Bioinformatics project

Virginia Leombruni

2024-11-06

Introduction

The analysis was conducted on the RData file `Breast_invasive_carcinoma`, in which the following three data frames are available:

1. `raw_counts_df` = contains raw RNA-seq counts;
2. `c_year_df` = contains sample names and conditions (case or control);
3. `r_year_df` = contains ENSEMBL gene IDs, lengths and gene symbols.

Task 1

```
# Load the RData file
load("C:/Users/virgi/Desktop/Breast_invasive_carcinoma.RData")
```

Task 2

The first step was to update `raw_count_df` and `r_anno_df` by extracting only the coding genes. This process required connecting to the Ensembl database. The getBM function from the biomaRt package was used to perform the query, specifying the attributes and providing the values from `r_anno_df`.

The results were then filtered to retain only protein-coding genes.

Finally, `raw_count_df` and `r_anno_df` were filtered to include only the rows corresponding to these genes, and the dimensions of both datasets were checked to confirm the filtering.

```
library(biomaRt)
library(tidyverse)
```

```
ensembl <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")
filters <- listFilters(ensembl)
query1 <- getBM(attributes=c("ensembl_gene_id",
                             "external_gene_name",
                             "gene_biotype",
                             "start_position",
                             "end_position",
                             "chromosome_name",
                             "strand",
                             "description",
                             "version"),
                filters=c("ensembl_gene_id"),
                values=list(r_anno_df$ensembl_gene_id),
                mart = ensembl)

query1_protein_coding <- query1[which(query1$gene_biotype=="protein_coding"),]

#filter the database to keep only protein-coding-genes
raw_counts_df_filt <- raw_counts_df[query1_protein_coding$ensembl_gene_id,]
raw_counts_df_filt <- raw_counts_df[which(rownames(raw_counts_df)
                                          %in%
                                            query1_protein_coding$ensembl_gene_id),]

r_anno_df_filt <- subset(r_anno_df,
                         ensembl_gene_id %in% query1_protein_coding$ensembl_gene_id)

#check the dimensions
dim(raw_counts_df)
```

```
[1] 62872   100
```

```
dim(r_anno_df)
```

```
[1] 62872     3
```

```
dim(raw_counts_df_filt)
```

```
[1] 22153   100
```

```
dim(r_anno_df_filt)
```

```
[1] 22153     3
```

The results on the dimensions confirm that the filtering process was applied consistently across both data frames.

Task 3

Differential expression analysis

Differential expression analysis was performed using the `edgeR` package. Initially, the raw count data were filtered to retain only genes with a raw count >20 in at least 5 cases or 5 control samples. Then, a DGElist was created, and the analysis selected up- and down-regulated genes using an adjusted p-value cutoff of 0.01, a log fold change ratio >1.5 for up-regulated genes and < -1.5 for down-regulated genes and a log CPM >1.

```
count_thr <- 20
repl_thr <- 5

#filter the datasets
filter_vec <- apply(raw_counts_df_filt, 1,
                    function(y) max(by(y,
                                       c_anno_df$condition, function(x) sum(x > count_thr))))

#table(filter_vec)

filter_counts_df <- raw_counts_df_filt[filter_vec >= repl_thr,]
dim(filter_counts_df)
```

```
[1] 17182   100
```

```
# apply the filter on gene annotation
filter_anno_df <- r_anno_df_filt[rownames(filter_counts_df),]

## DEG analysis with edgeR
```

```
library(edgeR)
```

```
edge_c <- DGEList(counts = filter_counts_df,
                  group = c_anno_df$condition,
                  samples = c_anno_df,
                  genes = filter_anno_df)
#edge_c

#normalization
edge_n <- calcNormFactors(edge_c)
#edge_n

cpm_table <- as.data.frame(round(cpm(edge_n), 2))

# define the experimental design matrix
design <- model.matrix(~ 0 + group, data = edge_n$samples)
colnames(design) <- levels(edge_n$samples$group)
```

```
rownames(design) <- edge_n$samples$sample

# calculate dispersion and fit with edgeR
edge_d <- estimateDisp(edge_n,design)
edge_f <- glmQLFit(edge_d,design)

# definition of the contrast (conditions to be compared)
contro <- makeContrasts("case-control", levels = design)

# fit the model with generalized linear models
edge_t <- glmQLFTest(edge_f,contrast=contro)
DEGs <- as.data.frame(topTags(edge_t,n=20000,p.value = 0.01,sort.by = "logFC"))
DEGs$class <- "="
DEGs$class[which(DEGs$logCPM>1&DEGs$logFC>1.5)] = "+"
DEGs$class[which(DEGs$logCPM > 1 & DEGs$logFC < (-1.5))] = '-'
DEGs <- DEGs[order(DEGs$logFC,decreasing = T),]

#View(DEGs)
table(DEGs$class)
```

```
   -    +    =
 904  915 8603
```

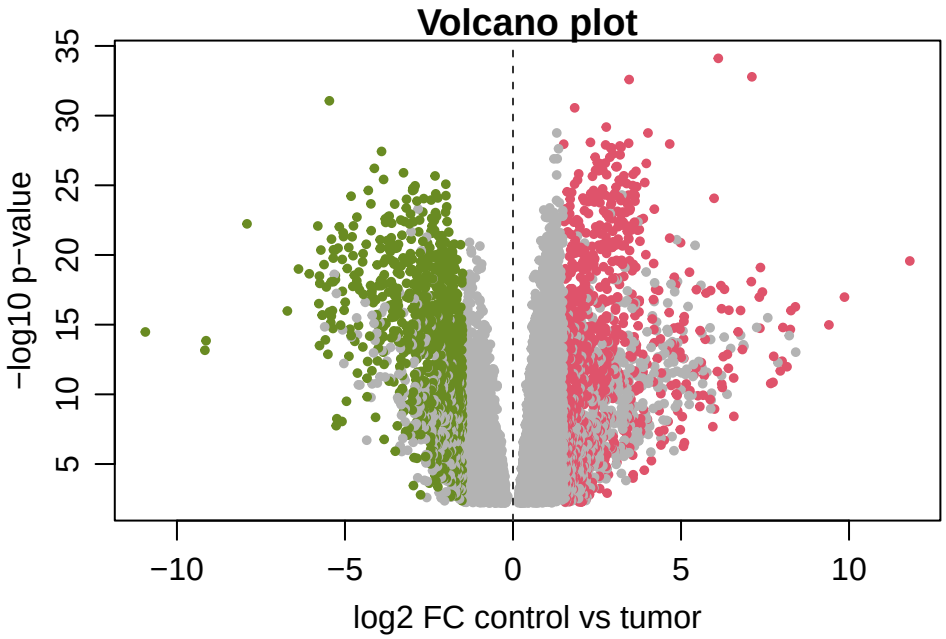
The number of up-regulated (915) and down-regulated (903) genes is fairly balanced, suggesting a similar number of genes are activated and repressed under the experimental conditions. Most genes (8600) do not show significant changes in expression, which is typical in differential expression analyses. This indicates that while certain groups of genes are affected by the conditions, the overall gene activity remains mostly unchanged.

Following, a volcano plot and a heatmap were created from the `DEGs` data frame.

