Bioinformatics Resources Project

Leonardo Golinelli

2024-08-20

Dataset: Kidney renal papillary cell carcinoma
Course: Bioinformatics Resources
Professor: Dr. Alessandro Romanel

Task 1

Load the RData file. The following three data-frames are available: a raw_counts_df = contains the raw RNA-seq counts; b c_anno_df = contains sample names and conditions (case or control); c r_ anno_df = contains the ENSEMBL genes ids, the length of the genes and the genes symbols.

```
load("Kidney_renal_papillary_cell_carcinoma.RData")
ls()
```

```
## [1] "c_anno_df"     "r_anno_df"     "raw_counts_df"
```

Task 2

Update raw_count_df and r_anno_df extracting only protein coding genes: a Use biomaRt package to retrieve the needed information; b Next tasks should use the new data-frames you have created.

```
suppressMessages(library(biomaRt)) # to access gene annotation
suppressMessages(library(glue)) #for more python-like printing

#load db of human genes from ensembl
ensembl <- useMart(biomart="ensembl",dataset="hsapiens_gene_ensembl")

#fetch protein coding gene names, including symbol and biotype for clarity
protein_coding_genes <- getBM(
  attributes = c("ensembl_gene_id", "hgnc_symbol", "gene_biotype"),
  filters = "biotype",
  values = "protein_coding",
  mart = ensembl
)

glue("initial number of rows in gene annotation: {nrow(r_anno_df)}")
```

```
## initial number of rows in gene annotation: 62940
```

```r
glue("initial number of rows in counts: {nrow(raw_counts_df)}")
```

```
## initial number of rows in counts: 62940
```

```r
#perform the filtering on gene annotation dataset
r_anno_df <- r_anno_df[r_anno_df[, "ensembl_gene_id"] %in% protein_coding_genes[,"ensembl_gene_id"],]

#perform the filtering on raw count dataset
raw_counts_df <- as.data.frame(raw_counts_df[rownames(raw_counts_df) %in% protein_coding_genes[,"ensembl

glue("final number of rows in gene annotation: {nrow(r_anno_df)}")
```

```
## final number of rows in gene annotation: 22142
```

```r
glue("final number of rows in counts: {nrow(raw_counts_df)}")
```

```
## final number of rows in counts: 22142
```

```r
#check if all gene IDs match across the filtered datasets
check_IDs <- all(rownames(r_anno_df) == rownames(raw_counts_df))
glue("Gene IDs of row counts match with gene IDs of annotation data frame: {check_IDs}")
```

```
## Gene IDs of row counts match with gene IDs of annotation data frame: TRUE
```

```r
table(c_anno_df$condition)
```

```
##
##    case control
##      50      50
```

