

# Vignette Title

true            true            true

Loading required libraries

```
library(fgsea)
library(org.Hs.eg.db)
library(biomaRt)
library(clusterProfiler)
library(enrichplot)
library(ggnewscale)
library(DOSE)
library(pathview)
library(tidyverse)
library(dplyr)
library(edgeR)
library(limma)
library(GenomicFeatures)
library(ggplot2)
library(stringr)
library(MotifDb)
library(seqLogo)
library(PWMErich)
library(PWMErich.Hsapiens.background)
library(igraph)
library(TxDb.Hsapiens.UCSC.hg38.knownGene)
library(BSgenome.Hsapiens.UCSC.hg38)
library(Biostrings)
library(TFBSTools)
library(tidyr)
```

## TASK 1

Loading the required data.

The `Liver_hepatocellular_carcinoma.RData` file contains 3 dataframes: - `raw_count_df` with the raw RNA-Seq counts - `c_anno_df` with the samples' names and condition - `r_anno_df` with the ENSEMBL gene IDs, the length of the genes and the genes' symbols

```
load("./Liver_hepatocellular_carcinoma.RData")
# To correctly load the file be sure that the .Rmd and .RData files are in the
# same folder
```

## TASK 2

In this first phase we filter and update gene annotations and raw counts data to focus on protein-coding genes using the BioMart database, facilitating subsequent analyses that require such specific gene information.

```
# Select the BioMart database and the dataset to use
ensembl <- useMart(biomart = "ensembl", dataset = "hsapiens_gene_ensembl")

# Retrieve all the attributes needed for the project
query <- getBM(attributes = c("ensembl_gene_id", "external_gene_name", "gene_biotype",
  "start_position", "end_position", "strand", "description", "version"), filters = "ensembl_gene_id",
  values = list(c(r_anno_df$ensembl_gene_id)), mart = ensembl)

head(query)
```

```
##   ensembl_gene_id external_gene_name  gene_biotype start_position end_position strand
## 1 ENSG00000056678          KIFC1         lncRNA      4827484      4827813        1
## 2 ENSG00000096150          RPS18 protein_coding    4466427      4470930        1
## 3 ENSG00000096155          BAG6 protein_coding    2894841      2908524       -1
## 4 ENSG00000096171          VARS1 protein_coding    3025292      3043727       -1
## 5 ENSG00000111971          LY6G5C protein_coding    2975968      2983321       -1
## 6 ENSG00000112459          OR14J1 protein_coding     572513      573478         1
##                                     description version
## 1               kinesin family member C1 [Source:HGNC Symbol;Acc:HGNC:6389]      11
## 2               ribosomal protein S18 [Source:HGNC Symbol;Acc:HGNC:10401]       9
## 3                   BAG cochaperone 6 [Source:HGNC Symbol;Acc:HGNC:13919]      15
## 4               valyl-tRNA synthetase 1 [Source:HGNC Symbol;Acc:HGNC:12651]     14
## 5      lymphocyte antigen 6 family member G5C [Source:HGNC Symbol;Acc:HGNC:13932]  14
## 6 olfactory receptor family 14 subfamily J member 1 [Source:HGNC Symbol;Acc:HGNC:13971]  5
```

```
# Select protein coding genes
query_pc <- query[which(query$gene_biotype == "protein_coding"), ]

# Extract only protein coding genes
gene_names <- intersect(r_anno_df$ensembl_gene_id, query_pc$ensembl_gene_id)

# Update r_anno_fin with the new genes
r_anno_df <- r_anno_df[, -1]
r_anno_fin <- r_anno_df[gene_names, ]

# Update raw_counts_fin with the new genes
raw_counts_fin <- raw_counts_df[gene_names, ]
raw_counts_fin <- raw_counts_fin + 1 # pseudocount for mathematical issues
```

## TASK 3

For this task we have to perform a differential expression analysis using edgeR package and select 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.

Firstly, we prepared the data. We filtered the genes based on the new thresholds: counts greater than 20 and an occurrences of 5 sample in both groups.

We created a DGEList object with filtered data and normalize it using TMM. The final DGEList object contains: a numeric matrix with the read counts, a dataframe with samples' condition, a dataframe with samples and a dataframe giving annotation information for each gene.

```
# Filter raw counts data retaining only genes with raw count > 20...
count_thr <- 20
# ... in at least 5 Cases or 5 Control samples
repl_thr <- 5

# Filter annotations and counts
filter_vec <- apply(raw_counts_fin, 1, function(y) max(by(y, c_anno_df$condition,
  function(x) sum(x >= count_thr))))

filter_counts_df <- raw_counts_fin[filter_vec >= repl_thr, ]
filter_anno_df <- r_anno_fin[rownames(filter_counts_df), ]

# Normalization of the selected data have via TMM (trimmed mean of M values)
edge_c <- DGEList(counts = filter_counts_df, group = c_anno_df$condition, samples = c_anno_df,
  genes = filter_anno_df)
edge_n <- calcNormFactors(edge_c, method = "TMM")
cpm_table <- as.data.frame(round(cpm(edge_n), 2))
```

Hence, we performed differential expression analysis with edgeR and classified genes in up- or down-regulated based on logCPM and logFC criteria.

```
# Differential expression analysis cut-off CRITERIA p-value = 0.01 and logFC
# ratio > 1.5 for up-regulated logFC ratio < -1.5 and logCPM > 1 for
# down-regulated

# 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
contrast <- makeContrasts("case-control", levels = design)

# Fit the model with generalized linear models
edge_t <- glmQLFTest(edge_f, contrast = contrast)
DEGs <- as.data.frame(topTags(edge_t, n = 20000, p.value = 0.01, sort.by = "logFC"))
DEGs$class <- "="

# Add labels for up (+) and down (-) regulations
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 = TRUE), ]
```

# GRAPHICAL REPRESENTATION OF DEG RESULTS

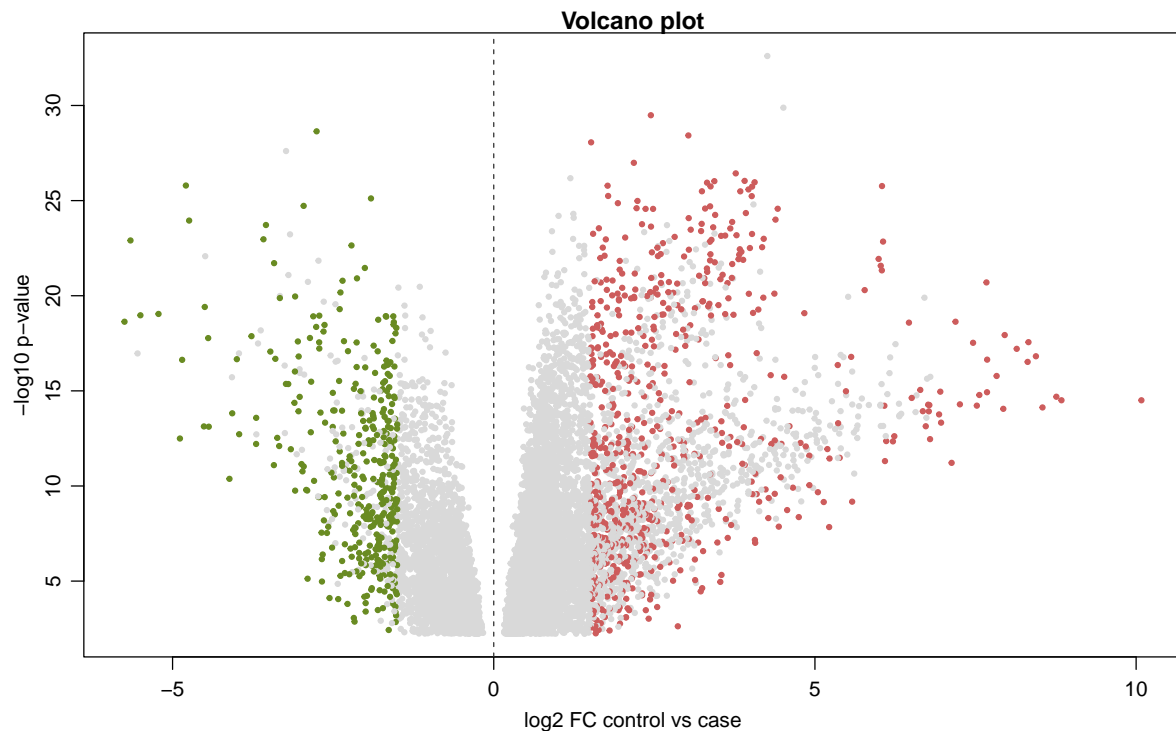
The results of DGE analysis are plotted through a volcano plot and an annotated heatmap.