The VOLCANO PLOT

```
library(edgeR)
```

```
input_df <- DEGs
xlabel <- "log2 FC control vs tumor"
ylabel <- "-log10 p-value"
par(fig=c(0,1,0,1), mar=c(4,4,1,2), mgp=c(2, 0.75, 0))
plot(input_df$logFC, -log(input_df$PValue,base=10),xlab=xlabel, ylab=ylabel,
     col=ifelse(input_df$class=="=","grey70",
                ifelse(DEGs$class == "+",
                       "#DF536B", "olivedrab4")),
     pch=20, frame.plot=TRUE, cex=0.8, main="Volcano plot")
abline(v=0,lty=2,col="grey20")
```



The volcano plots suggest that there are several genes with significant differential expression between the control and case conditions:

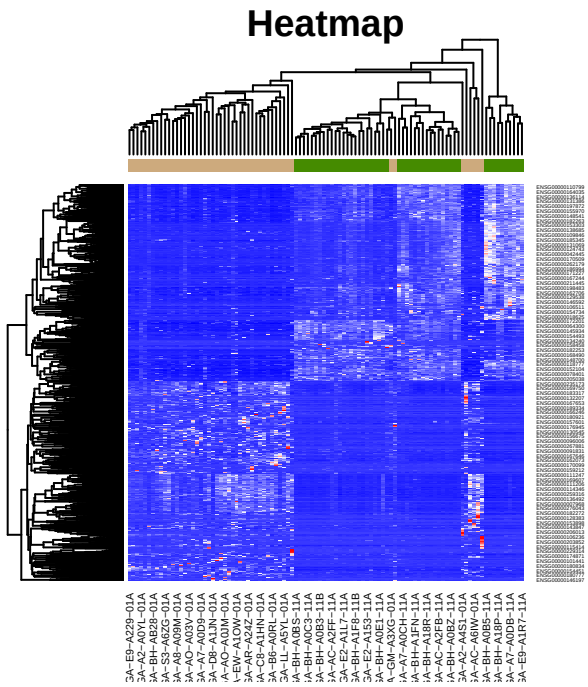
- The red points on the right side suggest several genes are significantly up-regulated in the case group.
- The green points on the left side indicate several genes are significantly down-regulated in the case group.

The plot clearly shows which genes have the biggest and most significant changes, helping to focus on these genes for further study.

The HEATMAP

```
col <- rep('chartreuse4', nrow(c_anno_df))
col[which(c_anno_df$condition == 'case')] <- 'burlywood3'

pal <- colorRampPalette(c('blue', 'white', 'red'))(100)

# Subset and scale the data

subset_cpm <- as.matrix(cpm_table[which(rownames(cpm_table)
                                        %in%DEGs$ensembl_gene_id
                                        [which(DEGs$class != '=')]),])


subset_cpm <- as.matrix(cpm_table[which(rownames(cpm_table) %in% DEGs$ensembl_gene_id[which(DEGs$
scaled_subset_cpm <- t(scale(t(subset_cpm)))

# Generate a heatmap using the base heatmap function
heatmap(scaled_subset_cpm,
        ColSideColors = col,
```

```
        col = pal,
        margins = c(4, 4),
        cexCol = 0.5,
        cexRow = 0.2,
        main = "Heatmap")
```



As evident from the results, the gene expression of most genes in the heatmap remains unchanged.

Task 4

Gene set enrichment analysis

In the beginning, the `biomaRt` package was again used to annotate the genes in DEGs with information from the Ensembl database(the `exInternal_gene_ name`, `enesembl_gene_id` and `enterzgene_id`). Two different datasets are then created based on the genes that are up-regulated (`up_DEGs`) and down-regulated (`down_DEGs`). Finally, gene set enrichment analysis was conducted, focusing on the Biological Process.

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

```
ensembl <- useEnsembl(biomart = "ensembl",dataset = "hsapiens_gene_ensembl")
```

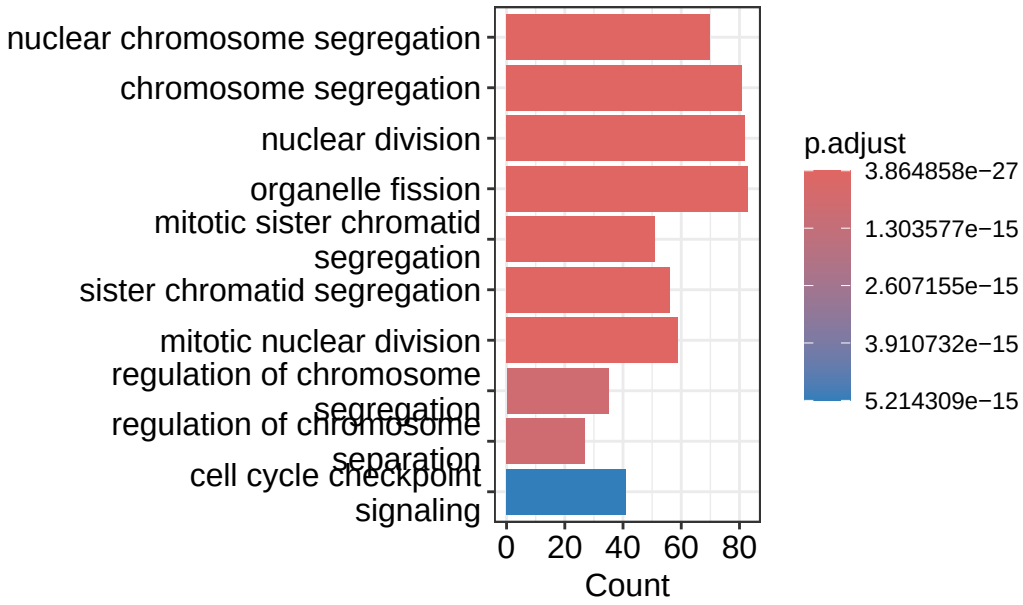
```
Ensembl site unresponsive, trying asia mirror
```

```
convert <- getBM(attributes=c("ensembl_gene_id","entrezgene_id","external_gene_name"),
                 filters=c("ensembl_gene_id"),
                 values= DEGs$ensembl_gene_id,
                 mart = ensembl)

DEGs <- merge(DEGs,convert,by.x="ensembl_gene_id",by.y="ensembl_gene_id")

DEGs <- DEGs[which(!is.na(DEGs$entrezgene_id)),]
DEGs <- DEGs[-which(duplicated(DEGs$entrezgene_id)),]

# Splitting up and down regulated ones
up_DEGs <- DEGs[DEGs$class == "+",]
down_DEGs <- DEGs[DEGs$class == "-",]

# Perform Gene Ontology enrichment analysis
up_ego_BP <- enrichGO(gene = up_DEGs$external_gene_name.x,
                   OrgDb = org.Hs.eg.db,
                   keyType = 'SYMBOL',
                   ont = "BP",
                   pAdjustMethod = "BH",
                   pvalueCutoff = 0.05,
                   qvalueCutoff = 0.05)

down_ego_BP <- enrichGO(gene = down_DEGs$external_gene_name.x,
                   OrgDb = org.Hs.eg.db,
                   keyType = 'SYMBOL',
                   ont = "BP",
                   pAdjustMethod = "BH",
                   pvalueCutoff = 0.05,
                   qvalueCutoff = 0.05)
```