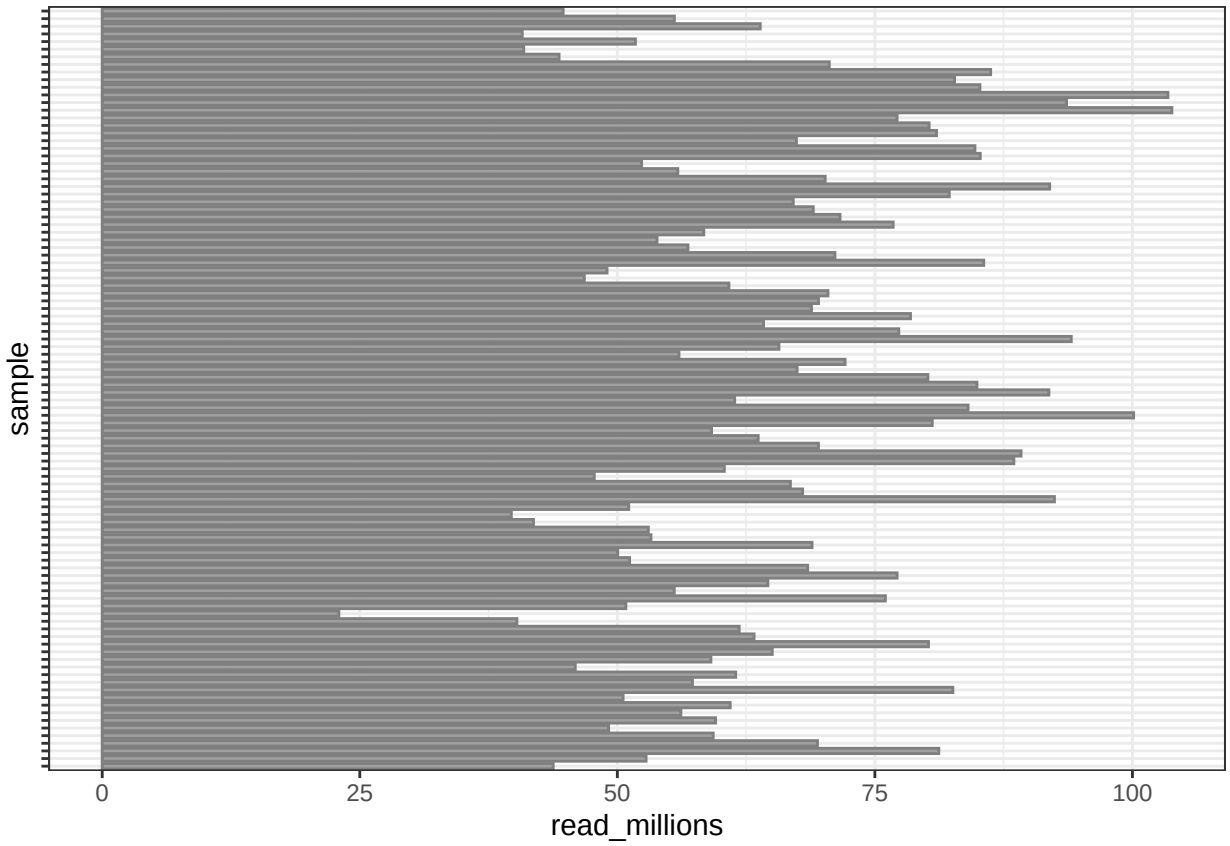
Task 3

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 less than1.5 for up-regulated genes and < (-1.5) for down-regulated genes and a log CPM >1. Relax the thresholds if no or few results are available. a Use the workflow we developed during the course; b Filter raw counts data retaining only genes with a raw count >20 in at least 5 Cases or 5 Control samples; c Create a volcano plot of your results;d Create an annotated heatmap focusing only on up- and down-regulated genes.

Plot Depth per sample

```r
suppressMessages(library(ggplot2))
suppressMessages(library(dplyr))

## Check the library size of each sample
size_df <- data.frame("sample"=colnames(raw_counts_df),
                      "read_millions"=colSums(raw_counts_df)/1000000)
```

```r
# Plot the depth per sample for the sampled subset
ggplot(data = size_df, aes(sample, read_millions)) +
  geom_bar(stat = "identity", fill = "grey50", colour = "grey50", width = 0.7, alpha = 0.7) +
  coord_flip() +
  theme_bw() +
  theme(axis.text.y = element_blank())
```



Samples show substantial variability in sequencing depth. This will need to be taken into account to prevent bias in differential expression analysis.

Filter out genes with counts lower than 20 in 90 or more replicates in both conditions.

```r
suppressMessages(library(tidyr))
# count threshold
count_thr <- 20
# number of replicates with more counts than the count threshold
repl_thr <- 10 # since we have 100 samples, I decided to increase the minimum number of replicates to 1

filter_vec <- apply(raw_counts_df,1,
    function(y) max(by(y, c_anno_df$condition, function(x) sum(x>=count_thr))))
# see statistics for the filtering

glue("number of genes with {count_thr} counts or more in N replicates")
```

```
## number of genes with 20 counts or more in N replicates
```

```
table(filter_vec)
```

```
## filter_vec
##     0     1     2     3     4     5     6     7     8     9    10    11    12
##  4176   419   207   142   106    84    65    64    45    60    55    47    54
##    13    14    15    16    17    18    19    20    21    22    23    24    25
##    33    39    47    40    38    40    37    39    42    38    42    30    40
##    26    27    28    29    30    31    32    33    34    35    36    37    38
##    35    44    36    41    37    46    40    47    49    45    57    55    38
##    39    40    41    42    43    44    45    46    47    48    49    50
##    62    70    73    75    71    76   110   120   166   181   345 14194
```