## Volcano Plot

The following volcano plot represents a high number of up-regulated genes (red = up-regulated, green = down-regulated), with a  $\logFC > 1$ . Meanwhile, the number of down-regulated genes are visually lower.

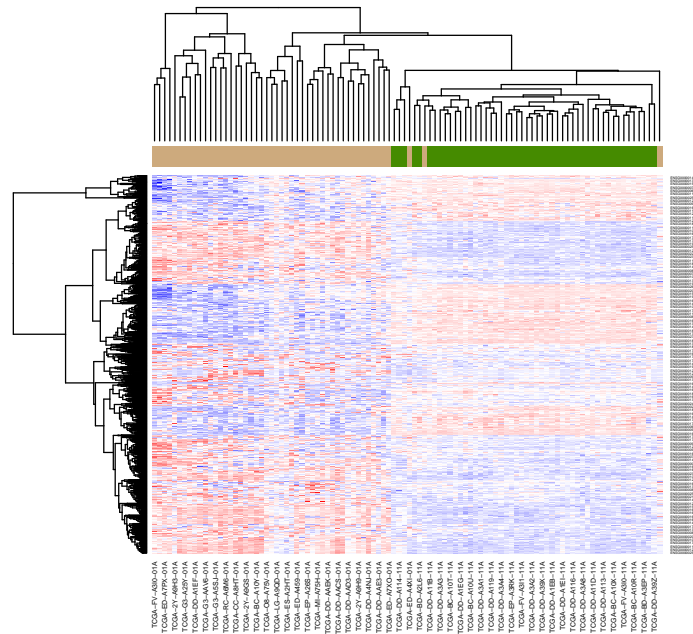
The volcano plot below visualizes the results of the differential expression analysis. The x-axis represents the log fold change ( $\logFC$ ) of gene expression between conditions, and the y-axis represents the p-value, indicating the statistical significance of the changes.

Genes with a high  $\logFC$  are considered to have significant changes in expression. Points on the right side of the plot represent up-regulated genes, while points on the left side represent down-regulated genes. The color coding highlights genes that are significantly up-regulated (red) and down-regulated (green) based on a defined threshold.



## Heatmap

The heatmap shows in the samples clustering a good stratification between control and test samples. Notably, there seem to be some differentially expressed genes (not colored in blue) in the test samples (yellow). Visually, a correlation can be found from some down-regulated genes and their sample type. Clearly some additional statistical test need to be performed to prove the correlation.



## TASK 4

In this task, we performed Gene Set Enrichment Analysis (GSEA) to identify biological processes that are significantly enriched in the up-regulated and down-regulated genes. This involved filtering the differentially expressed genes (DEGs) based on expression and significance criteria, converting gene IDs, and then performing Gene Ontology (GO) enrichment analysis.

We started by filtering the DEGs based on differential expression, class, and significance.

Hence, we performed Gene Ontology enrichment analysis with Biological Process (BP)

```
# Upregulated genes
up_DEGs <- DEGs %>%
  filter(class == "+")
down_DEGs <- DEGs %>%
  filter(class == "-")

up_ego_BP <- enrichGO(gene = up_DEGs$external_gene_name, OrgDb = org.Hs.eg.db, keyType = "SYMBOL",
  ont = "BP", pAdjustMethod = "BH", pvalueCutoff = 0.05, qvalueCutoff = 0.05)

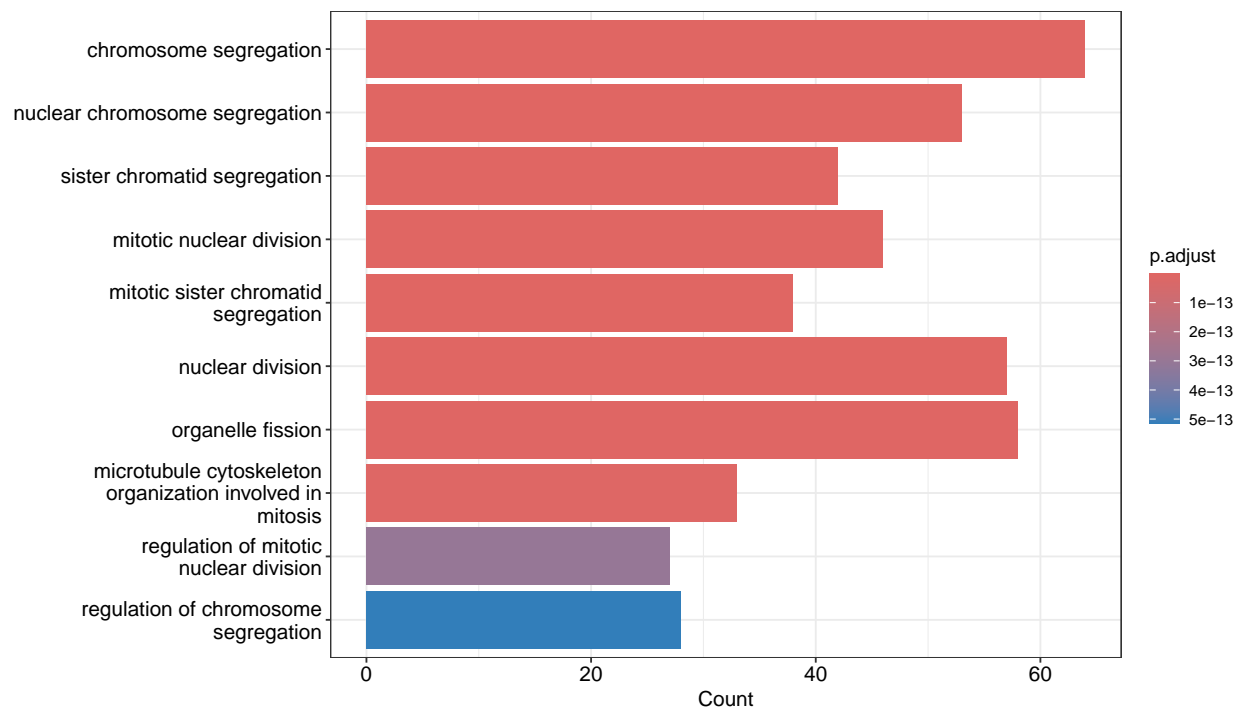
down_ego_BP <- enrichGO(gene = down_DEGs$external_gene_name, OrgDb = org.Hs.eg.db,
  keyType = "SYMBOL", ont = "BP", pAdjustMethod = "BH", pvalueCutoff = 0.05, qvalueCutoff = 0.05)
```

