2,1,2)
rownames(info)<-info$Sample
barcode=factor(info$Sample)
subgroup=factor(info$Subgroup)
group=factor(info$Group)
fraction<-factor(info$Fraction)

View(info)
```

```
y=DGEList(counts_kidney_human)
isexpr <- rowSums(cpm(y) > 1) >= 3
y=y[isexpr,keep.lib.size=FALSE]
y=calcNormFactors(y)
y$samples
```

```
##    group lib.size norm.factors
## C1     1 34879278    0.9281309
## C2     1 40873151    0.8705282
## N1     1 44616275    1.1017484
## N2     1 42462340    1.1233770
```

```
dim(y)
```

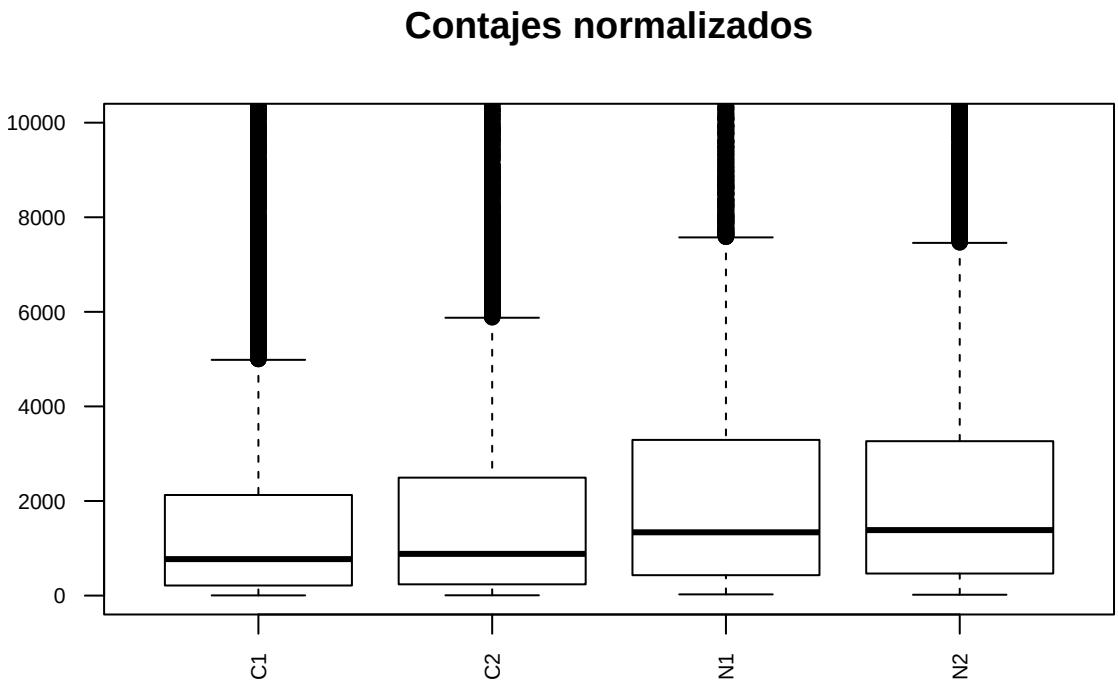
```
## [1] 13522     4
```

Exploración de los datos

Una vez descartados los genes poco expresados y con los recuentos almacenados en un objeto DGEList, podemos‘proceder a realizar algunos gráficos exploratorios para determinar si los datos aparentan buena calidad y/o si presentan algun problema.

Distribución de los contajes

```
boxplot(y$counts, col = y$samples$cols, las = 2, cex.axis = 0.7,
    main = "Contajes normalizados", ylim = c(0, 10000))
```



Análisis de similaridad entre las muestras

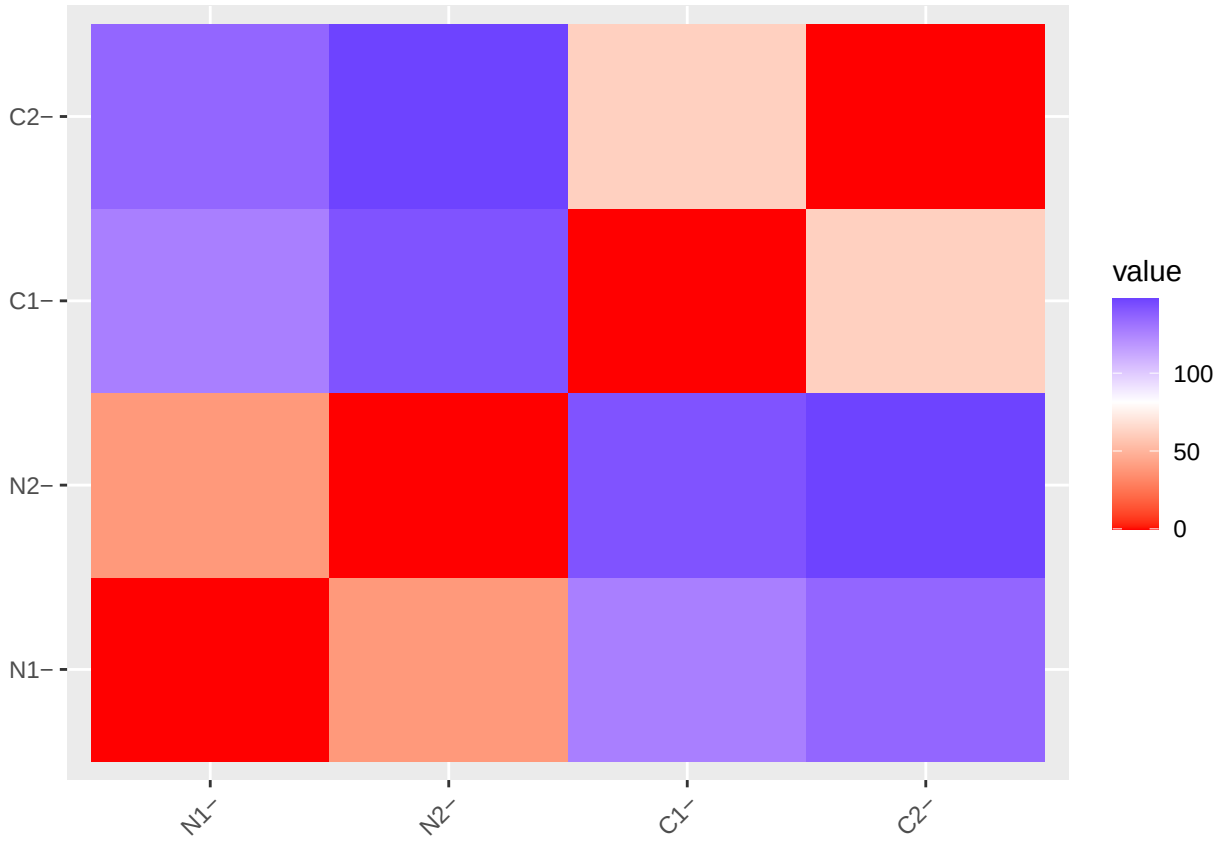
Distancia entre muestras

La función dist permite calcular una matriz de distancias que contiene las comparaciones dos a dos entre todas las muestras. Por defecto se utiliza una distancia euclídea.

```
log2count_norm <- cpm(y, log = TRUE)
sampleDists <- dist(t(log2count_norm))
round(sampleDists, 1)
```

```
##       C1    C2    N1
## C2  61.5
## N1 126.6 135.3
## N2 141.9 147.5  38.6
```

```
par(mfrow = c(1, 1))
fviz_dist(sampleDists)
```



Agrupamiento jerárquico

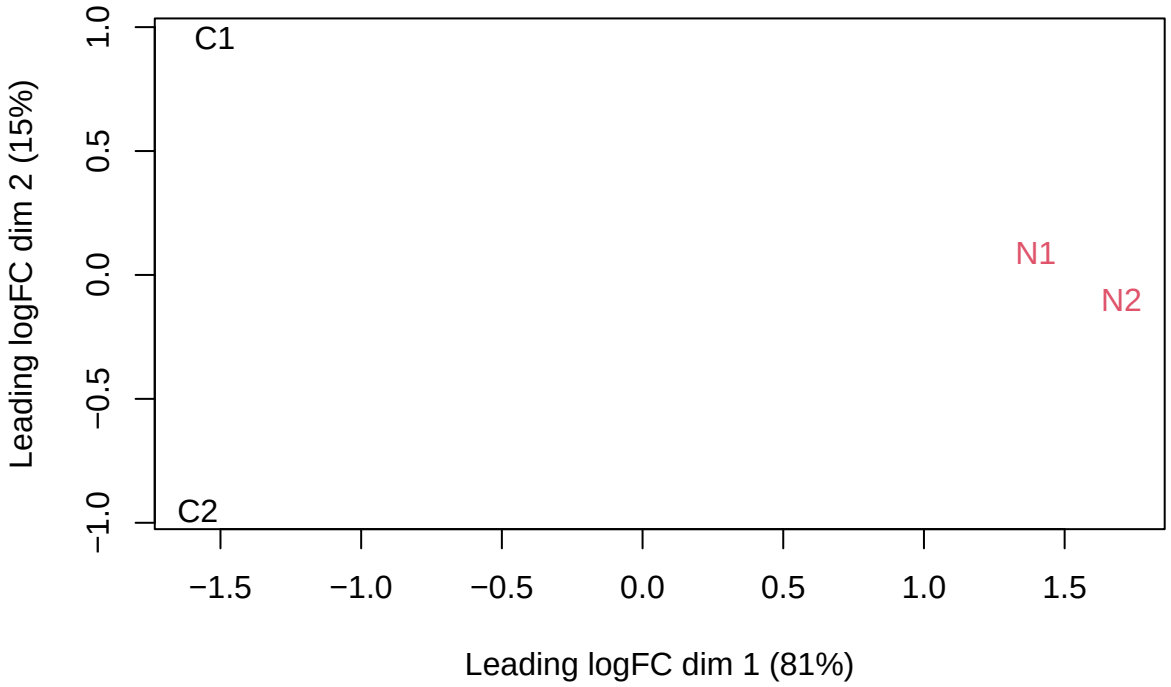
Un agrupamiento jerárquico proporciona una representación alternativa, también basada en la matriz de distancias.