E.g. 4176 genes have no replicates with counts >= threshold in either condition E.g. 14194 genes have 50 replicates with counts >= threshold in one or the other condition

Show initial and final number of genes

```
filter_counts_df <- raw_counts_df[filter_vec>=repl_thr,,drop=FALSE]
# check the dimension of the filtered matrix

# apply the filter on gene annotation
filter_anno_df <- r_anno_df[rownames(filter_counts_df),,drop=FALSE]

glue("Old number of genes: {nrow(raw_counts_df)} ")
```

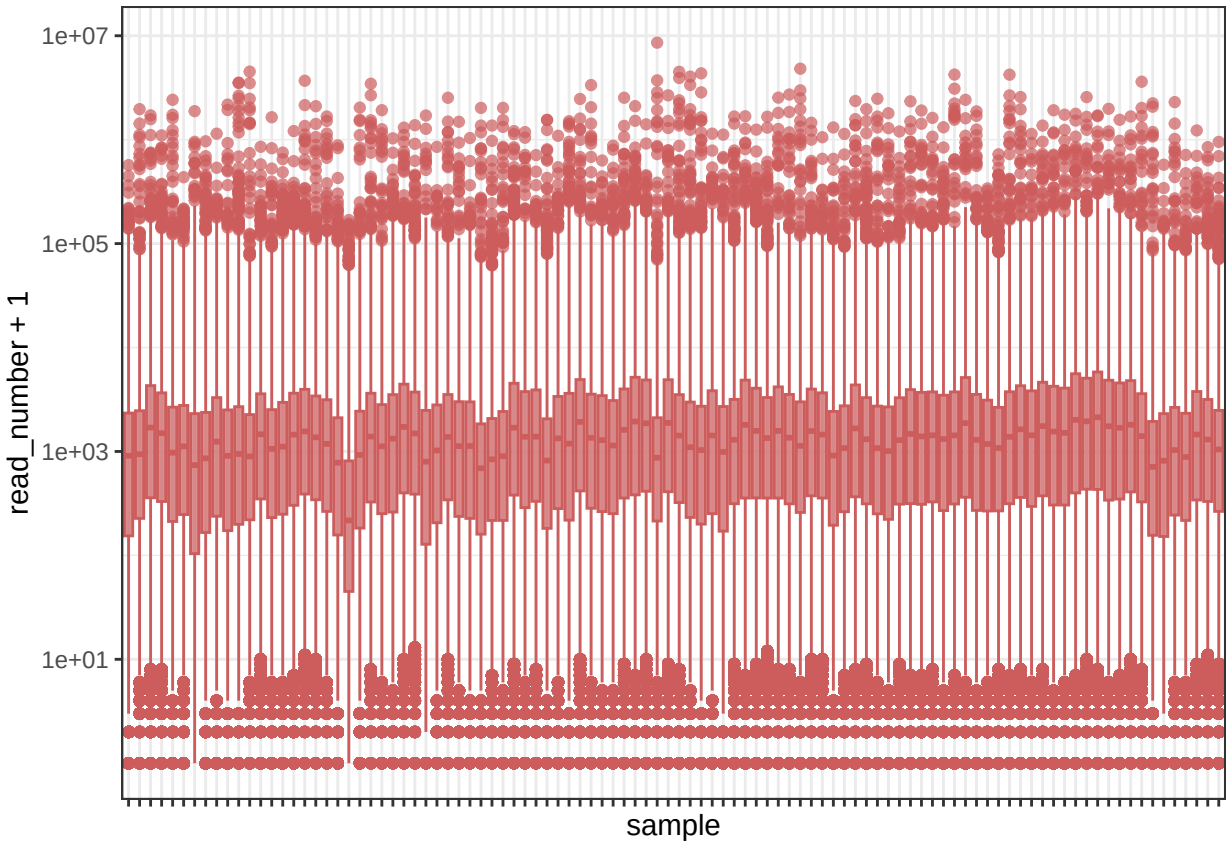
```
## Old number of genes: 22142
```

```
glue("New number of genes: {nrow(filter_counts_df)} ")
```

```
## New number of genes: 16774
```