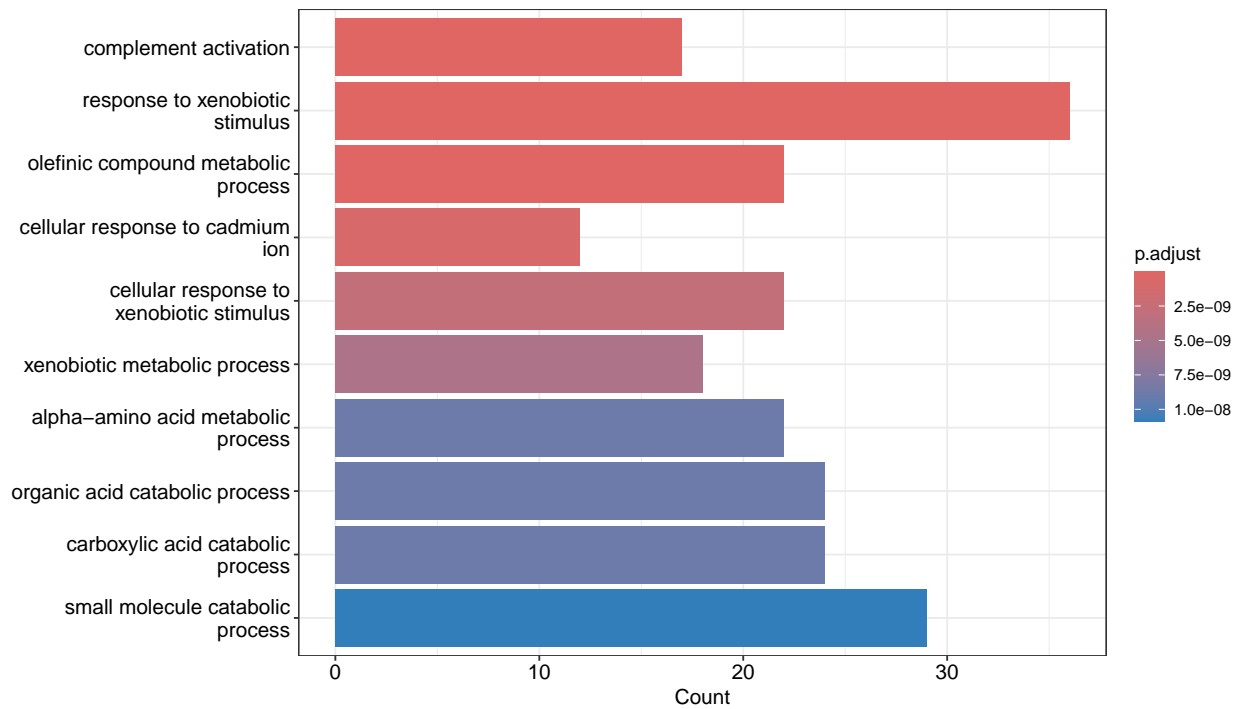
# GRAPHICAL REPRESENTATION OF GO

We visualize the results of the enrichment analysis using various plots to better understand the enriched biological processes.

## Barplot Visualization

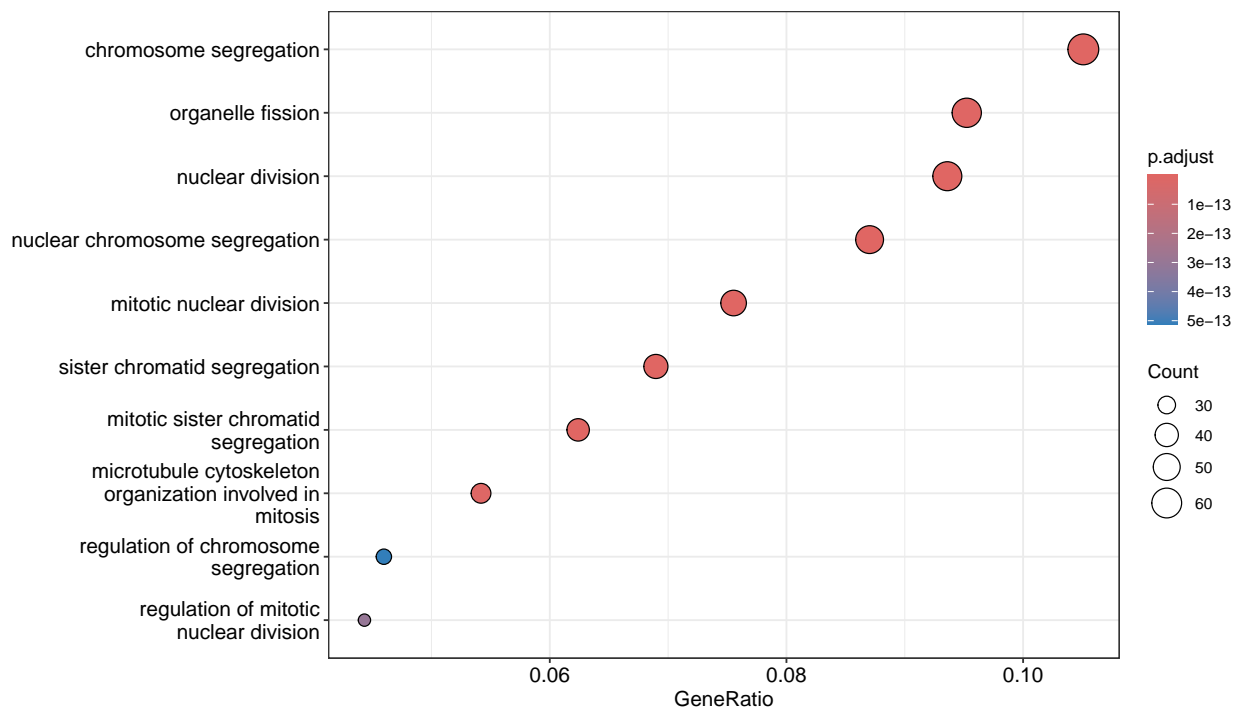
We use barplots to visualize the top 10 enriched terms.