```
hc <- hclust(sampleDists)
plot(hc, labels = colnames(log2count_norm), main = "Agrpamiento jerárquico de las muestras",
    cex = 0.8)
```

Análisis de Escalamiento Multidimensional (MDS)

Reducción dimensional

```
plotMDS(y, col=as.numeric(fraction), labels=barcode, cex = 1 )
```




```
pdf(paste("plotMDS.pdf",sep=""))
plotMDS(y, col=as.numeric(fraction), labels=barcode, cex = 1 )
dev.off()
```

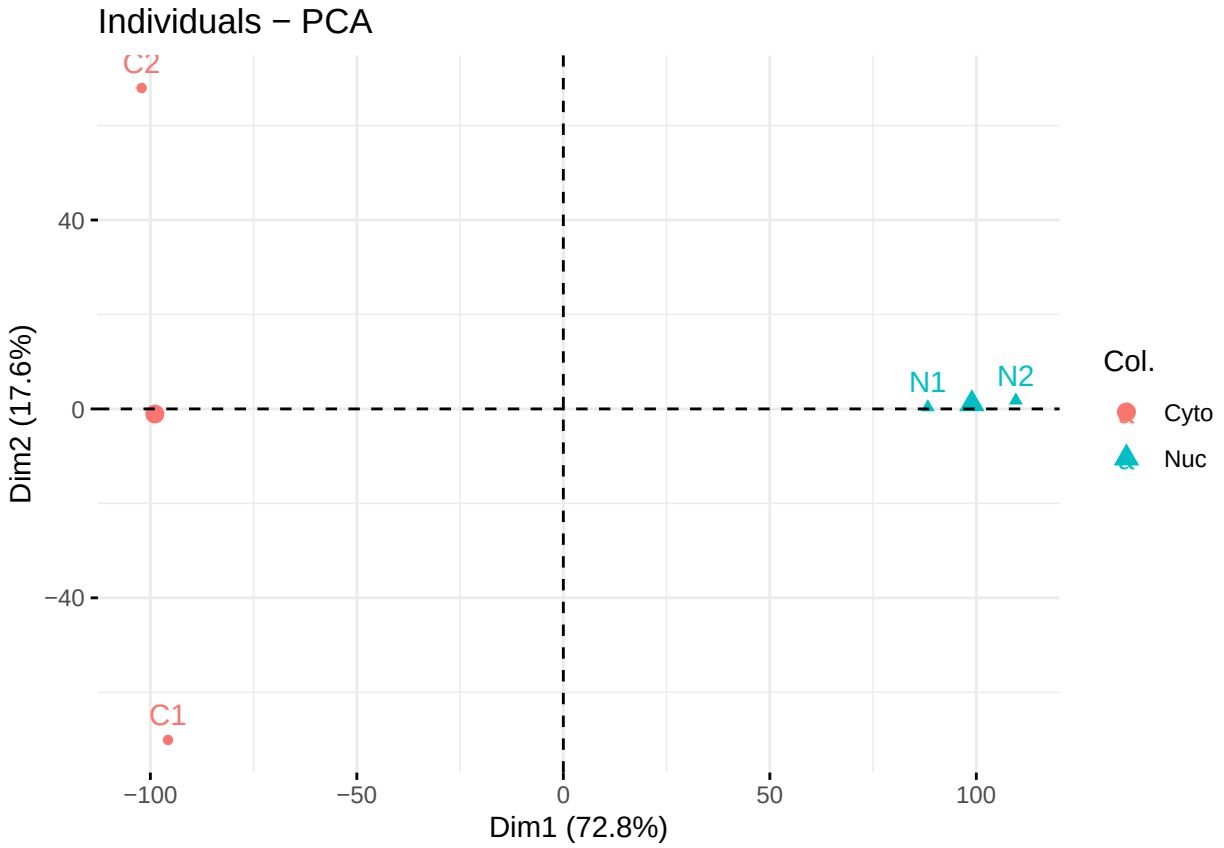
```
## pdf
##   2
```

PCA

```
library(FactoMineR)

pca.raw.y <- log2(y$counts+1)

pca.y <- PCA(t(pca.raw.y),graph = F)
fviz_pca_ind(pca.y, col.ind = fraction)
```




```
pdf(paste("PAC.pdf",sep=""))
fviz_pca_ind(pca.y, col.ind = fraction)
dev.off()
```

```
## pdf
##   2
```

Análisis de expresión diferencial (DE)

El objetivo del análisis de expresión diferencial es seleccionar genes cuya expresión difiere entre grupos.

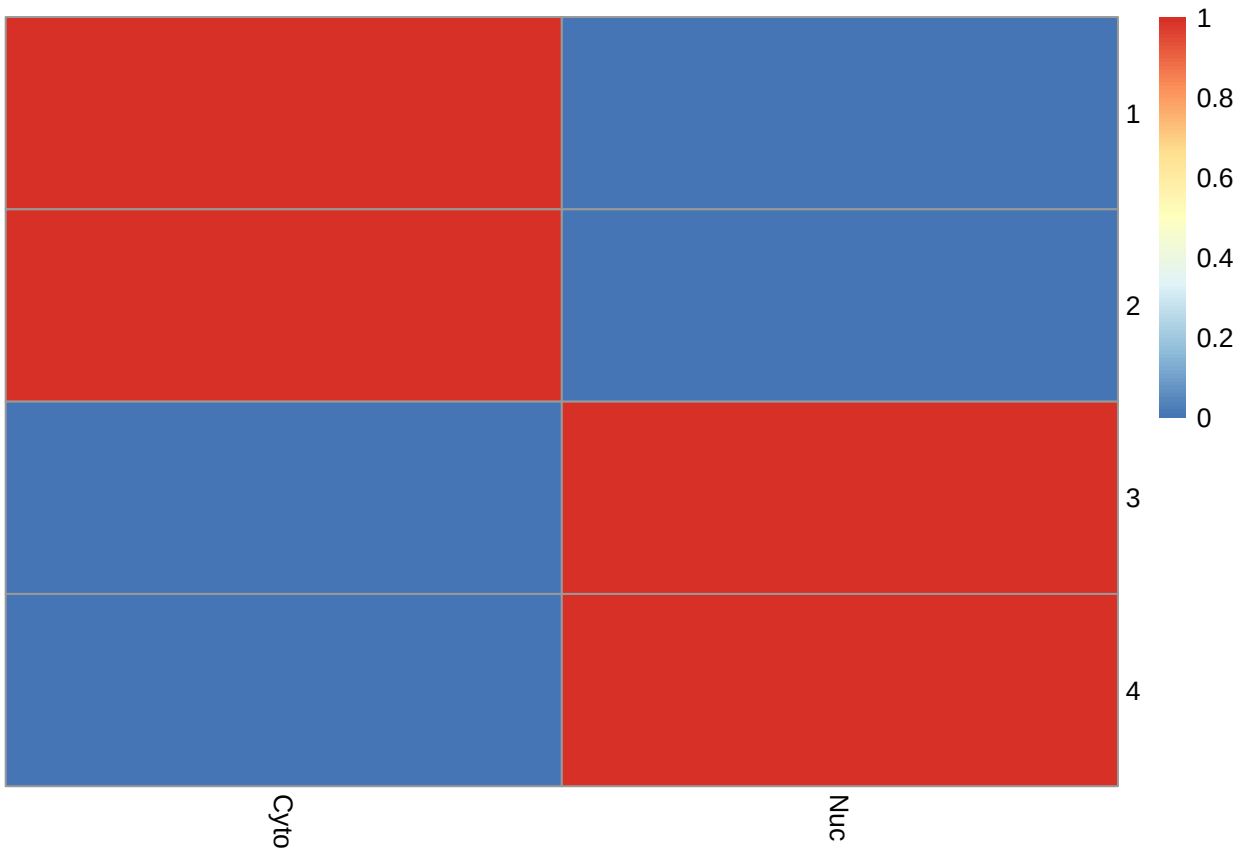
Selección de genes usando limma-Voom

La ventaja principal de esta aproximación es que permite trabajar con toda la flexibilidad de los modelos lineales para representar diseños experimentales, y, en muchos casos , aprovechar la experiencia previa del usuario en el manejo de limma.

Matriz de diseño

Utilizando la variable group podemos definir una matriz de diseño y, sobre ésta, los contrastes que nos interesan.