Plot boxplots of gene expression in different samples

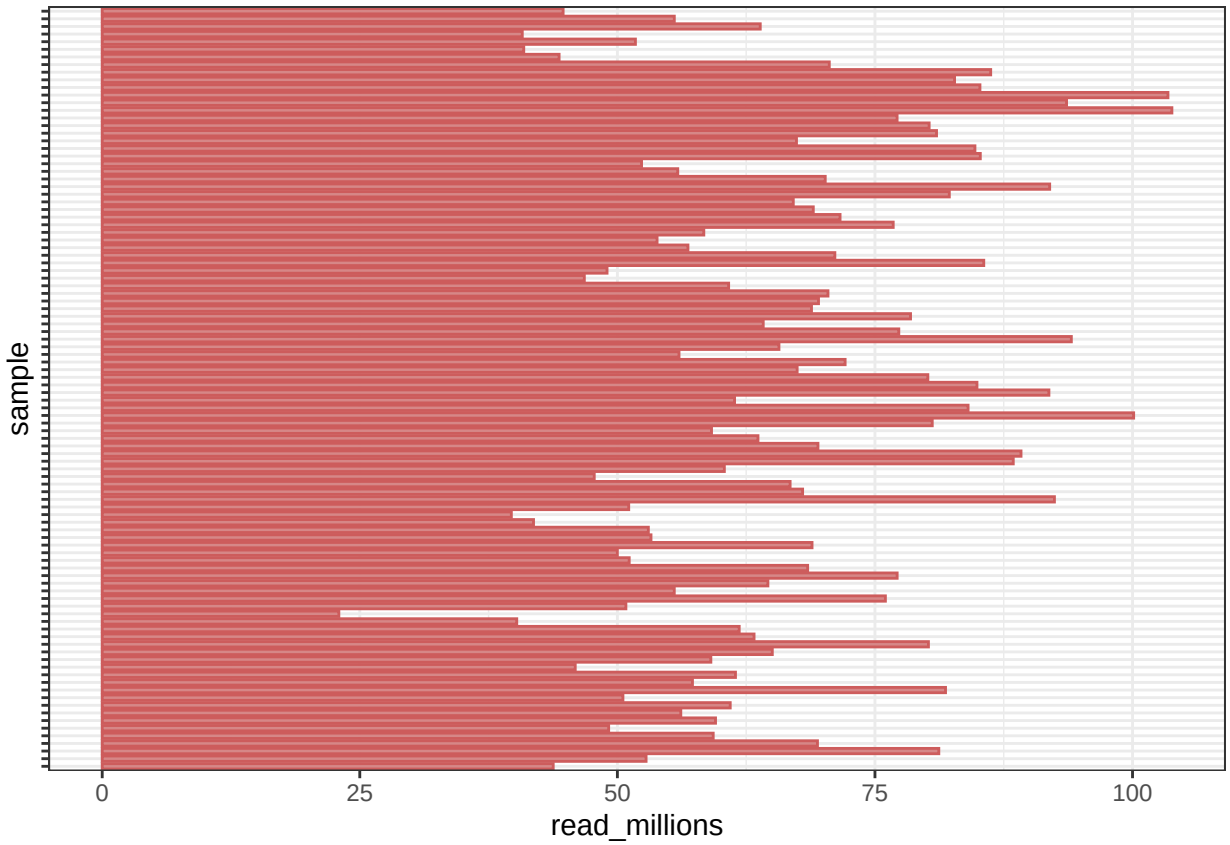
```
## Boxplot of gene counts after filtering absent genes

# gather is a function from the tidyr package
long_counts_df <- gather(filter_counts_df, key = "sample", value = "read_number")

# Plot the filtered data with vertical x-axis labels
ggplot(data=long_counts_df, aes(sample, read_number + 1)) +
   geom_boxplot(colour="indianred", fill="indianred", alpha=0.7) +
   theme_bw() +
   scale_y_log10() +
   theme(axis.text.x = element_blank())
```