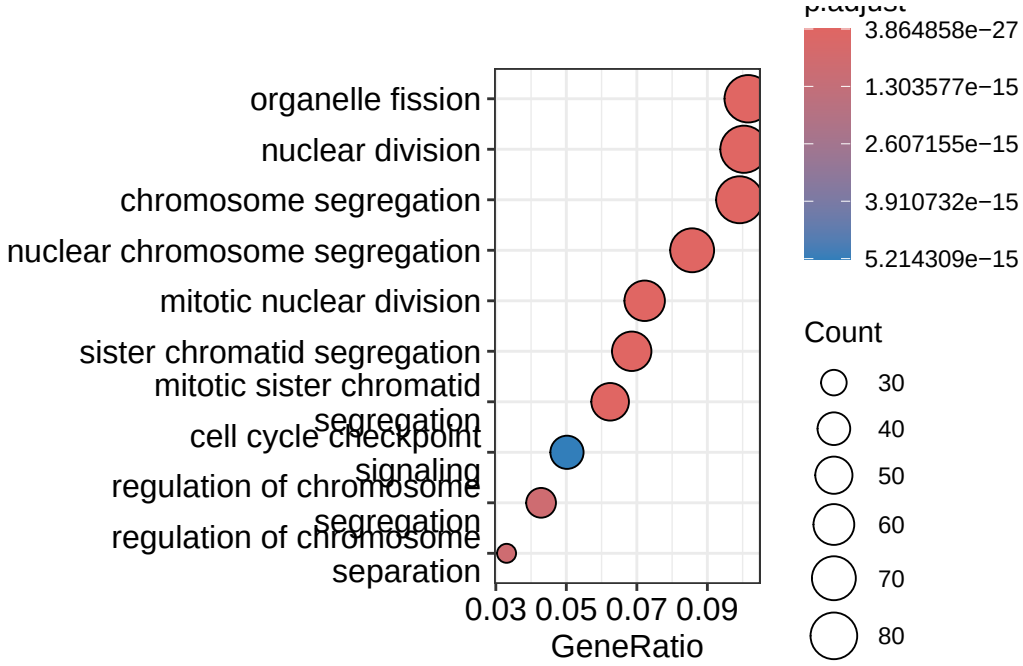
The following graphs visualize the top 10 enriched biological processes that were previously identified through gene enrichment analysis, using the barplot, the dotplot and the heatplot. Both up-regulated and down-regulated processes are represented. The first chunk shows plots of the up-regulated **process**, while the second set graphs the down-regulated genes.

PLOT UP-REGULATED

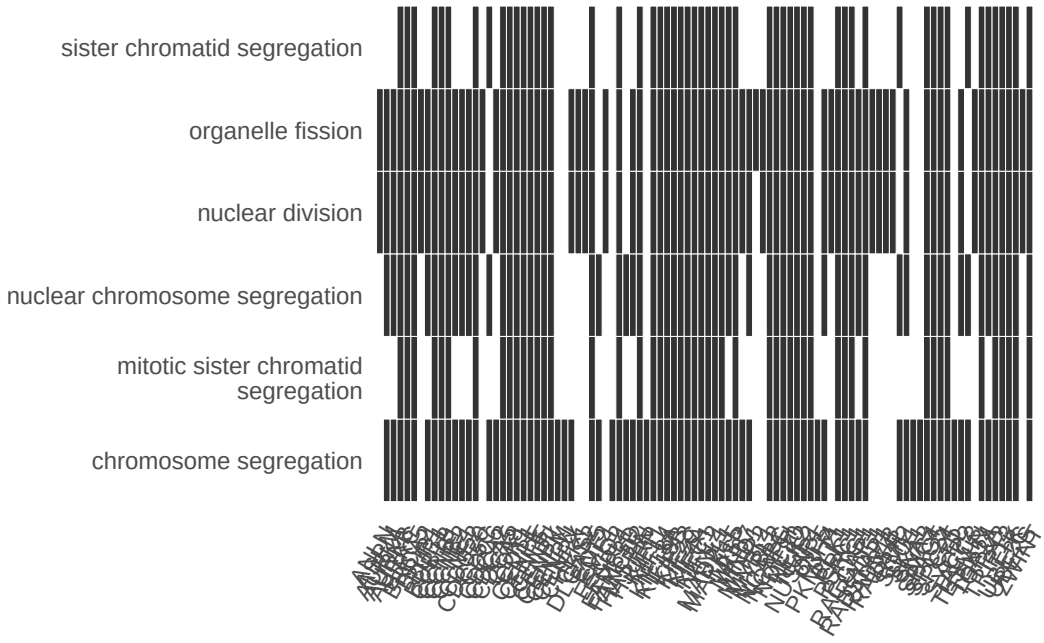
```
barplot(up_ego_BP,showCategory=10)
```




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

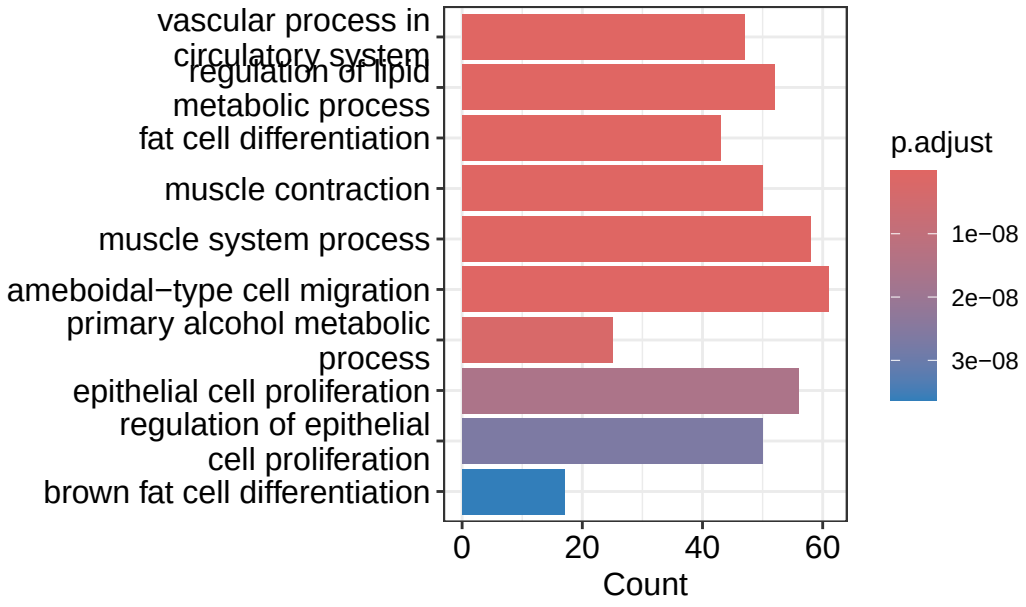



```
heatplot(up_ego_BP,showCategory = 6)
```