```
mod <- model.matrix(~0+fraction)
colnames(mod)=gsub("fraction","",colnames(mod))
pheatmap(mod,cluster_rows = FALSE,cluster_cols = FALSE)
```




```
mod
```

```
##   Cyto Nuc
## 1    1   0
## 2    1   0
## 3    0   1
## 4    0   1
## attr(,"assign")
## [1] 1 1
## attr(,"contrasts")
## attr(,"contrasts")$fraction
## [1] "contr.treatment"
```

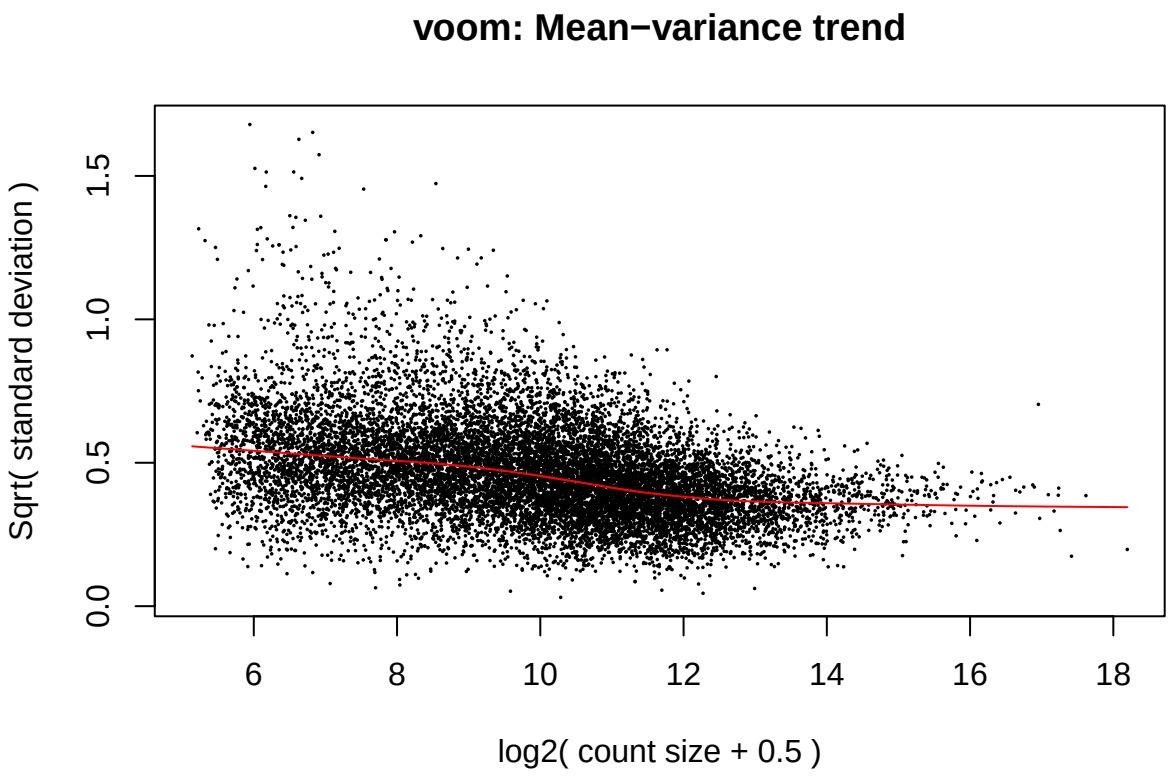
Matriz de contrastes

```
contr.matrix <- makeContrasts(
  Cyto_vs_Nuc = Cyto-Nuc,
levels=colnames(mod))
contr.matrix
```

```
##       Contrasts
## Levels Cyto_vs_Nuc
##   Cyto           1
##   Nuc           -1
```

Transformación de los contajes

```
v=voom(y,mod, plot = T)
```




```
v
```

```
## An object of class "EList"
## $targets
##    group lib.size norm.factors
## C1     1 32372536    0.9281309
## C2     1 35581229    0.8705282
## N1     1 49155909    1.1017484
## N2     1 47701214    1.1233770
##
## $E
##                       C1       C2       N1       N2
## ENSG00000000003 6.858859 6.670522 7.894866 7.754052
## ENSG00000000419 6.264362 6.497783 5.846528 5.593658
## ENSG00000000457 4.131778 3.709593 4.607721 4.407752
## ENSG00000000460 6.361325 6.145877 6.637653 6.456953
## ENSG00000001036 5.932763 5.567627 7.689544 7.578468
## 13517 more rows ...
```

```
##
## $weights
##          [,1]     [,2]     [,3]     [,4]
## [1,] 44.58200 46.21187 58.33730 58.16144
## [2,] 39.76723 41.53875 38.99623 38.39646
## [3,] 17.61405 18.14717 24.63159 24.25404
## [4,] 38.04559 39.88379 49.09573 48.60741
## [5,] 31.32887 33.08273 57.53391 57.33854
## 13517 more rows ...
##
## $design
##   Cyto Nuc
## 1    1   0
## 2    1   0
## 3    0   1
## 4    0   1
## attr(,"assign")
## [1] 1 1
## attr(,"contrasts")
## attr(,"contrasts")$fraction
## [1] "contr.treatment"
```

Selección de genes diferencialmente expresados

Como en el caso de los microarrays el objeto v y las matrices de diseño y contrastes se utilizaran para ajustar un modelo y, a continuación realizar las comparaciones especificadas sobre el modelo ajustado. El proceso finaliza con la regularización del estimador del error usando la función eBayes.

```
fit=lmFit(v,mod)
fit2 <- contrasts.fit(fit, contr.matrix)
fit2 <- eBayes(fit2)
(results<-topTable(fit2, coef = 1, adjust="BH"))
```

```
##                     logFC   AveExpr         t      P.Value    adj.P.Val
## ENSG00000229807 -5.249792 11.292743 -35.26005 1.271644e-17 1.719517e-13
## ENSG00000198886  4.446755 12.075137  31.48565 8.847266e-17 5.545282e-13
## ENSG00000198899  4.469159 10.752527  30.88431 1.230280e-16 5.545282e-13
## ENSG00000198712  3.931950 11.917380  27.56434 8.559871e-16 2.893664e-12
## ENSG00000198763  4.004971 10.978200  26.85142 1.336721e-15 3.615027e-12
## ENSG00000248527  4.406316  7.927375  26.43119 1.747655e-15 3.938631e-12
## ENSG00000212907  3.808996  8.776558  24.40908 6.737574e-15 1.301507e-11
## ENSG00000198840  4.415332  7.880593  22.17156 3.412534e-14 5.768035e-11
## ENSG00000245532 -4.449544  6.881616 -21.26482 6.882726e-14 1.034091e-10
## ENSG00000075826 -5.387170  4.214263 -20.79435 1.001480e-13 1.354201e-10
##                         B
## ENSG00000229807 30.33582
## ENSG00000198886 28.58356
## ENSG00000198899 28.25520
## ENSG00000198712 26.43302
## ENSG00000198763 25.99620
## ENSG00000248527 25.61553
## ENSG00000212907 24.40229
## ENSG00000198840 22.78374
```