Before normalization, discrepancies of gene distributions across different samples are evident. We expect these differences to come from technical variability, so it is important to address these before differential expression analysis to avoid biasing our results.

c

```
## Check the library size of each sample after filtering
size_df <- data.frame("sample"=colnames(filter_counts_df),
                      "read_millions"=colSums(filter_counts_df)/1000000)

ggplot(data=size_df,aes(sample,read_millions)) +
   geom_bar(stat="identity",fill="indianred",colour="indianred",width=0.7,alpha=0.7)+
   coord_flip()+
   theme_bw()+
   theme(axis.text.y = element_blank())
```

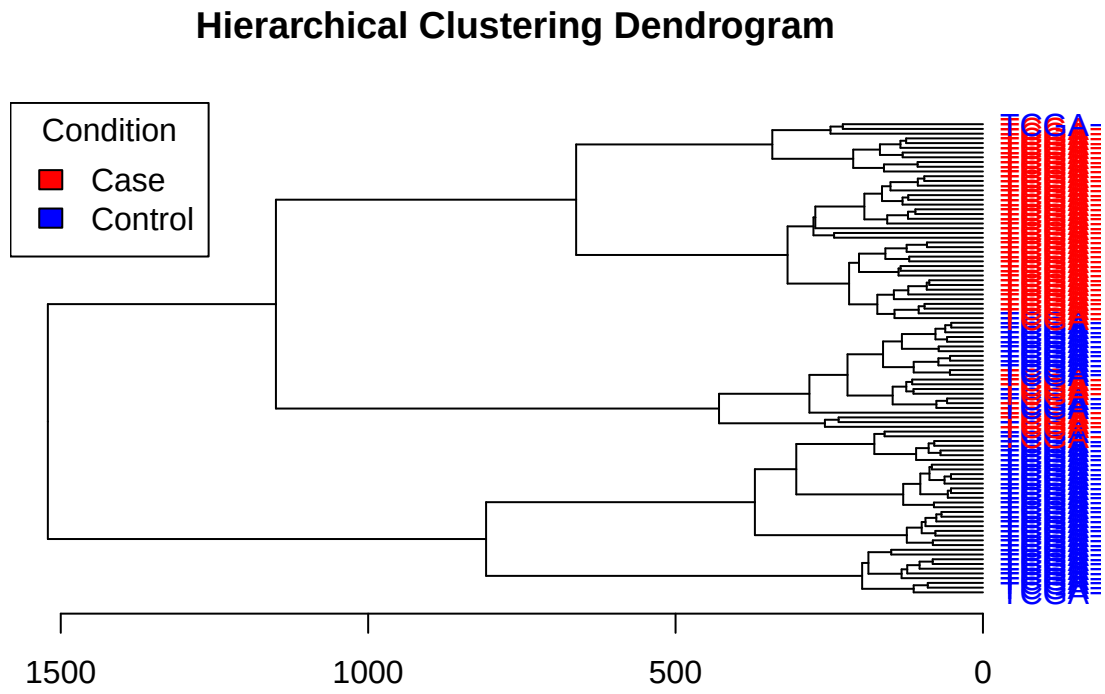


Gene filtering did not significantly affect the distribution and magnitude of total number of reads across samples.