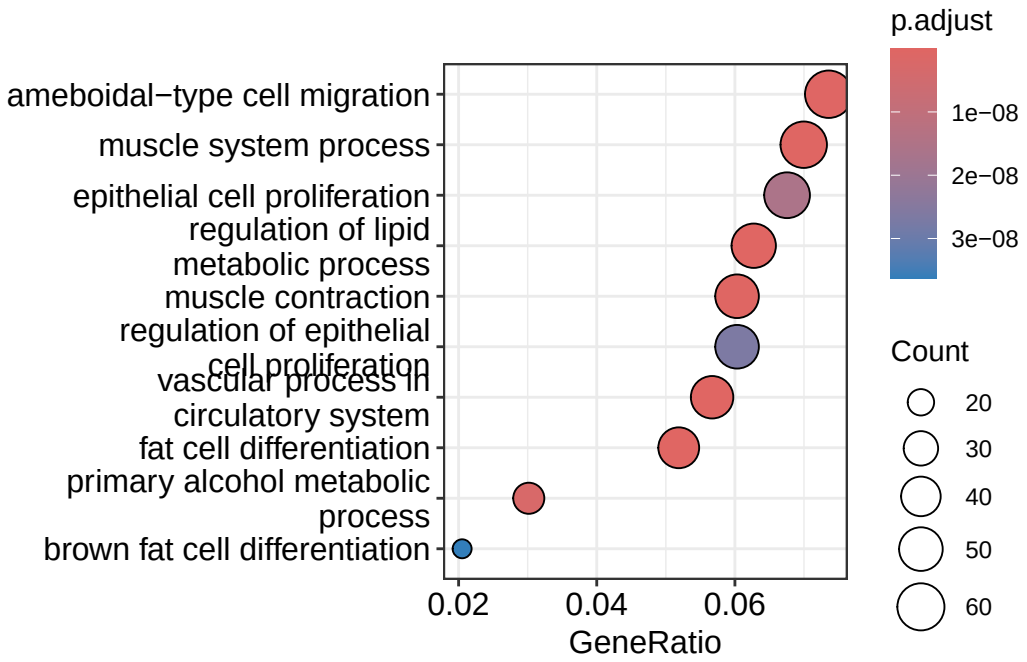


PLOT DOWN-REGULATED

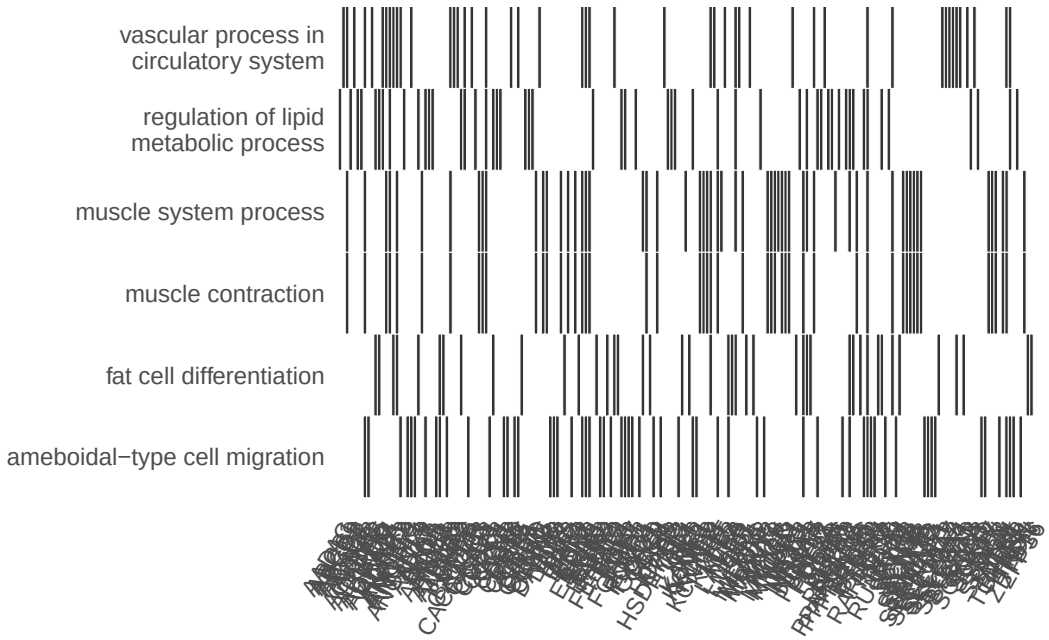
```
barplot(down_ego_BP,showCategory=10)
```




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




```
heatplot(down_ego_BP,showCategory = 6)
```

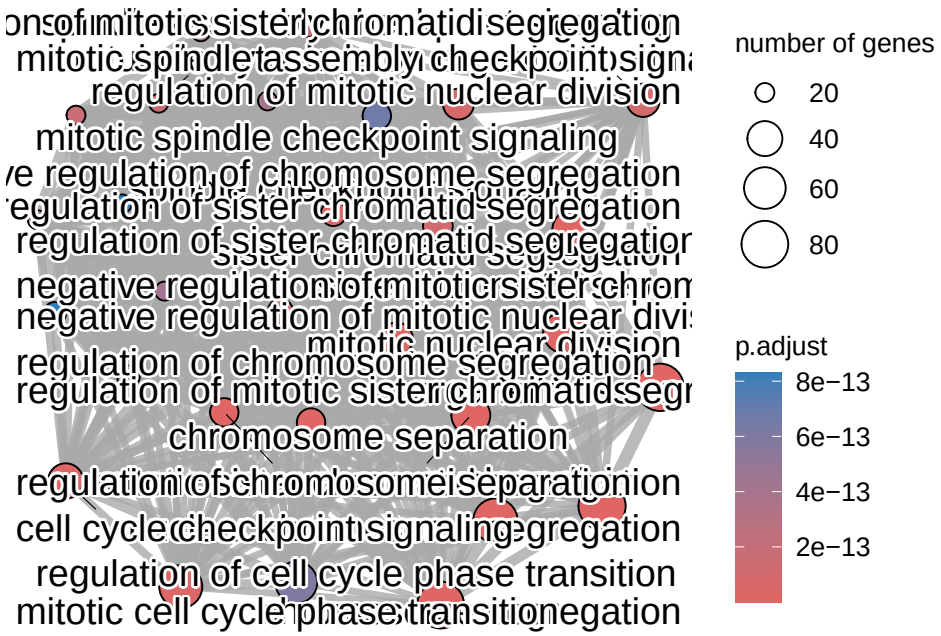


The significance cutoff used to identify enriched terms was set at a false discovery rate (FDR) of 0.05, as specified by the `pAdjustMethod = "BH"` argument in the `enrichGO()` function. This indicates that adjusted p-values or q-values below 0.05 were considered statistically significant.