```
## ENSG00000245532 21.97005
## ENSG00000075826 21.30725
```

```
summary(decideTests(fit2))
```

```
##        Cyto_vs_Nuc
## Down          4229
## NotSig        5611
## Up            3682
```

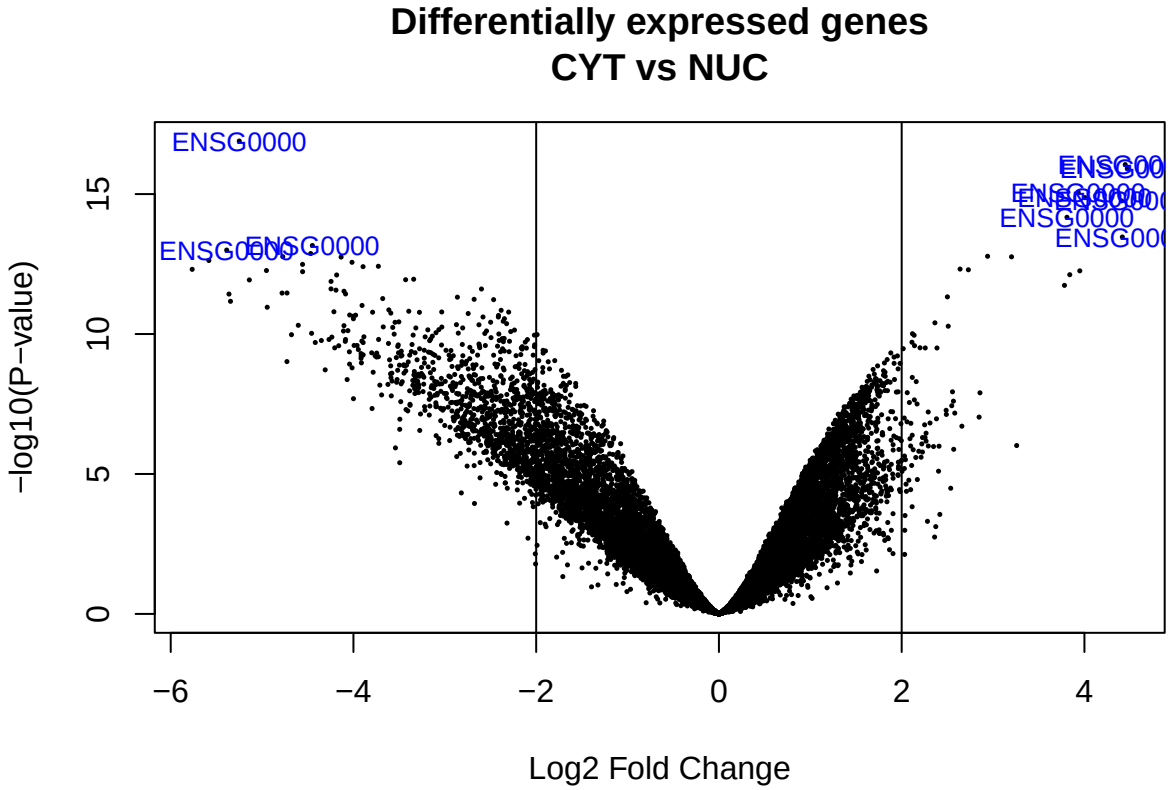
```
summa.fit <- decideTests(fit2, p.value = 0.01, lfc = 3)
summary(summa.fit)
```

```
##        Cyto_vs_Nuc
## Down           219
## NotSig       13291
## Up              12
```

Visualización de los resultados

Volcano Plot

```
volcanoplot(fit2, coef = 1, highlight = 10,names=rownames(fit2) ,main =paste( "Differentially expressed
abline(v=c(-2,2))
```



```
pdf(paste("volcanoplot.pdf",sep=""))
volcanoplot(fit2, coef = 1, highlight = 10,names=rownames(fit2) ,main =paste( "Differentially expressed
abline(v=c(-2,2))
dev.off()
```

```
## pdf
##   2
```

Perfiles de expresión

Con el fin de observar si existen perfiles de expresión diferenciados podemo realizar un mapa de colores con los genes más diferencialmente expresados.

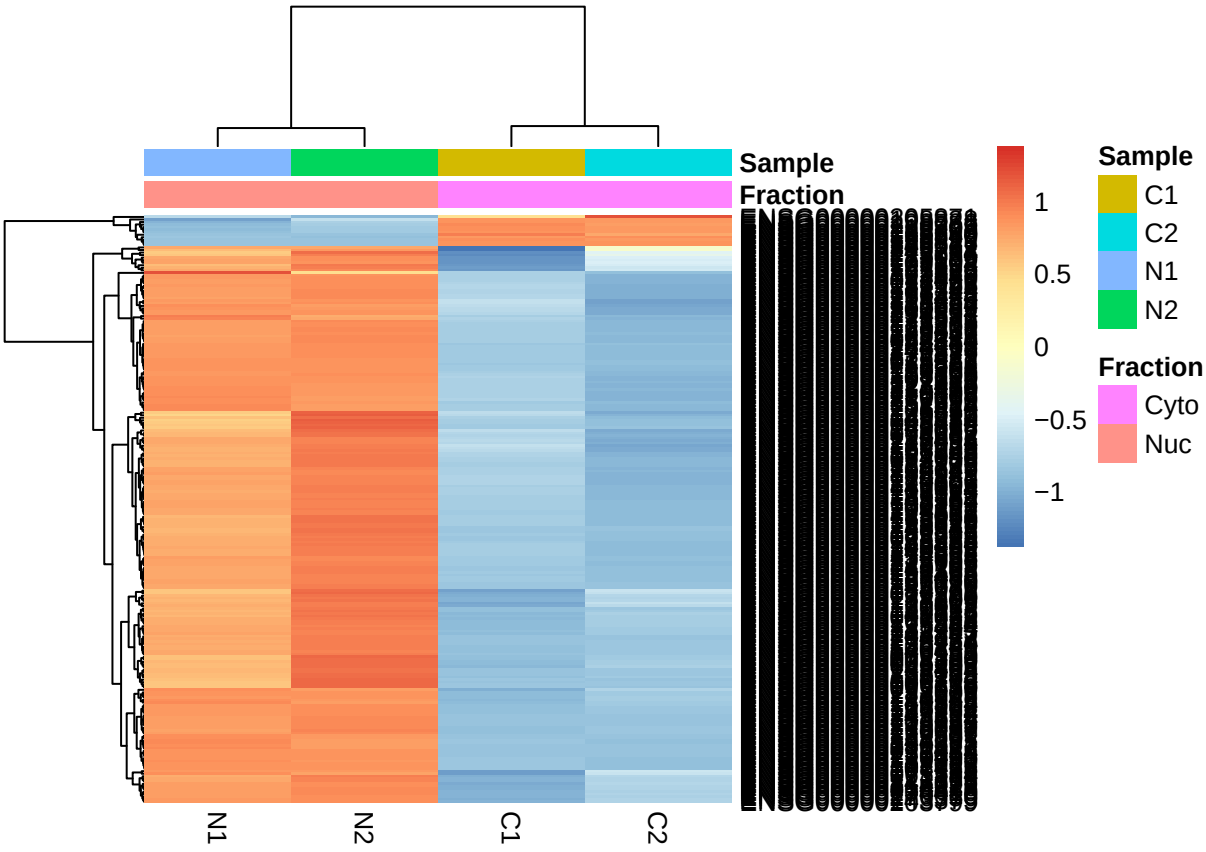
Es decir, fijamos un criterio de selección de genes y retenemos aquellos componentes de la tabla de resultados que lo cumplen. Por ejemplo: Genes con un p-valor ajustado inferior a 0.001 y un 'fold-change' superior a 6 o inferior a -6.

mapa de colores

```
for (i in  colnames(fit2$coefficients)){
  top=topTable(fit2,coef=i,sort="p", n=13522)
  genes=rownames(top[which(top$adj.P.Val<0.01 & abs(top$logFC)>3),])
  write.table(top,paste(i,"_limma_voom.txt",sep=""),quote=F)
  term1=strsplit(i,split="_vs_")[[1]][1]
  term2=strsplit(i,split="_vs_")[[1]][2]
  samples=rownames(subset(info,fraction==term1 | fraction==term2))
  expr=v$E[genes,samples]
  rownames(expr)=do.call(rbind, strsplit(genes, ','))[,1]
  if (length(genes) >1) {
    pdf(paste("pheatmap_DE_genes__01_",i,".pdf",sep=""), width = 10, height = 12)
    pheatmap(expr,scale="row",annotation_col=info[,c("Fraction","Sample")], border_color = "NA",show_ro
    dev.off()
  }}