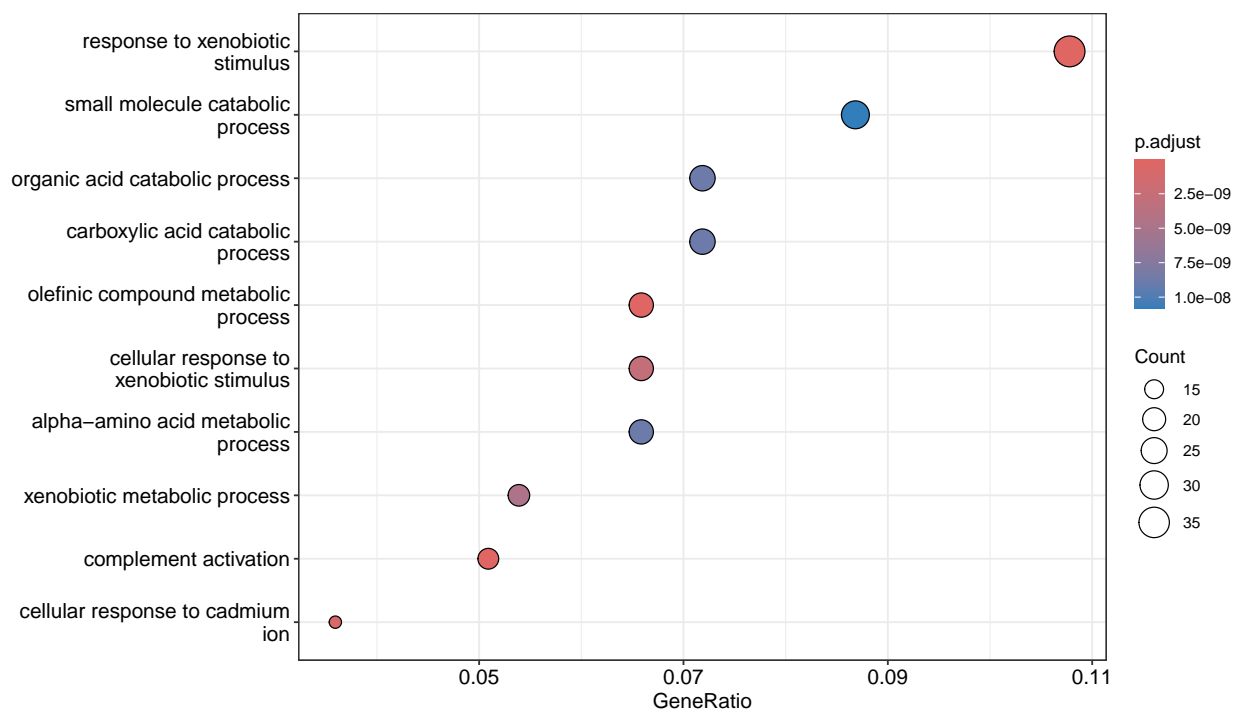




## Dotplot Visualization

We use dotplots to visualize the top 10 enriched terms.



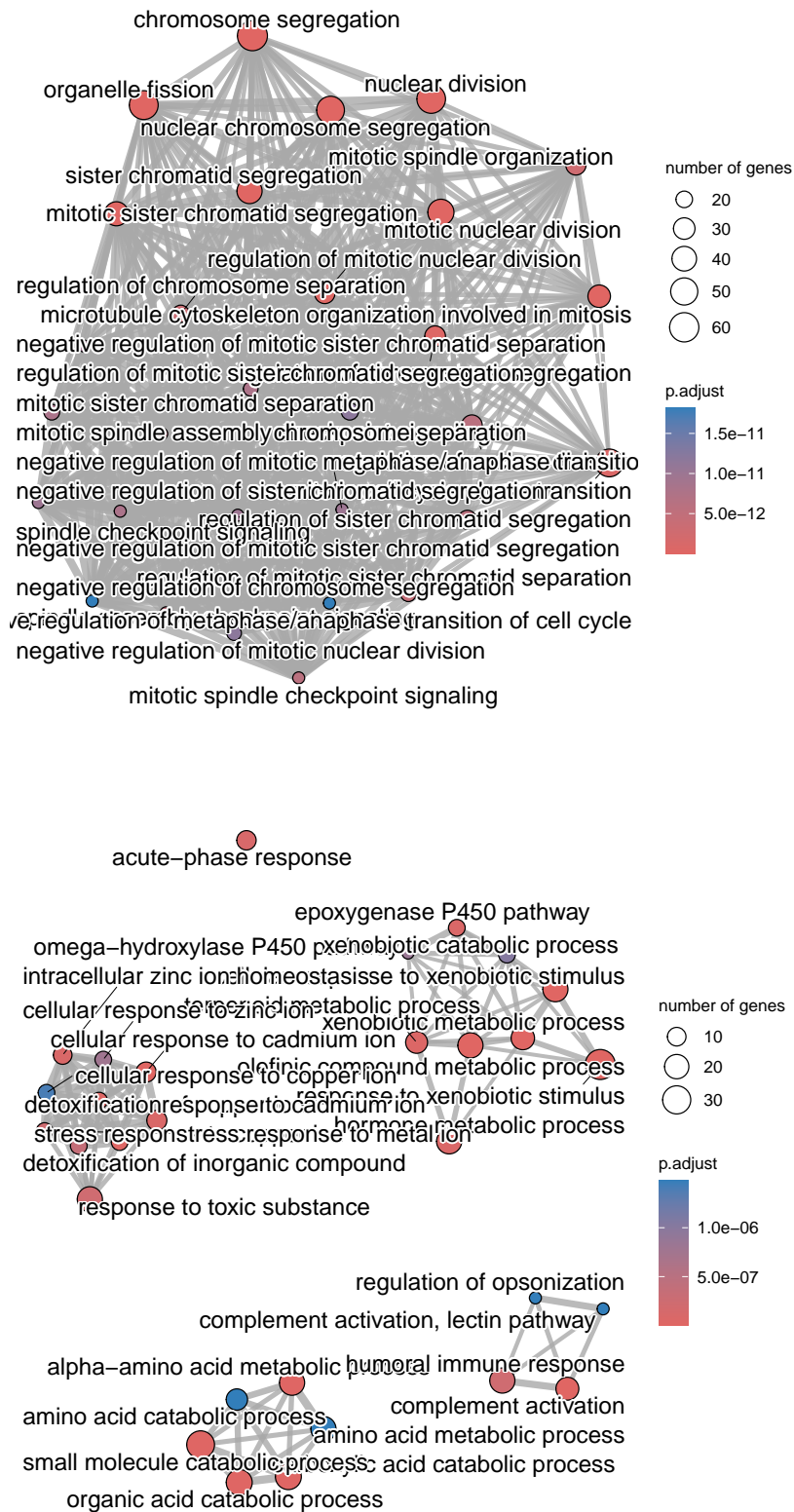


## Heatmap Visualization

We use heatmaps to visualize the top 6 enriched terms.