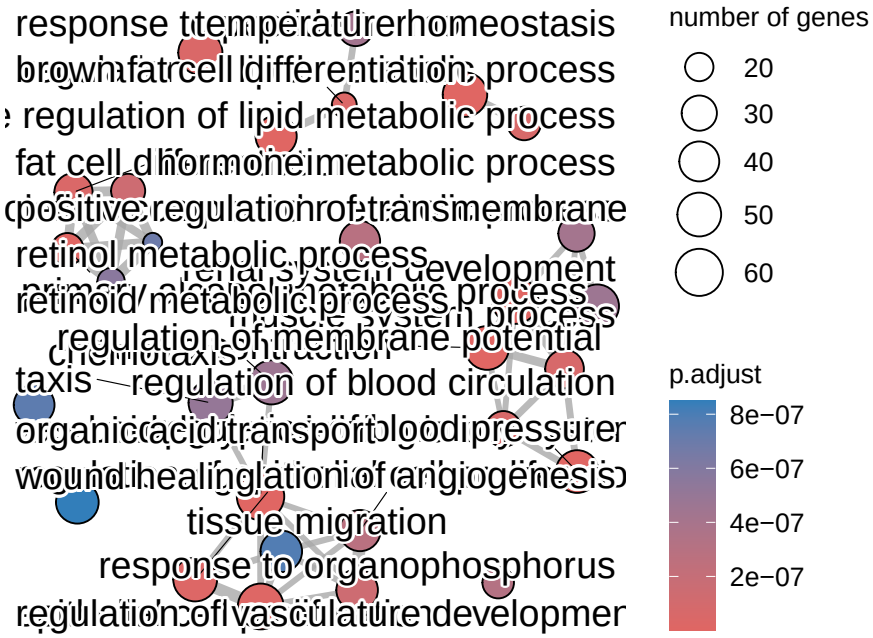
To visualize the relationships among the previously enriched biological process, the `pairwise_termsim` function is used for both up and down-regulated.

```
library(enrichplot)
```

```
x2 <- pairwise_termsim(up_ego_BP)
emapplot(x2)
```



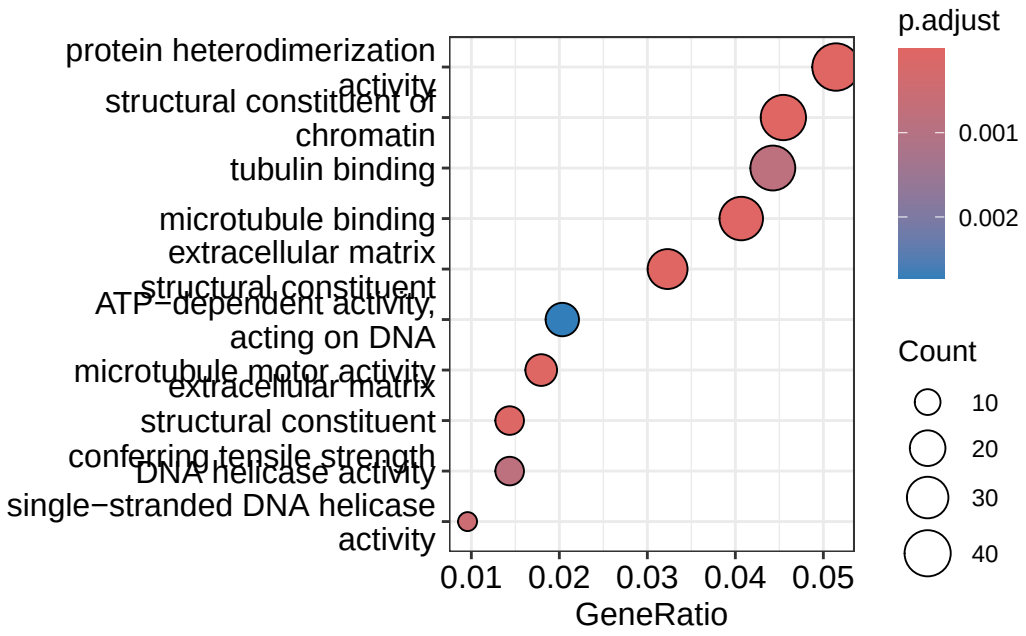

```
x3 <-  pairwise_termsim(down_ego_BP)
emapplot(x3)
```



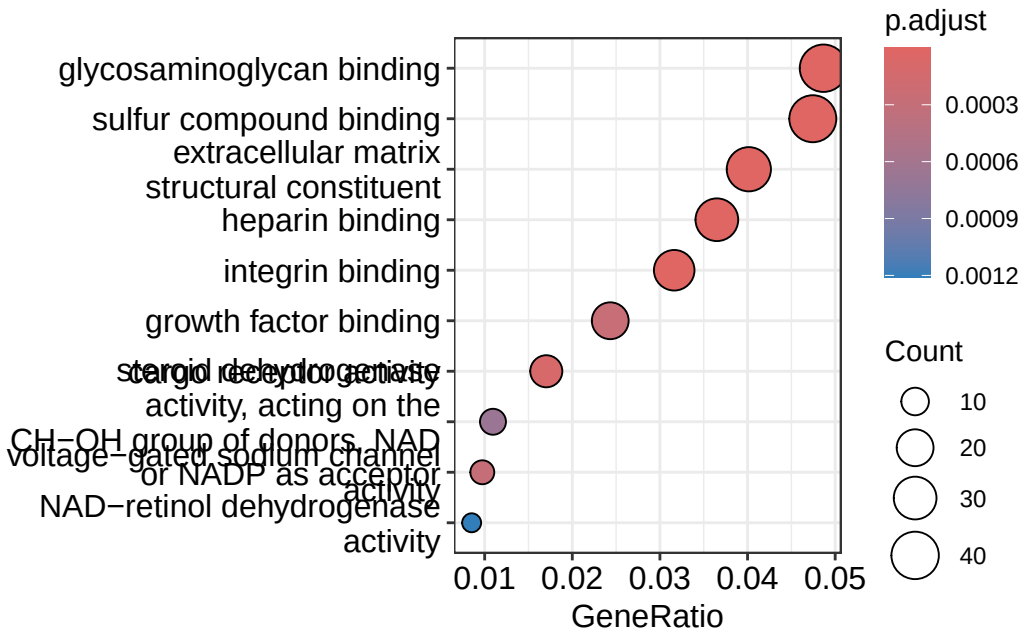
MF (Molecular Function)

The same code as previously is repeated, but this time performing the GO enrichment analysis considering the Molecular Function (MF). Then, the results are plotted.