write.table(v$E,"logcpm.txt",quote=F)
```

```
for (i in  colnames(fit2$coefficients)){
  top=topTable(fit2,coef=i,sort="p", n=13522)
  genes=rownames(top[which(top$adj.P.Val<0.01 & abs(top$logFC)>3),])
  write.table(top,paste(i,"_limma_voom.txt",sep=""),quote=F)
  term1=strsplit(i,split="_vs_")[[1]][1]
  term2=strsplit(i,split="_vs_")[[1]][2]
  samples=rownames(subset(info,fraction==term1 | fraction==term2))
  expr=v$E[genes,samples]
  rownames(expr)=do.call(rbind, strsplit(genes, ','))[,1]
  if (length(genes) >1) {
    pheatmap(expr,scale="row",annotation_col=info[,c("Fraction","Sample")], border_color = "NA",show_ro
  }}
```

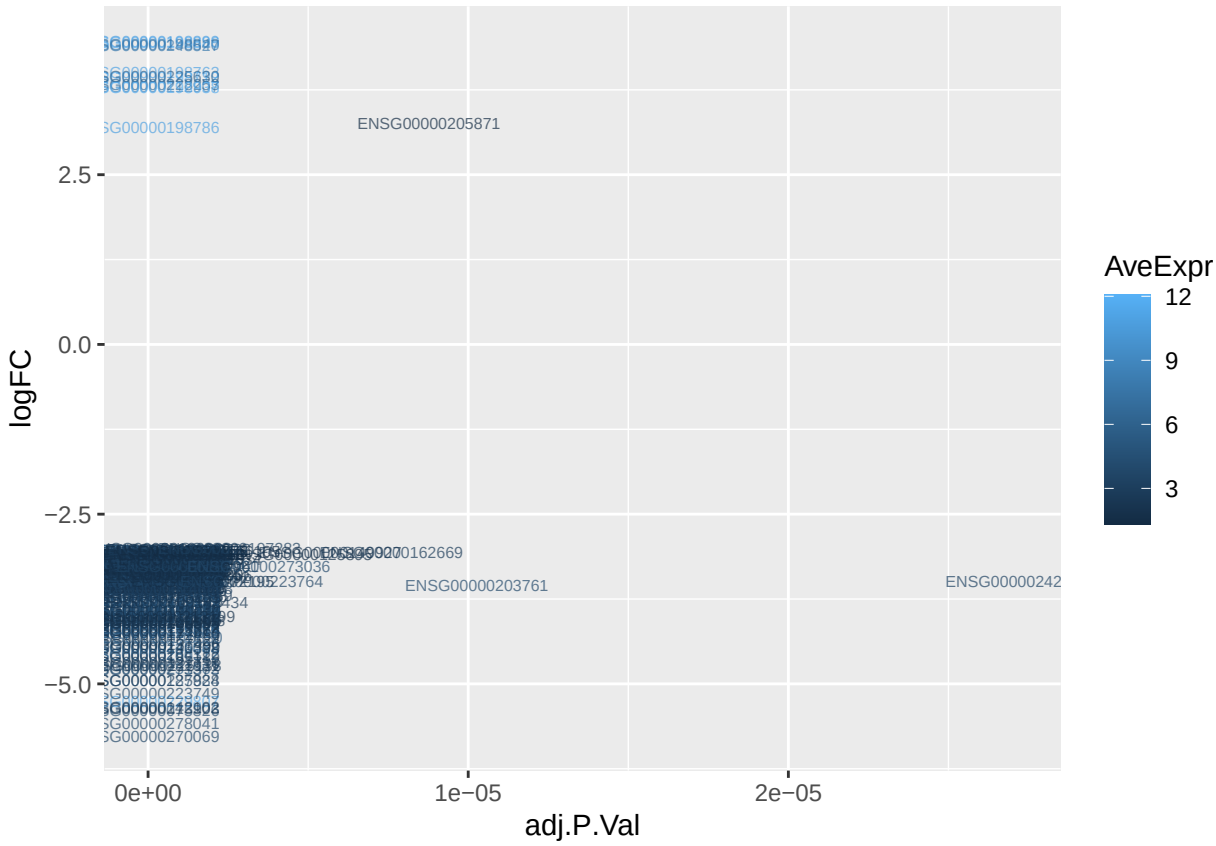



```
length(which(top$adj.P.Val < 0.01 & abs(top$logFC) > 3))
```

```
## [1] 231
```

```
p_data <-   top  %>% filter(adj.P.Val <0.01 & abs(logFC) > 3)

p_data %>%  ggplot(aes(x=adj.P.Val,y=logFC)) +
  geom_text(label=rownames(p_data), size=2.2,alpha=0.7, aes(col=AveExpr))
```



Top tables

```
top$Gene <- rownames(top)
DEGs <- top %>%  arrange(logFC) %>% filter(adj.P.Val <0.01 & abs(logFC) > 3)
head(DEGs)
```

```
##                       logFC   AveExpr         t      P.Value    adj.P.Val
## ENSG00000270069 -5.765019  3.608853 -18.91009 4.887407e-13 2.974701e-10
## ENSG00000278041 -5.582527  3.730863 -19.75468 2.360709e-13 1.995094e-10
## ENSG00000075826 -5.387170  4.214263 -20.79435 1.001480e-13 1.354201e-10
## ENSG00000213903 -5.362412  3.365680 -16.71929 3.739715e-12 1.233376e-09
## ENSG00000142102 -5.345629  2.798232 -16.12233 6.785876e-12 1.952311e-09
## ENSG00000229807 -5.249792 11.292743 -35.26005 1.271644e-17 1.719517e-13
##                        B            Gene
## ENSG00000270069 19.80722 ENSG00000270069
## ENSG00000278041 20.47648 ENSG00000278041
## ENSG00000075826 21.30725 ENSG00000075826
## ENSG00000213903 17.93917 ENSG00000213903
## ENSG00000142102 17.33107 ENSG00000142102
## ENSG00000229807 30.33582 ENSG00000229807
```

```
write.table(DEGs, file = "./DEG.txt", row.names = F, sep = "\t", quote = F)
```

```
#genes_sin_version <- sub("\\.\\d+$", "", rownames(top))
top$Gene <- rownames(top)
top <- top[,c("Gene", names(top)[1:6])]
write.table(top, file = "./Cyt_v_Nuc.txt", row.names = F, sep = "\t", quote = F)
```

Análisis de significació biológica

Nos centraremos únicamente en la lista de genes "up-regulados" y "down-regulados" es decir diferencialmente expresados con un logFC mayor que seis (más expresados en "cytosol" que en "nucleo").

Para el análisis de enriquecimiento utilizaremos la función `enrichGO` del paquete `clusterProfiler muy parecida a las de otros paquetes como`GOstats`.

```
library(org.Hs.eg.db)

head(top)
```

```
##                            Gene     logFC   AveExpr         t      P.Value
## ENSG00000229807 ENSG00000229807 -5.249792 11.292743 -35.26005 1.271644e-17
## ENSG00000198886 ENSG00000198886  4.446755 12.075137  31.48565 8.847266e-17
## ENSG00000198899 ENSG00000198899  4.469159 10.752527  30.88431 1.230280e-16
## ENSG00000198712 ENSG00000198712  3.931950 11.917380  27.56434 8.559871e-16
## ENSG00000198763 ENSG00000198763  4.004971 10.978200  26.85142 1.336721e-15
## ENSG00000248527 ENSG00000248527  4.406316  7.927375  26.43119 1.747655e-15
##                    adj.P.Val        B
## ENSG00000229807 1.719517e-13 30.33582
## ENSG00000198886 5.545282e-13 28.58356
## ENSG00000198899 5.545282e-13 28.25520
## ENSG00000198712 2.893664e-12 26.43302
## ENSG00000198763 3.615027e-12 25.99620
## ENSG00000248527 3.938631e-12 25.61553
```

```
allEntrezs <- rownames(top)
selectedEntrezsUP <- rownames(subset(top, (abs(logFC) > 3) & (adj.P.Val < 0.01)))
length(allEntrezs); length(selectedEntrezsUP)
```

```
## [1] 13522
```

```
## [1] 231
```

```
library(clusterProfiler)
library(org.Mm.eg.db)
ego <- enrichGO(gene = selectedEntrezsUP,
                universe = allEntrezs,
                keyType = "ENSEMBL",
                OrgDb = org.Hs.eg.db,
                ont = "BP",
                pAdjustMethod = "BH",
                qvalueCutoff = 0.01,
                readable = TRUE)
```

El objeto resultante almacena las categorías GO enriquecidas, los genes anotados en ellas y los valores de los estadísticos que llevan a afirmar que dichas categorías se encuentran significativamente sobre-representadas como resultado de un test de enriquecimiento.