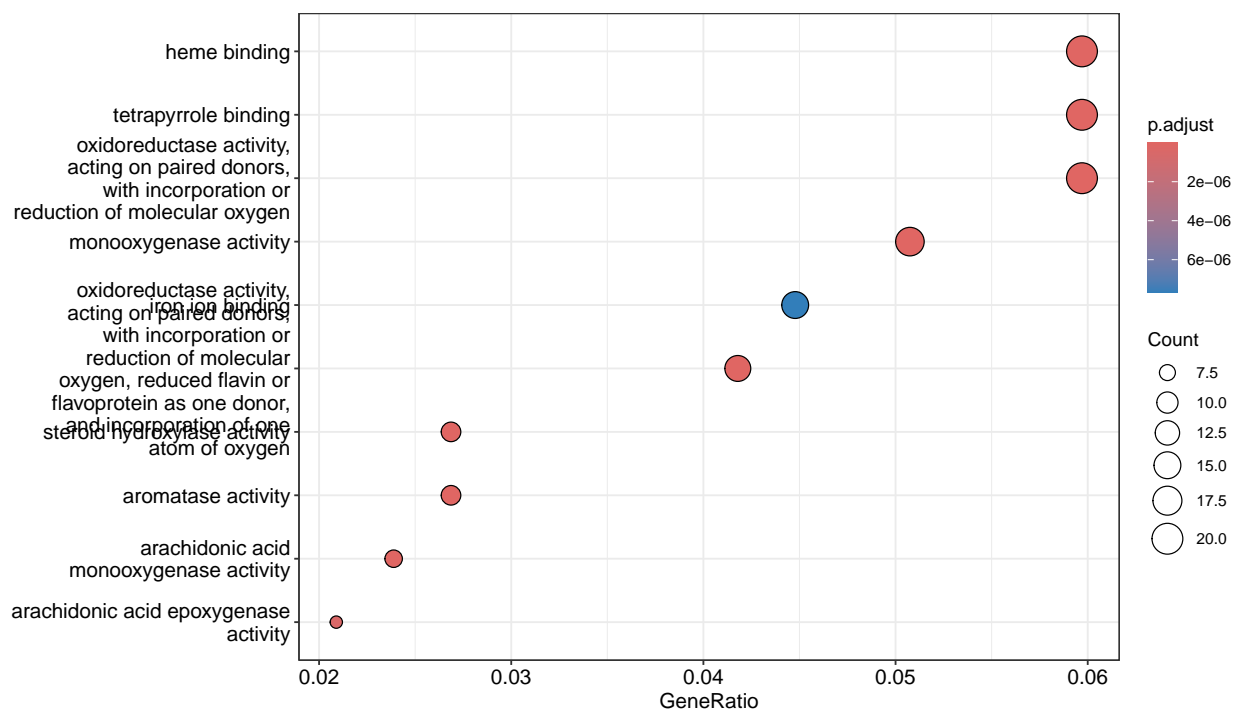
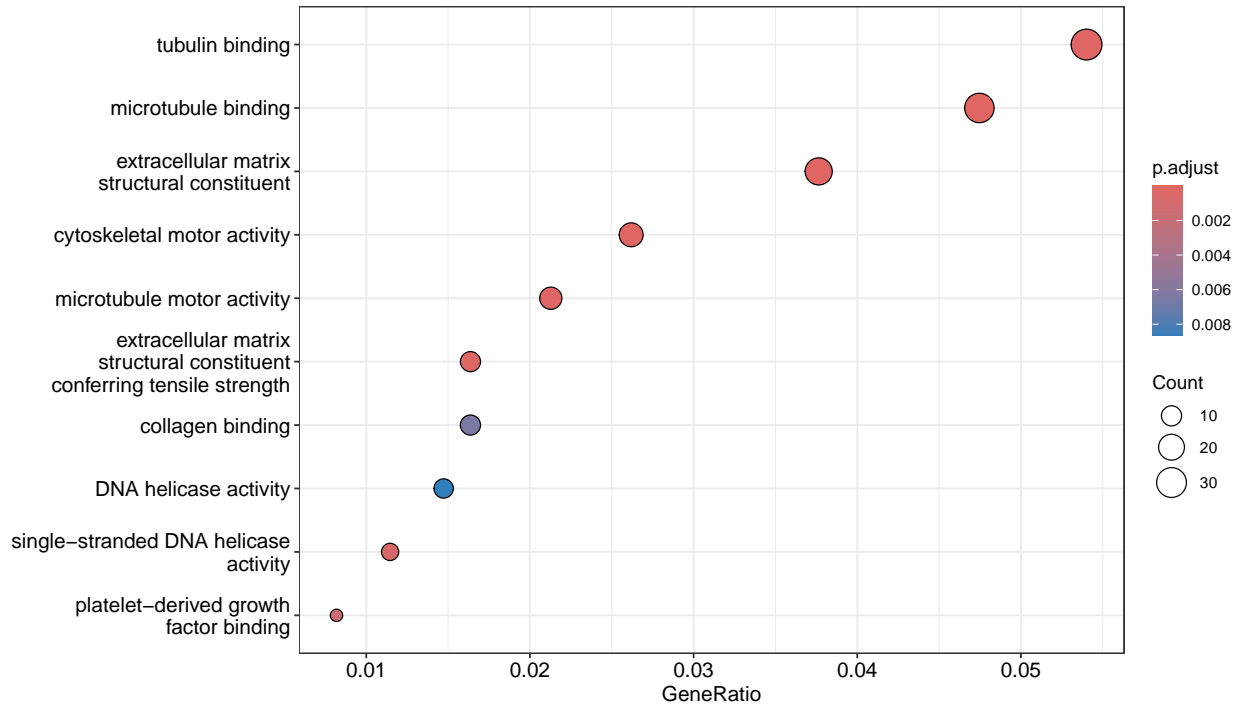




At this point, we repeated the enrichment analysis for the molecular function (MF). We want to identify molecular functions that are significantly enriched in the up-regulated and down-regulated genes.