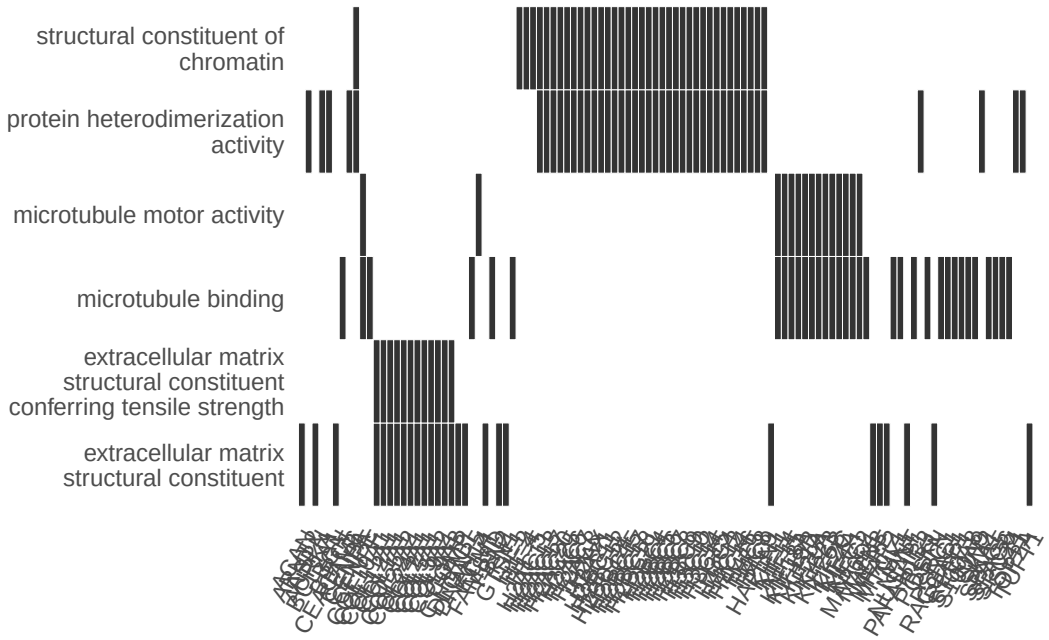
```r
up_ego_MF <- enrichGO(gene = up_DEGs$external_gene_name.x,
                      OrgDb = org.Hs.eg.db,
                      keyType = 'SYMBOL',
                      ont = "MF",
                      pAdjustMethod = "BH",
                      pvalueCutoff = 0.05,
                      qvalueCutoff = 0.05)

down_ego_MF <- enrichGO(gene = down_DEGs$external_gene_name.x,
                        OrgDb = org.Hs.eg.db,
                        keyType = 'SYMBOL',
                        ont = "MF",
                        pAdjustMethod = "BH",
                        pvalueCutoff = 0.05,
                        qvalueCutoff = 0.05)

dotplot(up_ego_MF, showCategory = 10)
```



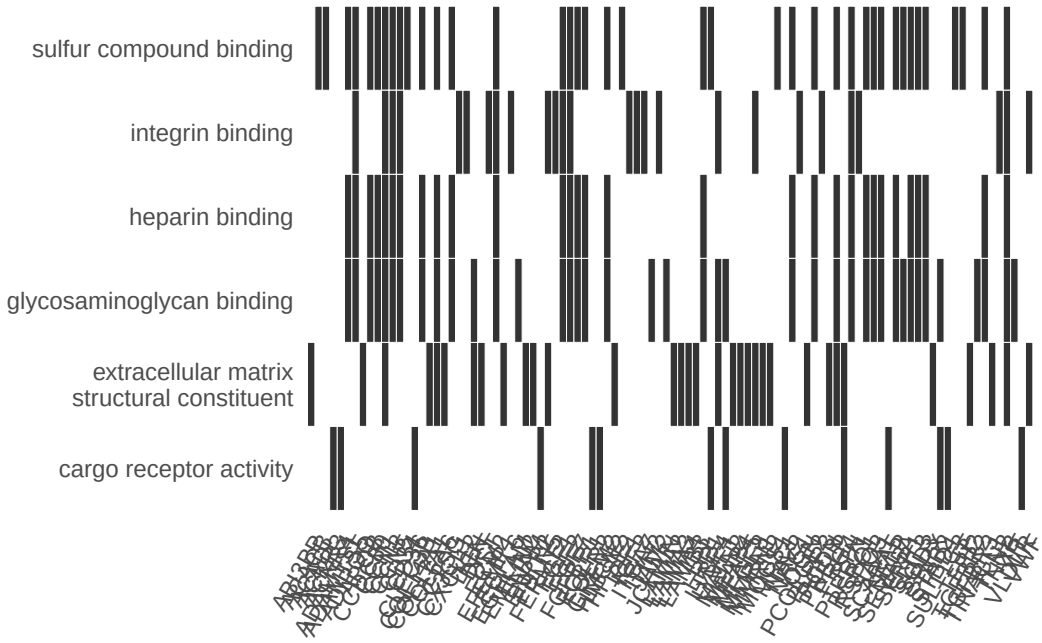

```r
dotplot(down_ego_MF, showCategory = 10)
```



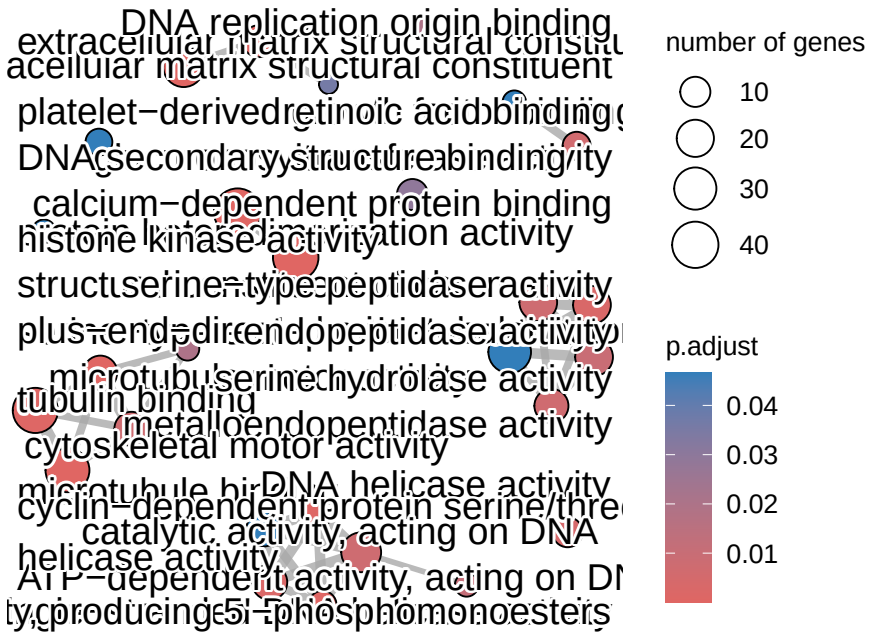

```
heatplot(up_ego_MF, showCategory = 6)
```



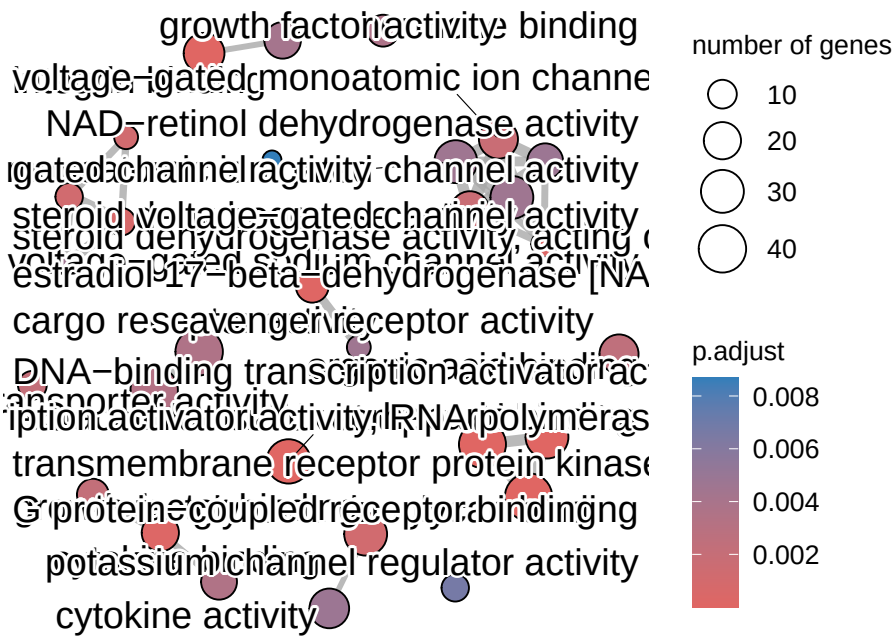

```
heatplot(down_ego_MF, showCategory = 6)
```




```
x2 <- pairwise_termsim(up_ego_MF)
emapplot(x2)
```




```
x2 <- pairwise_termsim(down_ego_MF)
emapplot(x2)
```



This analysis helps in understanding the functional roles of differentially expressed genes (`DEGs`) by identifying enriched Gene Ontology terms related to their molecular functions.