```
head(ego)
```

```
##                    ID                                          Description
## GO:0030198 GO:0030198                       extracellular matrix organization
## GO:0045229 GO:0045229           external encapsulating structure organization
## GO:0043062 GO:0043062                     extracellular structure organization
## GO:0034220 GO:0034220                  monoatomic ion transmembrane transport
## GO:0098742 GO:0098742 cell-cell adhesion via plasma-membrane adhesion molecules
## GO:0007229 GO:0007229                      integrin-mediated signaling pathway
##            GeneRatio   BgRatio       pvalue     p.adjust       qvalue
## GO:0030198    39/744 147/10677 1.371147e-13 2.400176e-10 2.119671e-10
## GO:0045229    39/744 147/10677 1.371147e-13 2.400176e-10 2.119671e-10
## GO:0043062    39/744 148/10677 1.740940e-13 2.400176e-10 2.119671e-10
## GO:0034220    79/744 492/10677 1.077948e-12 1.114598e-09 9.843365e-10
## GO:0098742    29/744  92/10677 1.743599e-12 1.442305e-09 1.273745e-09
## GO:0007229    25/744  70/10677 2.584351e-12 1.781479e-09 1.573280e-09
##
## GO:0030198
## GO:0045229
## GO:0043062
## GO:0034220 ND4/ATP6/COX2/ND5/COX1/CYTB/ATP8/FLNA/AHNAK/COX3/UTRN/HCN3/SLC26A6/SLC25A27/ATP2A2/CATSPE
## GO:0098742
## GO:0007229
##            Count
## GO:0030198    39
## GO:0045229    39
## GO:0043062    39
## GO:0034220    79
## GO:0098742    29
## GO:0007229    25
```

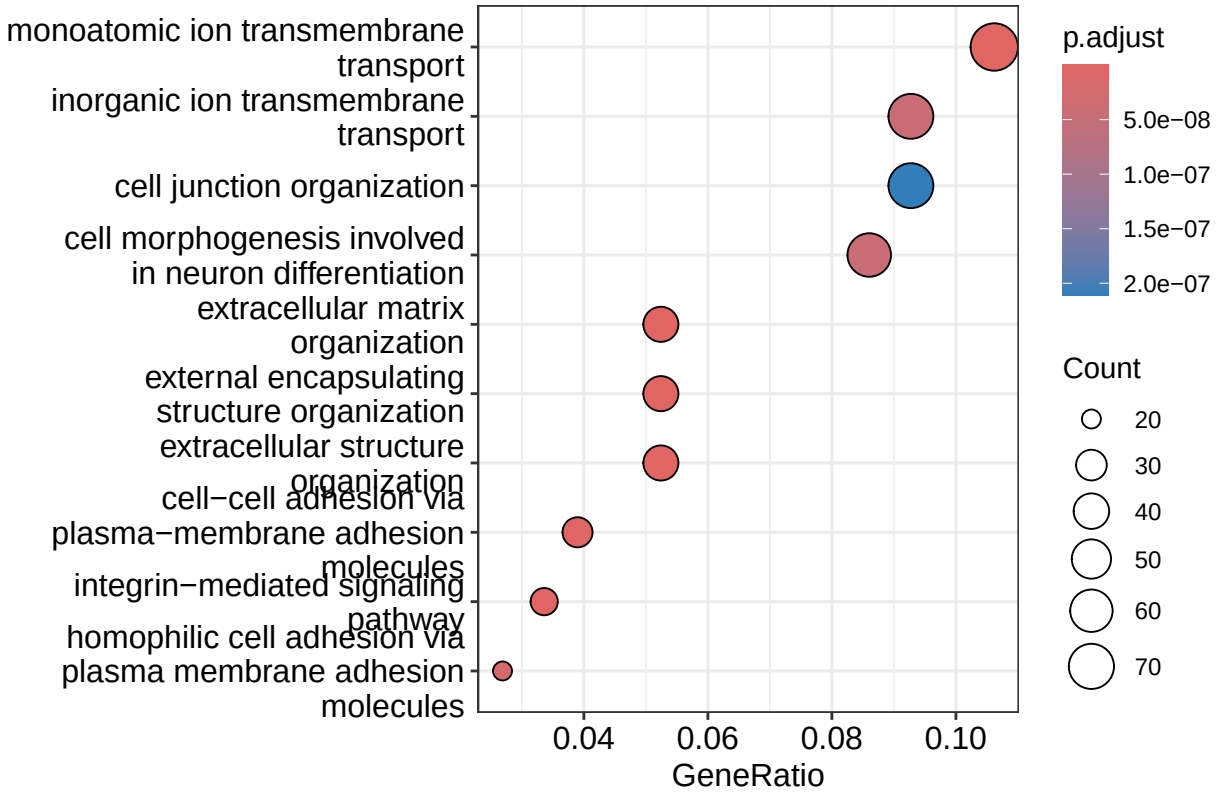
```
ego_results <- data.frame(ego)
write.csv(ego_results, "clusterProfiler_ORAresults_UpGO.csv")
```

Visualización de los resultados del análisis de enriquecimiento

Uno de los aspectos interesantes del paquete `clusterProfiler` es que permite visualizar los resultados mediante algunos gráficos creados específicamente para tal fin.

##Dotplot de los 9 términos más enriquecidos Este gráfico compara visualmente las categorías enriquecidas (de más a menos enriquecidas) visualizando simultáneamente cuan enriquecidas estan y el p-valor del test de enriquecimiento.

```
dotplot(ego, showCategory=10)
```




```
pdf(paste("dotplot.pdf",sep=""))
dotplot(ego, showCategory=10)
dev.off()
```

```
## pdf
##   2
```

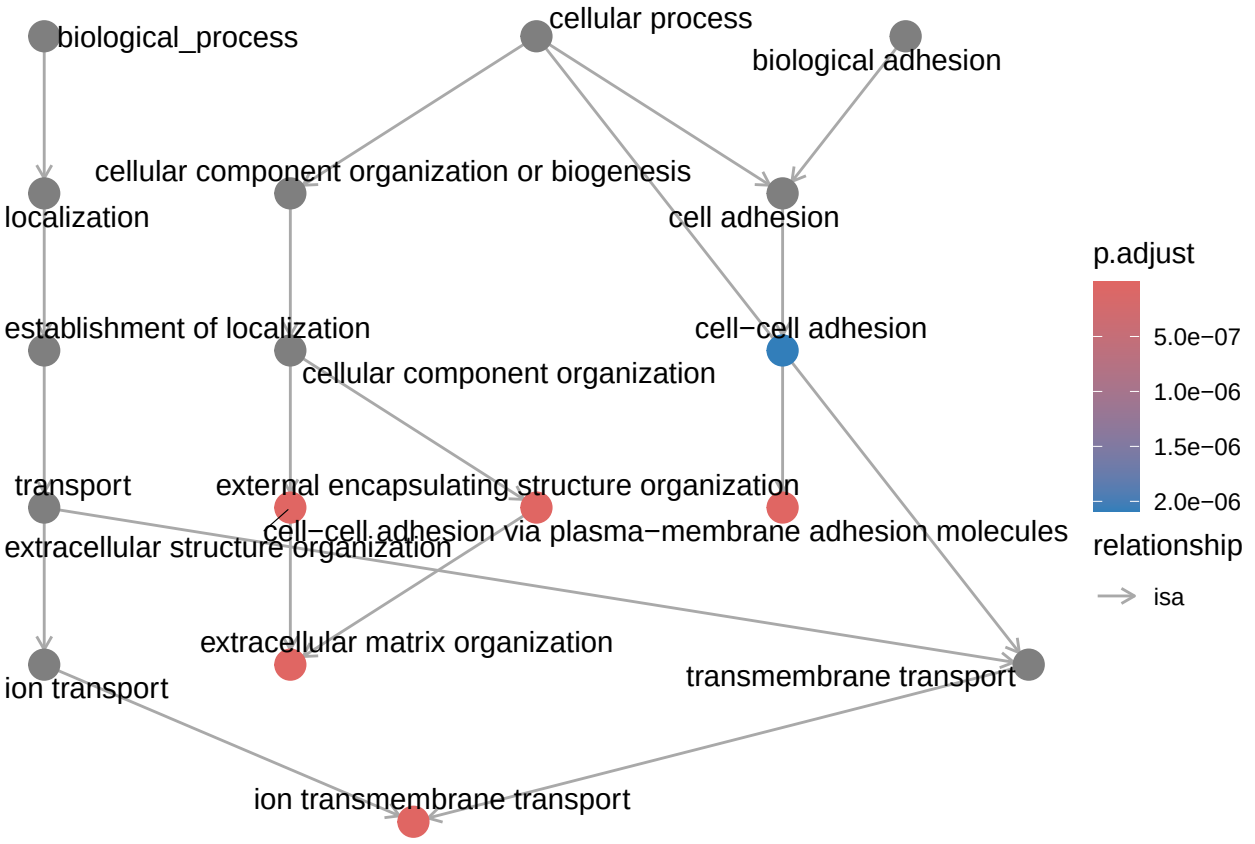
Visualización jerárquica de los términos GO

Este gráfico permite visualizar los términos seleccionados dentro del sub-grafo de la GO que los contiene. Esto nos, permite por ejemplo, hacernos una idea de si estan muy dispersos, o no, en la jerarquía y de si se trata de términos muy generales o más específicos.

```
pdf(paste("GO.pdf",sep=""))
goplot(ego, showCategory=5, cex=0.5)
dev.off()
```