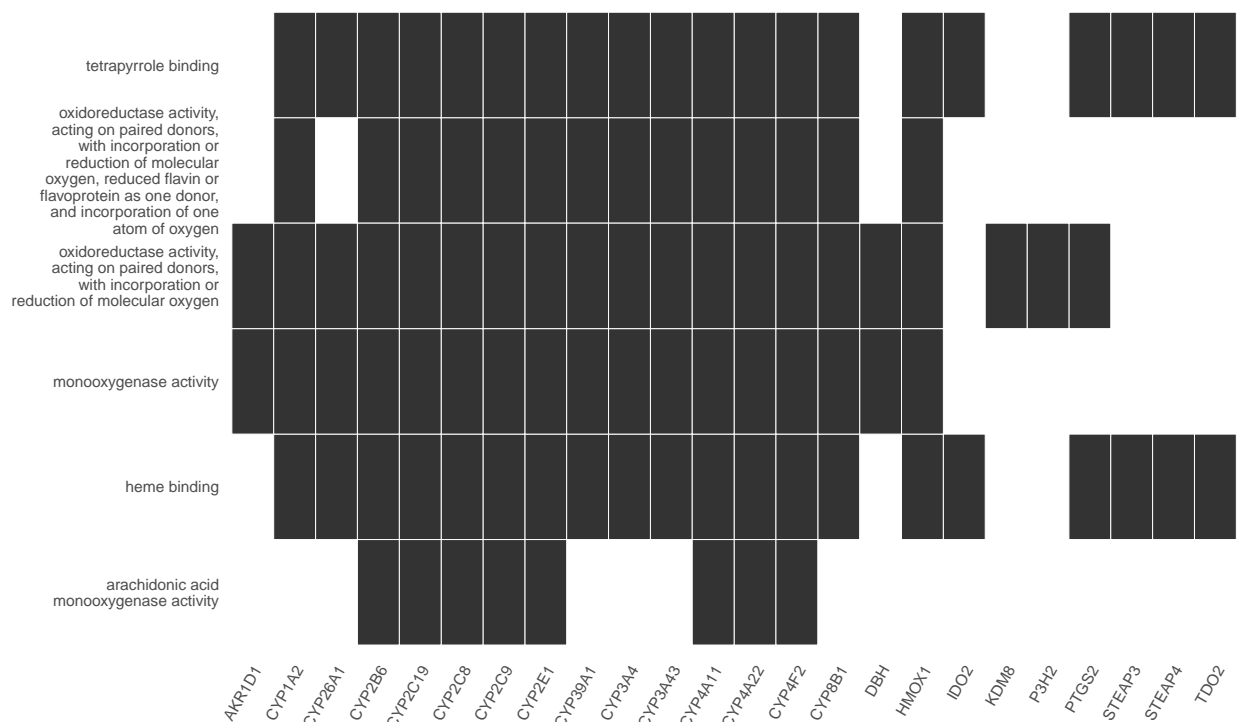
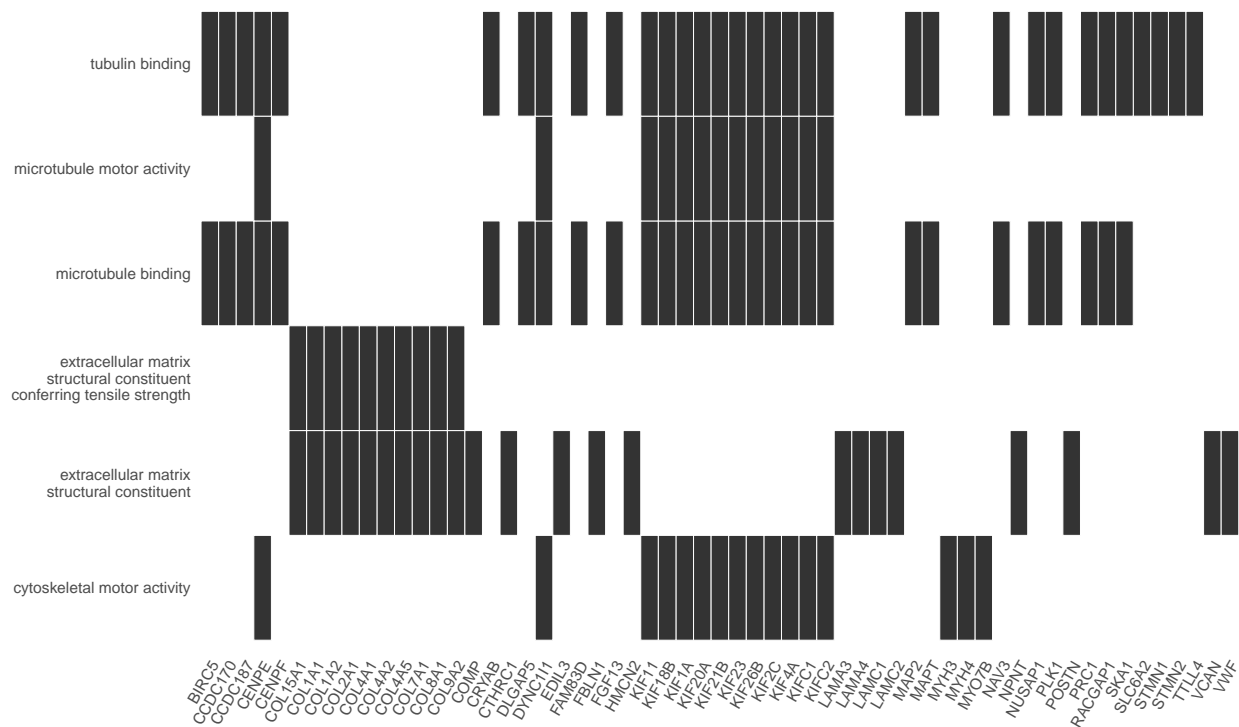
## Dotplot Visualization

We use dotplots to visualize the top 10 enriched terms.



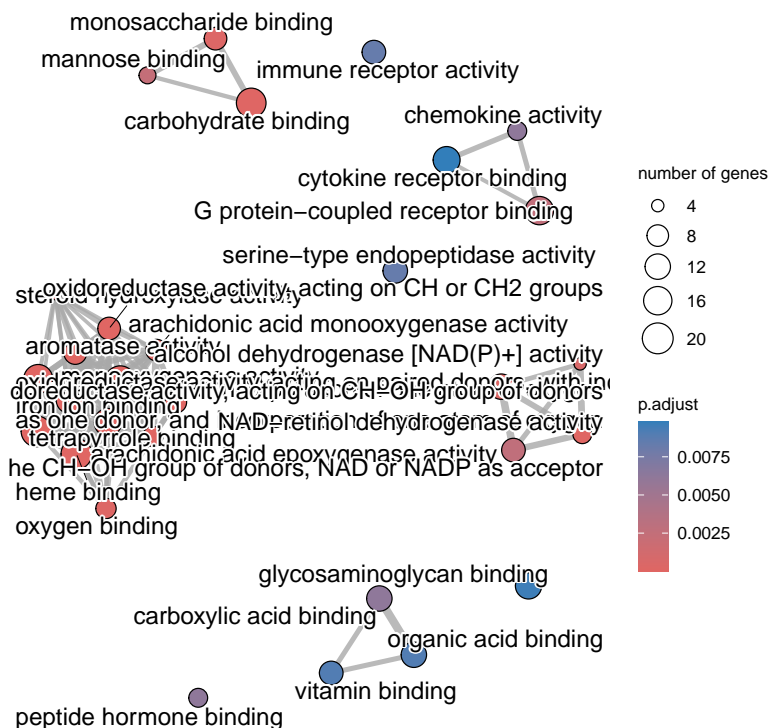
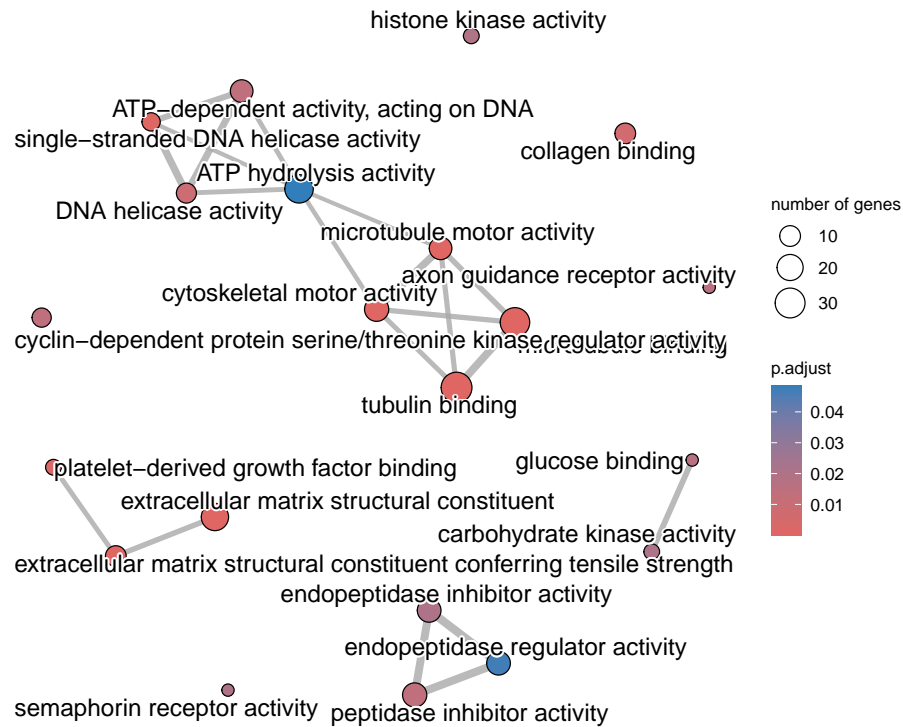
## Heatmap Visualization

We use heatmaps to visualize the top 6 enriched terms.



## Network Visualization

We use network plots to visualize the connections between the enriched terms.



## TASK 5: PATHWAY RETRIEVAL

In this task, we aimed to identify the biological pathways in which our up-regulated genes are enriched. Filtered the genes according to their differential expression, we performed KEGG pathway enrichment analysis using the `enrichKEGG` function to identify pathways enriched in the up-regulated genes. The function filters pathways based on a p-value cutoff of 0.05 and a q-value cutoff of 0.1.