KEGG

Pathway enrichment analysis using KEGG is performed for both up-regulated genes and down-regulated. The most enriched pathways are displayed.

```
up_eWP <- enrichKEGG(gene = up_DEGs$entrezgene_id,
                     organism = 'human',
                     pvalueCutoff = 0.05,
                     qvalueCutoff = 0.1)
```

```
Reading KEGG annotation online: "https://rest.kegg.jp/link/hsa/pathway"...

Reading KEGG annotation online: "https://rest.kegg.jp/list/pathway/hsa"...
```

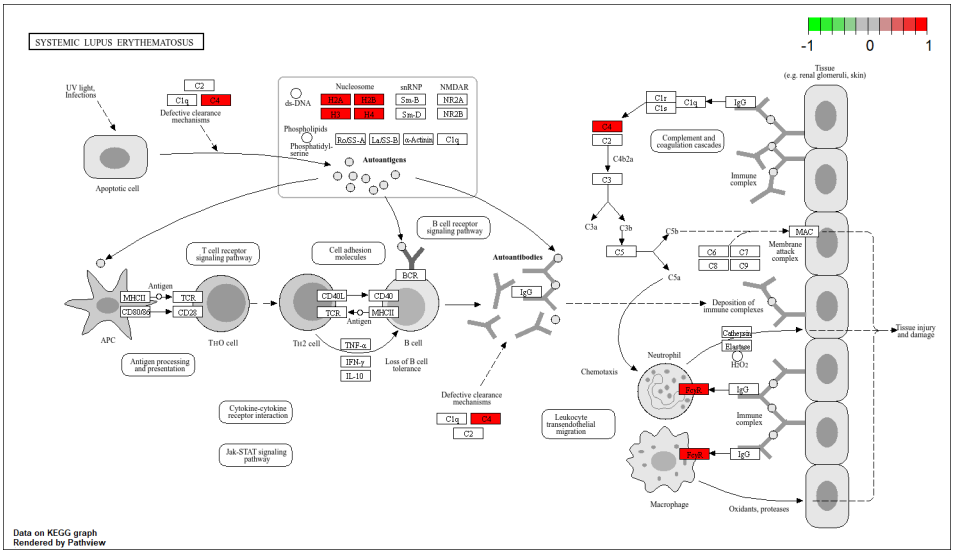
```
#head(up_eWP, n=10)

down_eWP <- enrichKEGG(gene = down_DEGs$entrezgene_id,
                     organism = 'human',
                     pvalueCutoff = 0.05,
                     qvalueCutoff = 0.1)

#head(down_eWP, n=10)
```

Task 5

Here, the visualization of one of the most enriched pathways for the up-regulated gene list is presented. In this case, the `hsa05322` pathway was utilized, representing Human Diseases.

```
library(pathview)
```

```
logFC <- up_DEGs$logFC
names(logFC) <- up_DEGs$entrezgene_id
pathview(gene.data = logFC,
         pathway.id = 'hsa05322',
         species = 'human')
```



The image shows the molecular pathway involved in Systemic Lupus Erythematosus (SLE), an autoimmune disease triggered by the immune system's recognition of apoptotic cells. This leads to the production of autoantibodies by B cells and activation of the complement system, resulting in tissue damage.

Task 6

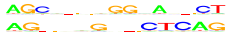
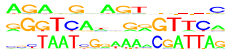
Identification of Transcription Factors (TFs)

This task aimed to identify transcription factors (TFs) with enriched scores in the promoters of all up-regulated genes. Specifically, the analysis focused on a window of 500 nucleotides upstream of each gene was used. So, analyzing the promoter sequences, several motifs, that are significantly enriched, were identified. Then, the results were plotted as sequence logos and ranked according to their enrichment scores.

```
library(TxDb.Hsapiens.UCSC.hg38.knownGene)
library(BSgenome.Hsapiens.UCSC.hg38)
library(biomaRt)
library(PWMEnrich)
library(PWMEnrich.Hsapiens.background)
library(MotifDb)
library(seqLogo)
```

```
genes_of_interest <- up_DEGs$external_gene_name.x
genes <- genes(TxDb.Hsapiens.UCSC.hg38.knownGene)
ensembl <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")
df <- getBM(attributes = c("external_gene_name",'entrezgene_id'),
            values=names(genes),
            filters ='entrezgene_id', mart = ensembl)

names(genes) <- df$external_gene_name[match(genes$gene_id,df$entrezgene_id)]

# Extract the genes names present both in genes and genes of interest
gene_names <- genes_of_interest[genes_of_interest %in% names(genes)]

x <- promoters(genes, upstream = 500, downstream = 0)[gene_names]

seq <- getSeq(BSgenome.Hsapiens.UCSC.hg38,x)
```

```
# Calculating motif enrichment scores
data(PWMLogn.hg19.MotifDb.Hsap)
res = motifEnrichment(seq,
                      PWMLogn.hg19.MotifDb.Hsap,
                      score = "affinity")
report = sequenceReport(res, 1)
#report
plot(report[report$p.value < 0.025], fontsize=7, id.fontsize=6)
```

Rank	Target	PWM	Motif ID	Raw score	P−value
1	UW.Motif.0579		UW.Motif.0579	44.9	0.000377
2	UW.Motif.0323		UW.Motif.0323	63.8	0.000485
3	UW.Motif.0275		UW.Motif.0275	42.1	0.000654
4	UW.Motif.0278		UW.Motif.0278	47.1	0.00152
5	UW.Motif.0568		UW.Motif.0568	44.3	0.0021
6	NFE2L2		NFE2L2	40.2	0.00243
7	UW.Motif.0003		UW.Motif.0003	47.6	0.00293
8	UW.Motif.0079		UW.Motif.0079	39.5	0.00333
9	UW.Motif.0094		UW.Motif.0094	49.9	0.00367
10	UW.Motif.0344		UW.Motif.0344	26.2	0.00376
11	UW.Motif.0643		UW.Motif.0643	20.6	0.00395
12	UW.Motif.0634		UW.Motif.0634	38.4	0.00441
13	BARX1		Hsapiens−jolma2013−BARX1	62.8	0.00484
14	UW.Motif.0249		UW.Motif.0249	20.5	0.0052
15	UW.Motif.0107		UW.Motif.0107	25.6	0.00571
16	UW.Motif.0653		UW.Motif.0653	22.4	0.00676
17	UW.Motif.0109		UW.Motif.0109	27.3	0.00678
18	UW.Motif.0317		UW.Motif.0317	17.3	0.00684
19	TIA1		TIA1	8.99	0.00872
20	IRF5		Hsapiens−jolma2013−IRF5−2	5	0.00947
21	UW.Motif.0155		UW.Motif.0155	25.8	0.0137
22	UW.Motif.0453		UW.Motif.0453	16.8	0.0143
23	ZNF435		Hsapiens−jolma2013−ZNF435	0.054	0.0158
24	MSX2		Hsapiens−jolma2013−MSX2	6.56	0.0163
25.5	HINFP		HINFP	2.18	0.0167
25.5	MIZF		MIZF	2.18	0.0167
27	UW.Motif.0199		UW.Motif.0199	26.1	0.017
28	VDR		Hsapiens−jolma2013−VDR	0.206	0.0171
29	RAX		RAX	5.23	0.0175
30	SMAD3		Hsapiens−jolma2013−SMAD3	0.793	0.0194
31	UW.Motif.0349		UW.Motif.0349	24.8	0.0197
32	RXRA::VDR		RXRA::VDR	2.65	0.0204
33	VENTX		Hsapiens−jolma2013−VENTX−2	0.00132	0.0206
34	CEBPA		M3031_1.02	5.47	0.022
35	MSX1		Hsapiens−jolma2013−MSX1	8.37	0.0226
36	CEBPA		M6169_1.02	4.19	0.0228
37	UW.Motif.0001		UW.Motif.0001	6.1	0.0236
38	AP1		AP1	4.63	0.0247