```
## pdf
##   2
```

```
goplot(ego, showCategory=5, cex=0.5)
```

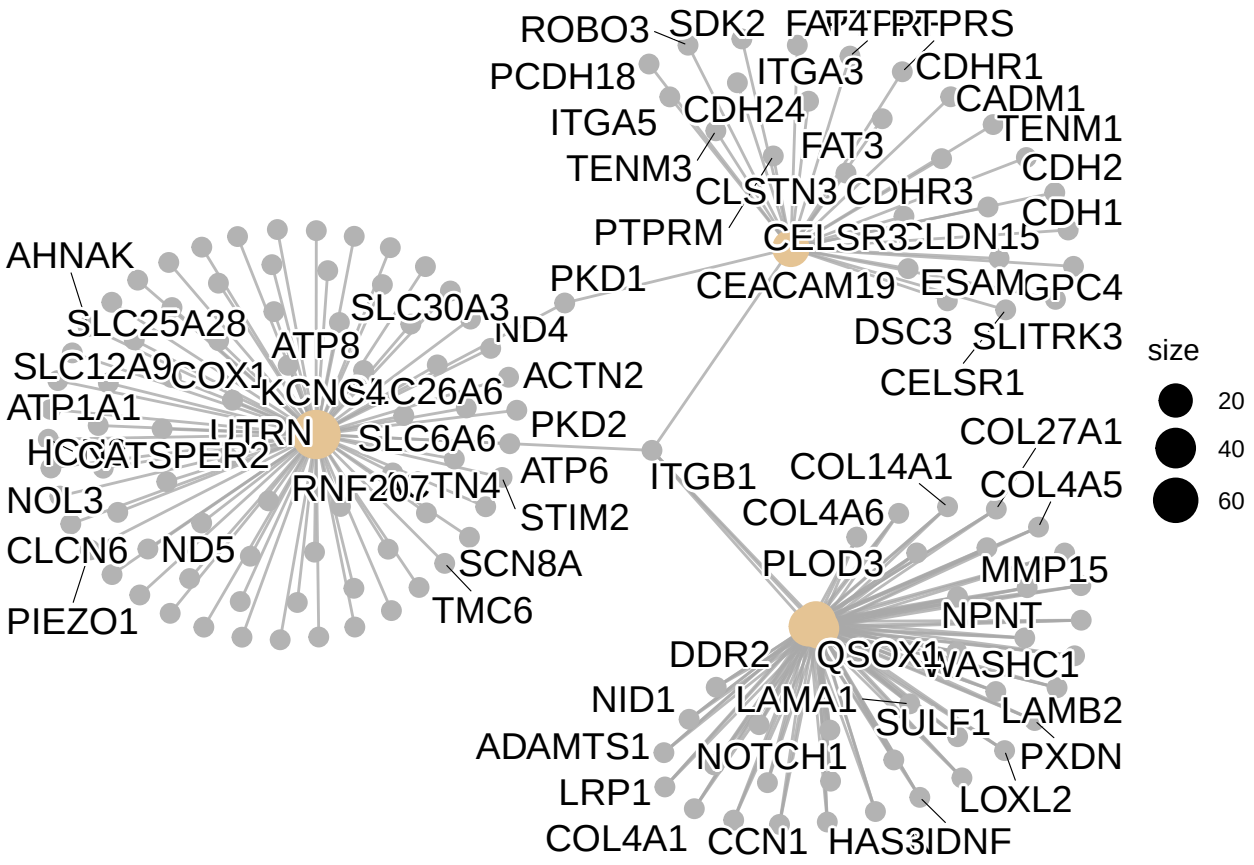


De forma parecida una red de genes nos permite visualizar la asociación entre los genes y las categorías seleccionadas en las que éstos genes estan anotados.

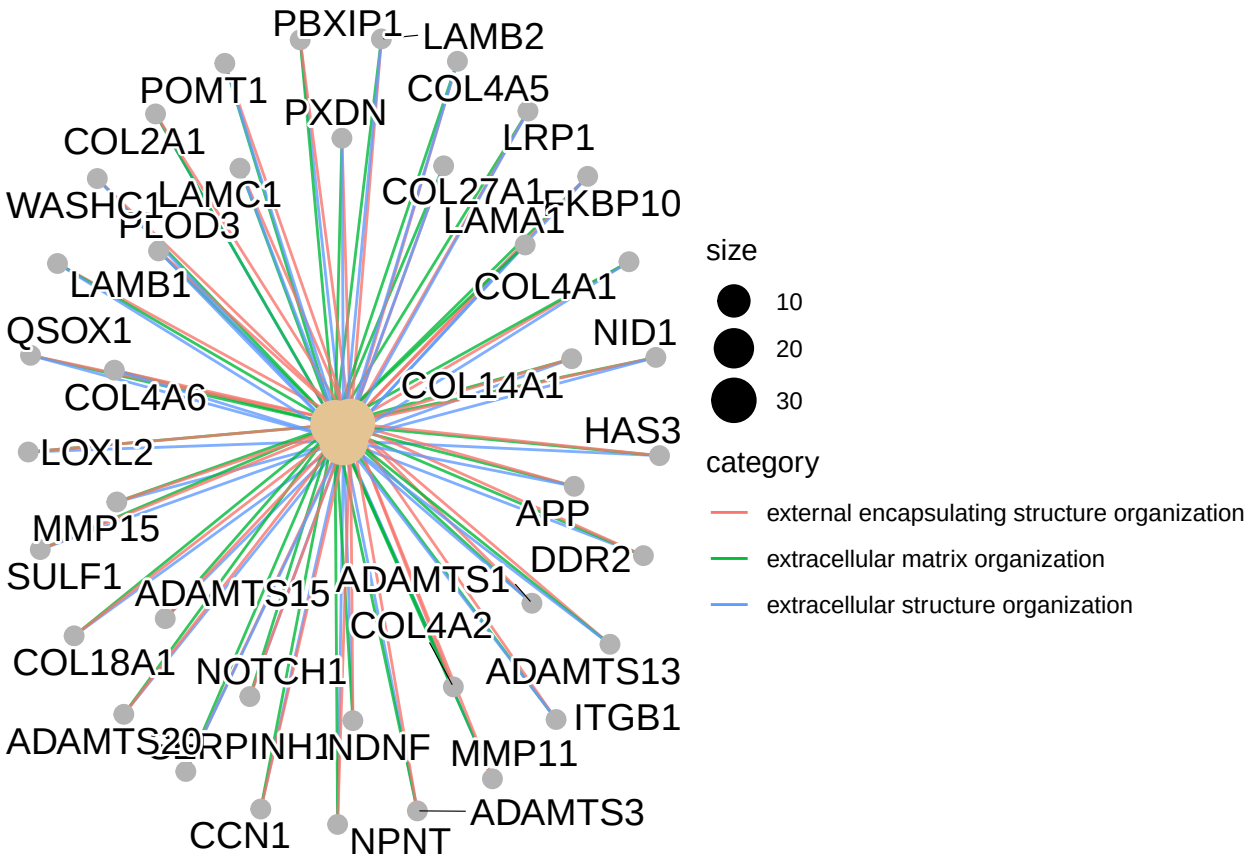
```
## Gene network para los términos seleccionados
pdf(paste("cneplot.pdf",sep=""))
cnetplot(ego)
dev.off()
```