For this project, we focused on different types of tumors and specifically analyze the pathway “hsa05202” which is associated with transcriptional misregulation in cancer. By focusing on the “hsa05202” pathway, we can gain insights into how transcriptional misregulation contributes to different types of tumors. The visualization helps in understanding the specific genes involved and their interactions within this critical biological process.

```
# Starting with a filtering of the genes depending on the up or lower
# expression
log_FC_pos <- up_DEGs$logFC

# Set the correct options for the pathviewer function and then check manually
# the image produced
up_eKEGG <- enrichKEGG(gene = up_DEGs$entrezgene_id, organism = "human", pvalueCutoff = 0.05,
                        qvalueCutoff = 0.1)

names(log_FC_pos) <- up_DEGs$entrezgene_id

# Visualize the pathway using the pathview function
pathview(gene.data = log_FC_pos, pathway.id = "hsa05202", species = "human")
```

## TASK 6:

In this task, we identified transcription factors (TFs) with enriched scores in the promoters of up- and down-regulated genes. This involved loading the genome sequence and gene annotations, retrieving gene names, generating promoter regions, extracting promoters for upregulated genes, and performing motif enrichment analysis.

We started by loading the necessary genome sequence and gene annotations. We retrieved the gene names corresponding to the Entrez gene IDs and generated promoter regions for the genes, specifying 500 bp upstream and 0 bp downstream. At this point, we extracted only those promoters for upregulated genes.

```
# Load the genome sequence and gene annotations
genome <- BSgenome.Hsapiens.UCSC.hg38
genes <- genes(TxDb.Hsapiens.UCSC.hg38.knownGene)
ensembl <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")

# Retrieve Gene Names
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)]

# Generate Promoter Regions
x <- promoters(genes, upstream = 500, downstream = 0)

# Extract the promoters for upregulated genes
```

```

up_tf_names <- intersect(up_DEGs$external_gene_name, x@ranges@NAMES)
up_tf <- subset(up_DEGs, external_gene_name %in% up_tf_names)
up_promoters <- x[names(x) %in% up_tf_names]
up_promoter_seqs <- getSeq(genome, up_promoters)

```

We performed motif enrichment analysis on the upregulated promoters, generating a report of the enriched transcription factors with a p-value below 0.01. In order to run easily the code, we computing the enrichment only once and saved the result in a .RData file, which was subsequently loaded.

```

# Perform motif enrichment analysis on the upregulated promoters

# data(PWMLogn.hg19.MotifDb.Hsap) up_enr <- motifEnrichment(up_promoter_seqs,
# PWMLogn.hg19.MotifDb.Hsap, score = 'affinity') save(up_enr, file =
# 'enriched_transcription.RData')
load("enriched_transcription.RData")


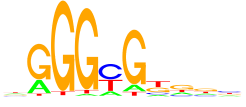


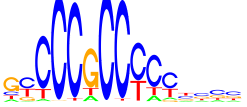
# Generate a report of the enriched transcription factors filter to keep only
# those with a p-value below 0.025
report_up <- groupReport(up_enr)

```

## GRAPHICAL REPRESENTATION OF ENRICHMENT ANALYSIS

Here the 5 most representative motifs for transcription factors (TFs) in the promoters of up-regulated genes. In particular the graphs shows in order: - The rank of the motif compared to the other motifs found - The target TF - The Positional Weight Matrix for the motif - The Motif ID - The raw enrichment score - The p-value

This plot will be exploited to solve the next tasks.

Rank	Target	PWM	Motif ID	Raw score	P-value	In top motifs
1	PGAM2		PGAM2	7.99	3.73e-56	19 %
2	TFAP4		M2944_1.02	3.63	4.58e-55	18 %
3	CEBPB		M4556_1.02	2.43	4.66e-55	19 %
4	ZMAT2		ZMAT2	2.26	6.67e-53	17 %
5	SP2		SP2	121	6.69e-53	19 %

## TASK 7

In this task, we selected one of the top enriched transcription factors (TFs) for the up-regulated genes. We then compute the empirical distributions of scores for all position weight matrices (PWMs) found in MotifDB for the selected TF and determine the distribution (log2) threshold cutoff at 99.75%.

For this example, we focused on the transcription factor PGAM2, which is among the top enriched TFs for the up-regulated genes.

This analysis helps in identifying the transcription factor binding sites that are most likely to be biologically significant based on their enrichment scores in the promoters of up-regulated genes. By determining a stringent threshold cutoff, we ensure that only the most relevant motifs are considered for further analysis.

```
# UP-REGULATED genes; selecting PGAM2
library(MotifDb)

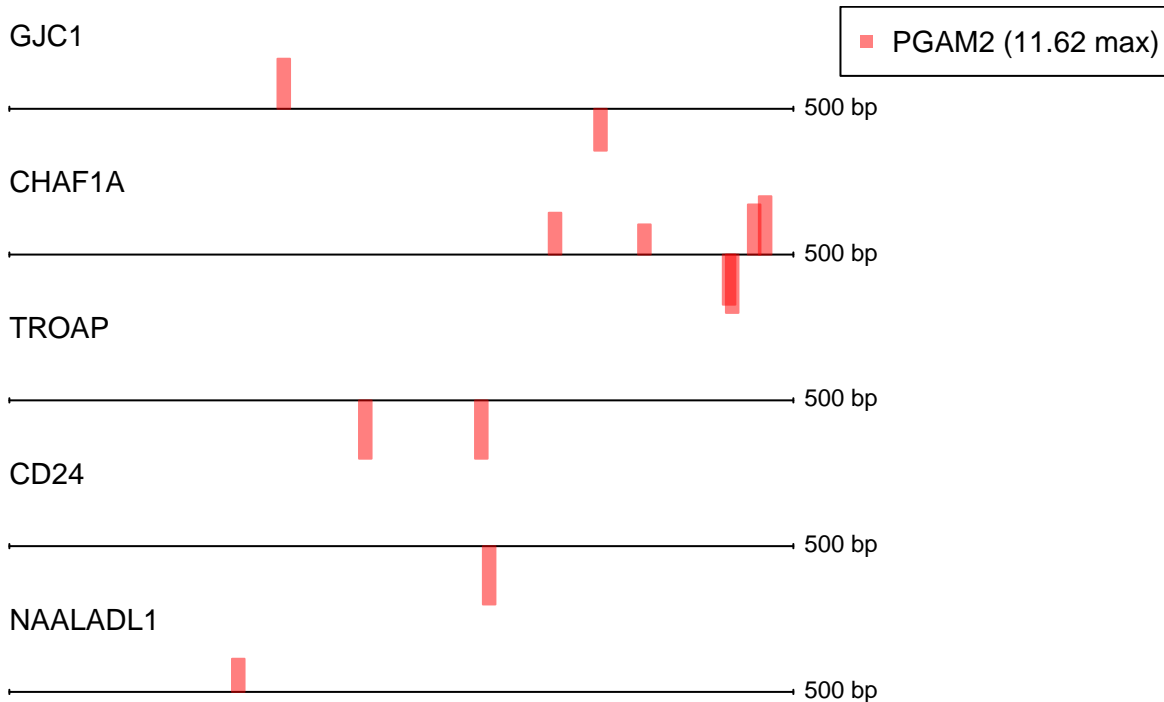
mdb_human_up <- subset(MotifDb, organism == "Hsapiens" & geneSymbol == "PGAM2")
PWM_up <- toPWM(as.list(mdb_human_up))

names_up <- sapply(names(PWM_up), function(x) {
  parts <- str_split(x, "-")[1]
  return(parts[3])
})
names(PWM_up) <- names_up

# Compute the empirical distribution score for our set of motif
ecdf_up <- motifEcdf(PWM_up, organism = "hg19", quick = TRUE)
threshold_up <- log2(quantile(ecdf_up$PGAM2, 0.9975))
scores_up <- motifScores(up_promoter_seqs, PWM_up, raw.scores = T)
```



```
# Plot the first 5 scores
plotMotifScores(scores_up[1:5], sel.motifs = "PGAM2", cols = c("red", "green", "blue"),
  cutoff = threshold_up)
```