Task 7

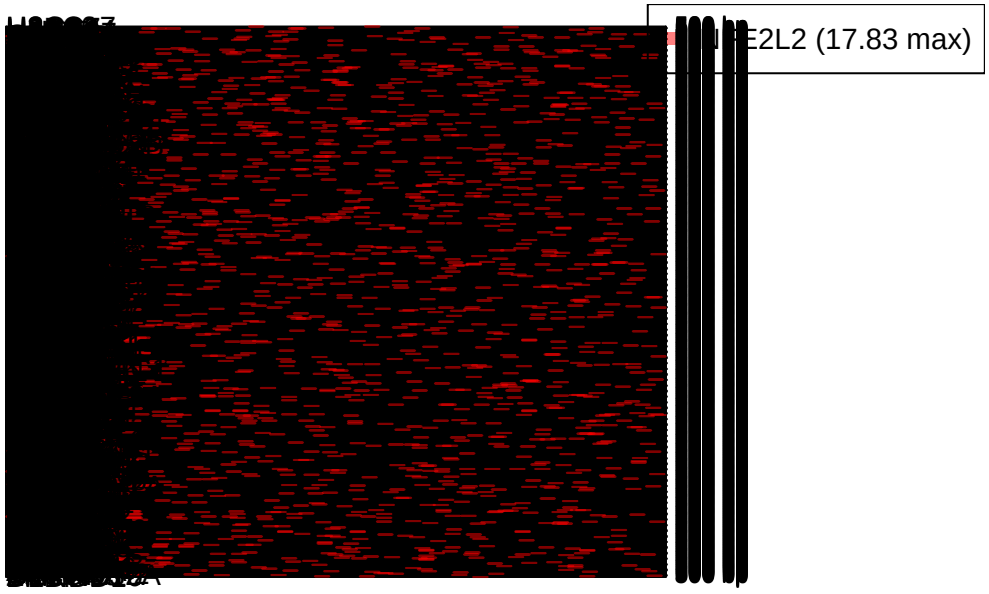
One of the main enriched TFs, NFE2L2, was selected. Empirical score distributions were calculated for all Position Weight Matrices (PWMs) found in MotifDB for the selected transcription factor. The cutoff of the distribution (log2) was determined to be 99.75 % for all.

(Some TFs with the term 'UW.M' caused problems, so the first TF without this term was selected).

```
# Empirical distribution
mdb.human = subset(MotifDb, organism=='Hsapiens' & geneSymbol == "NFE2L2")
PWM = toPWM(as.list(mdb.human))
names(PWM) <-  sapply(names(PWM),function(x) strsplit(x,"-")[[1]][3])
```

```
ecdf = motifEcdf(PWM,organism = "hg19",quick=TRUE)
threshold = log2(quantile(ecdf$"NFE2L2", 0.9975))
threshold
scores = motifScores(seq,PWM,raw.score=TRUE)
```

```
plotMotifScores(scores,sel.motifs="NFE2L2",cols=c("red","green","blue"), cutoff=threshold)
```



Task 8

Among the up-regulated genes, regions in their promoters with binding scores above the thresholds calculated for any of the previously selected PWMs were checked. By iterating through the sequences, the names of the genes with scores above the previously calculated threshold were stored.

```
genes_with_high_scores <- list()

for (i in 1:length(seq)) {
  scores <- motifScores(seq[i], PWM, raw.score = TRUE, cutoff = threshold)

  scores <- unlist(scores)
  if (any(scores >= threshold)) {
    genes_with_high_scores[[length(genes_with_high_scores) + 1]] <- gene_names[i]
  }
}
#print(genes_with_high_scores)
```

Task 9

The STRING database

The STRING database was used to find PPI interactions between differentially expressed genes and to export the network in TSV format.

```
# create a file with all the genes name (up-regulated)
write.table(unique(up_DEGs$external_gene_name.x),sep = '\t', file = 'UPDEGs.txt',row.names = F, c
# upload the file on STRING
```

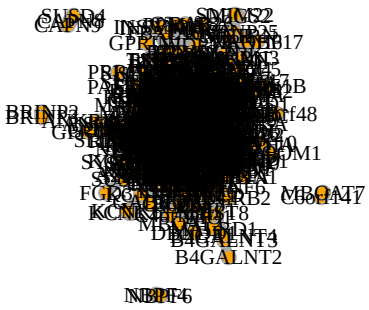
Task 10

The network was imported into R, and the largest connected component was identified using the `igraph` package. Initially, it was necessary to filter the interactions to keep only those occurring between the identified nodes. Then, connections with the Ensembl database were established to retrieve gene annotations. Finally, the graph was visualised using the plot.

```
library(igraph)
library(biomaRt)
```

```
links <- read.delim("C:/Users/virgi/Desktop/string_interactions NON.tsv")

filter_nodes <- unique(c(links$node2, links$X.node1))
# Filter the links
links <- links[links$X.node1 %in% filter_nodes & links$node2 %in% filter_nodes,]

ensembl <- useMart(biomart="ensembl", dataset="hsapiens_gene_ensembl")
nodes <- getBM(attributes=c("external_gene_name","ensembl_gene_id","description","gene_biotype","
                    filters="external_gene_name",
                    values=filter_nodes,
                    mart=ensembl)
nodes = unique(nodes[,c(1,3:6)])
# Create the graph with only the edges
```

```r
net <- graph_from_data_frame(d = links, directed = FALSE)

#class(net)

plot(net,
     edge.width=5,
     vertex.color="orange",
     vertex.size=10,
     vertex.frame.color="darkgray",
     vertex.label.color="black",
     vertex.label.cex=0.7,
     edge.curved=0.1)
```



The initial plot was chaotic and not informative. To obtain a clearer plot, the settings on the STRING website were adjusted to remove nodes without connections and set the confidence level to “highest”. The following plot was obtained.


SLX1B
FOXP3
TNFRSF1B