```
## pdf
##   2
```

```
cnetplot(ego)
```




```
library(clusterProfiler)
library(ggplot2)
ego2 = clusterProfiler::simplify(ego, cutoff = 0.01, by = "p.adjust")
png("./cnetplot_transp.png", units = "in", width = 24, height = 16, res = 600,
    bg = "transparent")
par(bg = NA)
a <- cnetplot(ego2, showCategory = 5, cex_category = 1, cex_label_category = 2.5,
    cex_gene = 1, cex_label_gene = 1, circular = FALSE, colorEdge = TRUE)
a
invisible(dev.off())
a
```

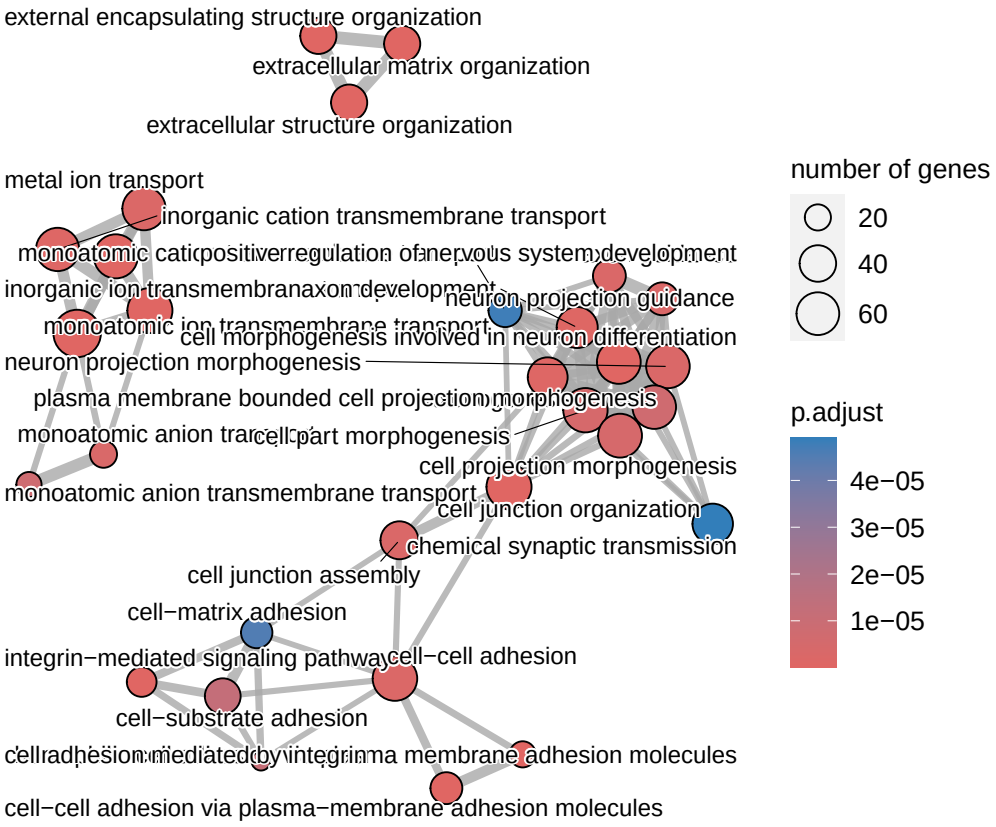


Finalmente este gráfico permite simplificar las visualizaciones y agrupa los 104 términos más significativos basándose en alguna medida de similaridad entre los mismos (por ejemplo "similaridad semántica" definida a partir de su interdistancia dentro del grafo).

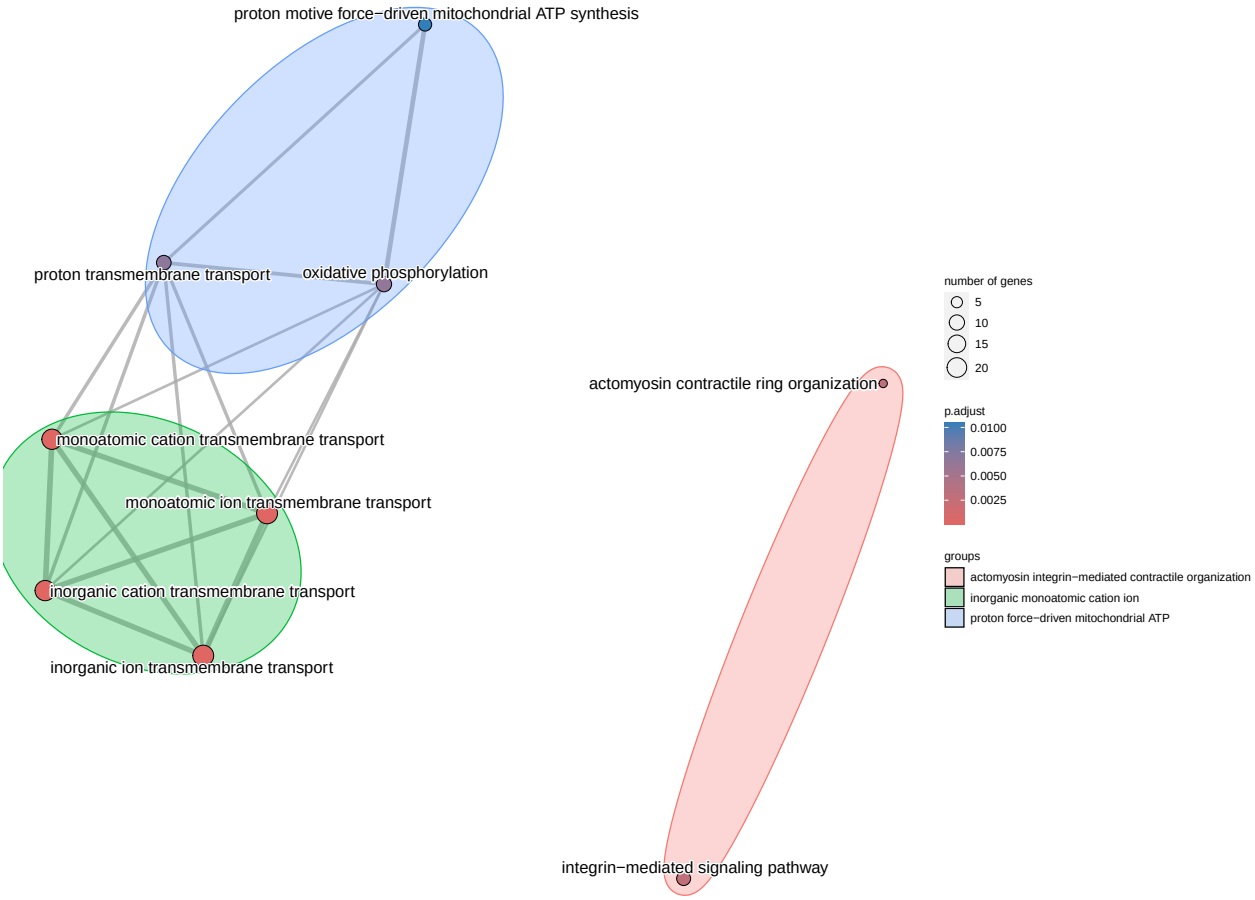
```
## Enrichment Map
library(enrichplot)
ego_sim <- pairwise_termsim(ego)
pdf(paste("emaplot.pdf",sep=""))
emapplot(ego_sim, cex_label_category=0.6)
dev.off()
```

```
## pdf
##   2
```

```
emapplot(ego_sim, cex_label_category=0.6)
```



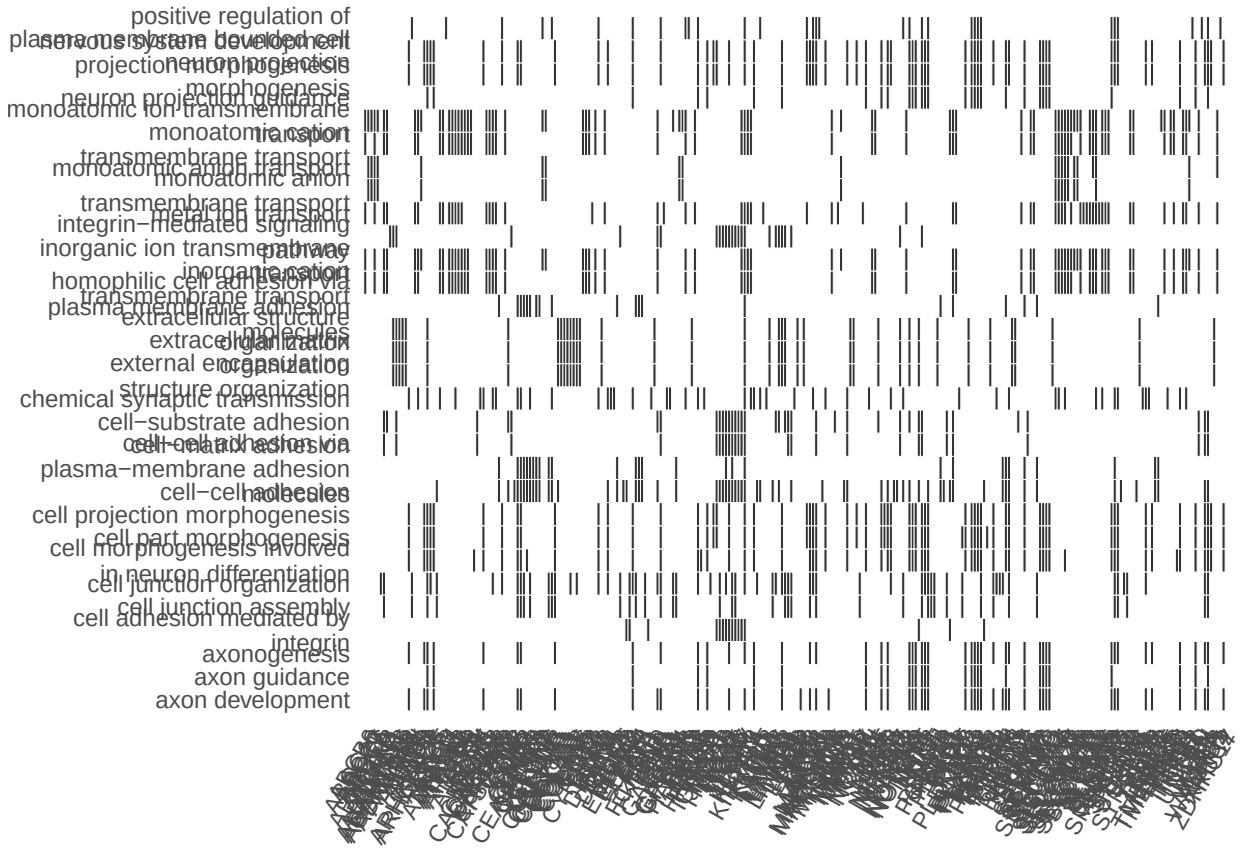

```
term_similarity_matrix = pairwise_termsim(ego)
emapplot(term_similarity_matrix, showCategory = 15, group_category = TRUE, group_legend = TRUE)
```




```
pdf(paste("emaplot_grouped.pdf",sep=""),width = 13, height = 15)
emapplot(term_similarity_matrix, showCategory = 15, group_category = TRUE, group_legend = TRUE)
dev.off()
```

```
## pdf
##   2
```

```
library(enrichplot)
heatplot(ego)
```




```
library('AnnotationDbi')
library('org.Hs.eg.db')
ensembl_codes <- gsub("\\..*", "", row.names(res_SFIvsNIT))
# Add symbol column
res_SFIvsNIT$symbol <- mapIds(org.Hs.eg.db,
                     keys = ensembl_codes,
                     column = "SYMBOL",
                     keytype = "ENSEMBL",
                     multiVals = "first")

# Add Entrez ID
res_SFIvsNIT$entrez <- mapIds(org.Hs.eg.db,
                     keys = ensembl_codes,
                     column = "ENTREZID",
                     keytype = "ENSEMBL",
                     multiVals = "first")

# Add GENENAME
res_SFIvsNIT$genename <- mapIds(org.Hs.eg.db,
                     keys = ensembl_codes,
                     column = "GENENAME",
                     keytype = "ENSEMBL",
                     multiVals = "first")

res_SFIvsNIT <- res_SFIvsNIT[order(res_SFIvsNIT$padj), ]
```

“‘