## TASK 8

Analogous to the previous plot, this graph shows the motif scores for every promoter sequence retrieved from Ensembl. We iterate through each promoter sequence to identify matches and plot the best scoring motifs. We iterated through each promoter sequence, computed the motif scores, and identified matches based on the threshold. We then visualized the best scoring motifs.

```
# pattern matching
matches <- c()
nomatches <- c()
best <- 0
best_scores <- 0

for (n in 1:length(up_promoter_seqs)) {
  scores = motifScores(up_promoter_seqs[n], PWM_up, cutoff = threshold_up)
  if (any(scores) > 0) {
    matches <- append(matches, n)
    if (sum(scores) > best) {
      best <- n
    }
  }
}
```

```

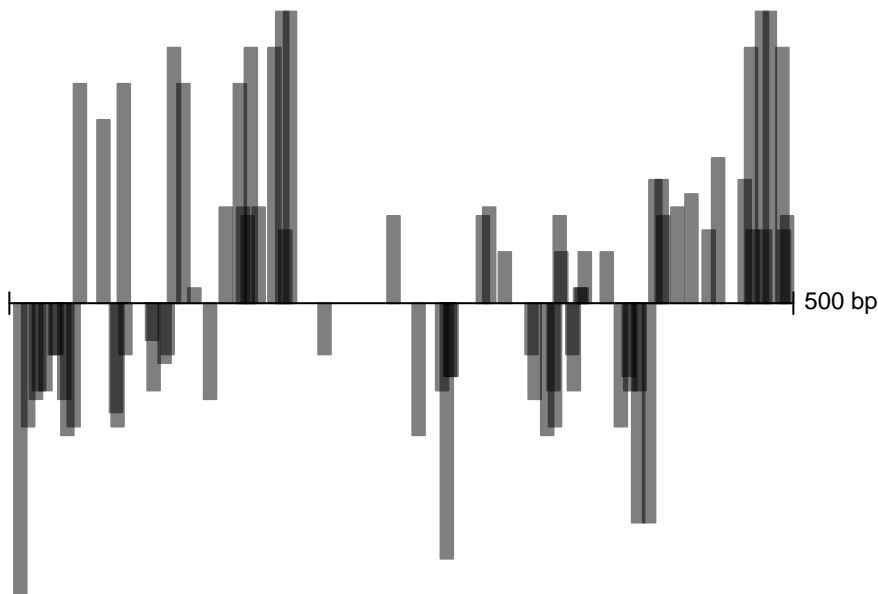
        best_scores <- motifScores(up_promoter_seqs[n], PWM_up, cutoff = threshold_up,
                                   raw.scores = TRUE)
      }
    } else {
      nomatches <- append(nomatches, n)
    }
  }
}

plotMotifScores(best_scores, legend.cex = 0.5)

```

IGF2BP2

■ PGAM2 (13.26 max)



## TASK 9

In this task, we used the STRING database to find protein-protein interactions (PPI) among differentially expressed genes (DEGs) and exported the network in TSV format. We identified the up- and down-regulated genes that are differentially expressed with a p-value lower of equal than 0.05.

```

up_reg <- up_DEGs["external_gene_name"]
up_reg <- as.list(up_reg)

down_reg <- down_DEGs["external_gene_name"]
down_reg <- as.list(down_reg)

# Identify which are the genes across the dataframe which may be significant

```

```
diff_express <- subset(DEGs, class == "+" & PValue <= 0.05 | class == "-" & PValue <=
0.05)

# Produce variables for their storing and manipulation
final_set <- diff_express["external_gene_name"]

final_list <- as.list(final_set)

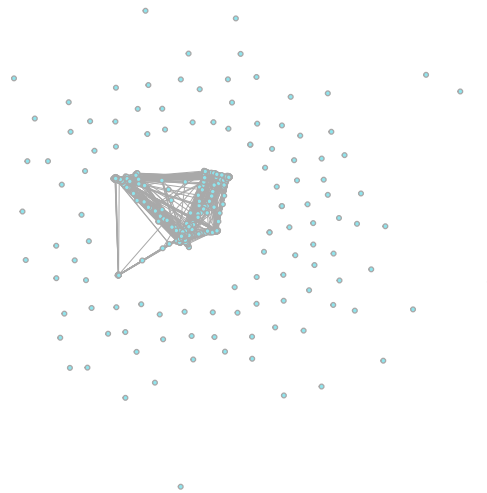
# Produce a file in which the list of all the significant genes are present
reg <- write.table(final_list, file = "final_list.txt", sep = " ", row.names = F,
col.names = F)
```

## TASK 10

After obtaining the file, the procedure is: - Go to STRING db. - Select the “Multiple proteins” option. - Load the file and check the network.

After exporting the TSV table of the coordinates and links from STRING and importing them into RStudio, we can visualize the full network and then extract and visualize the most connected nodes.

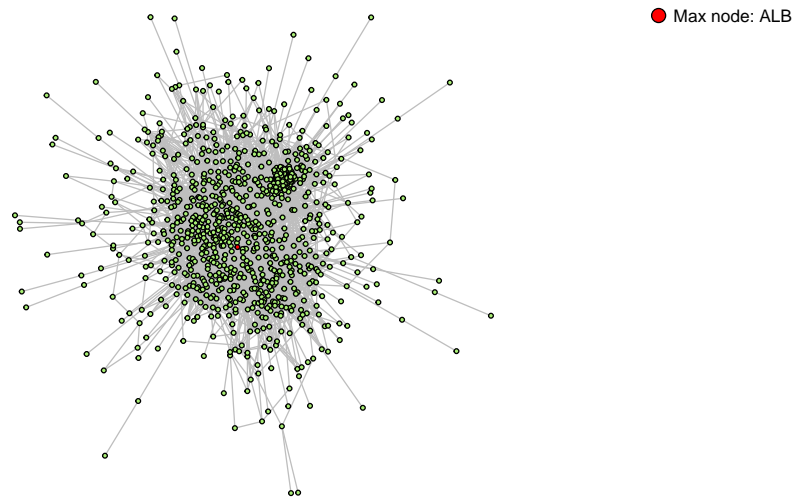
We load the data exported from STRING and visualize the full PPI network.



Here a clustering has been performed in order to highlight the largest connected component LCC.

The node most connected among the network is representing the ‘ALB’ gene. ALB is known for being related to the production of the most abundant protein in human blood: “Albumin”, acting as a carrier for a wide range of endogenous molecule and helps in the regulation of plasma osmotic pressure. Characteristic of the HCC.

### Largest Connected Component



This analysis helps in visualizing the PPI network and identifying key nodes that may play significant roles in the biological processes of interest, such as the production of Albumin in liver hepatocellular carcinoma.