Perform hierarchical clustering on filtered count data and generate dendogram

```
suppressMessages(library(dendextend))

# Perform hierarchical clustering
clu_data <- scale(t(filter_counts_df))
dd <- dist(clu_data, method = "euclidean")
hc <- hclust(dd, method = "ward.D")

# convert hclust object to a dendrogram
dend <- as.dendrogram(hc)

# Match labels in the dendrogram with their condition
dend_labels <- labels(dend)

# Match the labels with the condition in `c_anno_df`
match_idx <- match(dend_labels, c_anno_df$sample)
sample_conditions <- c_anno_df$condition[match_idx]

# Assign colors based on condition
sample_colors <- ifelse(sample_conditions == "case", "red", "blue")
```

```r
# Apply the colors to the dendrogram labels
labels_colors(dend) <- sample_colors

# Plot the dendrogram with colored labels
plot(dend, main = "Hierarchical Clustering Dendrogram", horiz = TRUE)

legend("topleft", legend = c("Case", "Control"), fill = c("red", "blue"), title = "Condition")
```

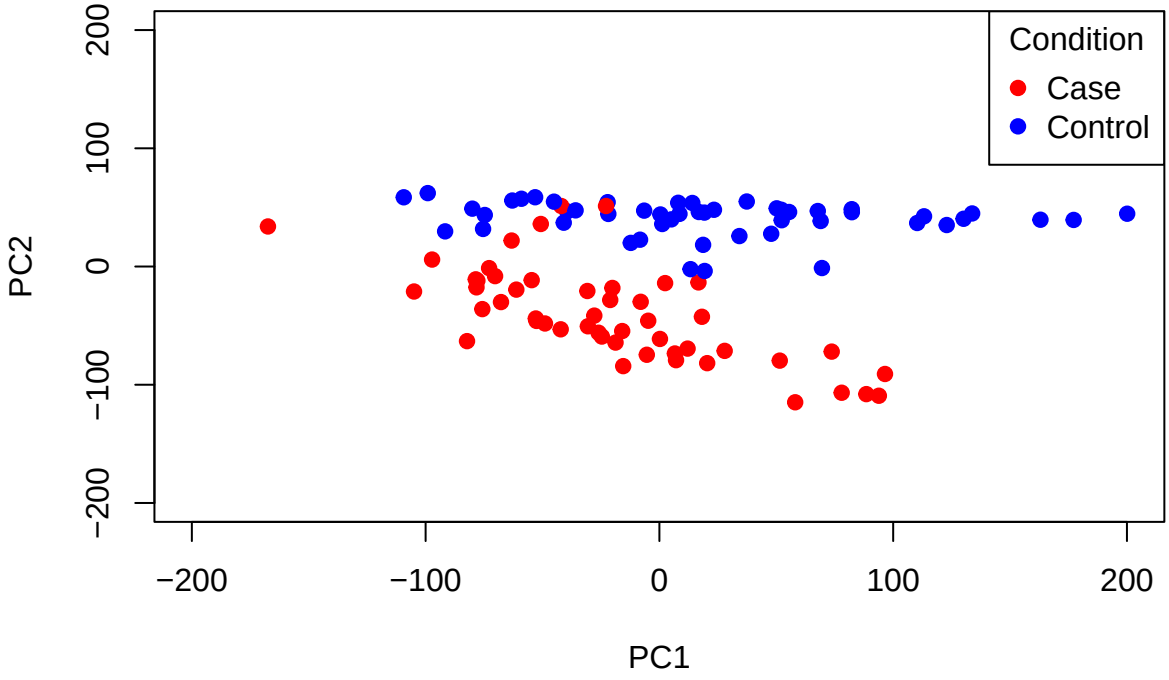


THe filtered unnormalized data presents some degree of mixing between the case and control groups. This could be technically or biologically driven, or both. We must check whether this persists even after normalization.

Run PCA on filtered count data

```r
data.matrix <- filter_counts_df

# Assign colors based on condition
color <- rep(NA, ncol(filter_counts_df))
color[which(c_anno_df$condition == "case")] <- "red"
color[which(c_anno_df$condition == "control")] <- "blue"

# Perform PCA
data.PC <- prcomp(t(data.matrix), scale. = TRUE)
```

```r
# Plot the PCA
plot(data.PC$x[, 1:2], xlim = c(-200, 200), ylim = c(-200, 200), col = color, pch = 19, xlab = "PC1", yl

# Add a legend
legend("topright", legend = c("Case", "Control"), col = c("red", "blue"), pch = 19, title = "Condition")
```



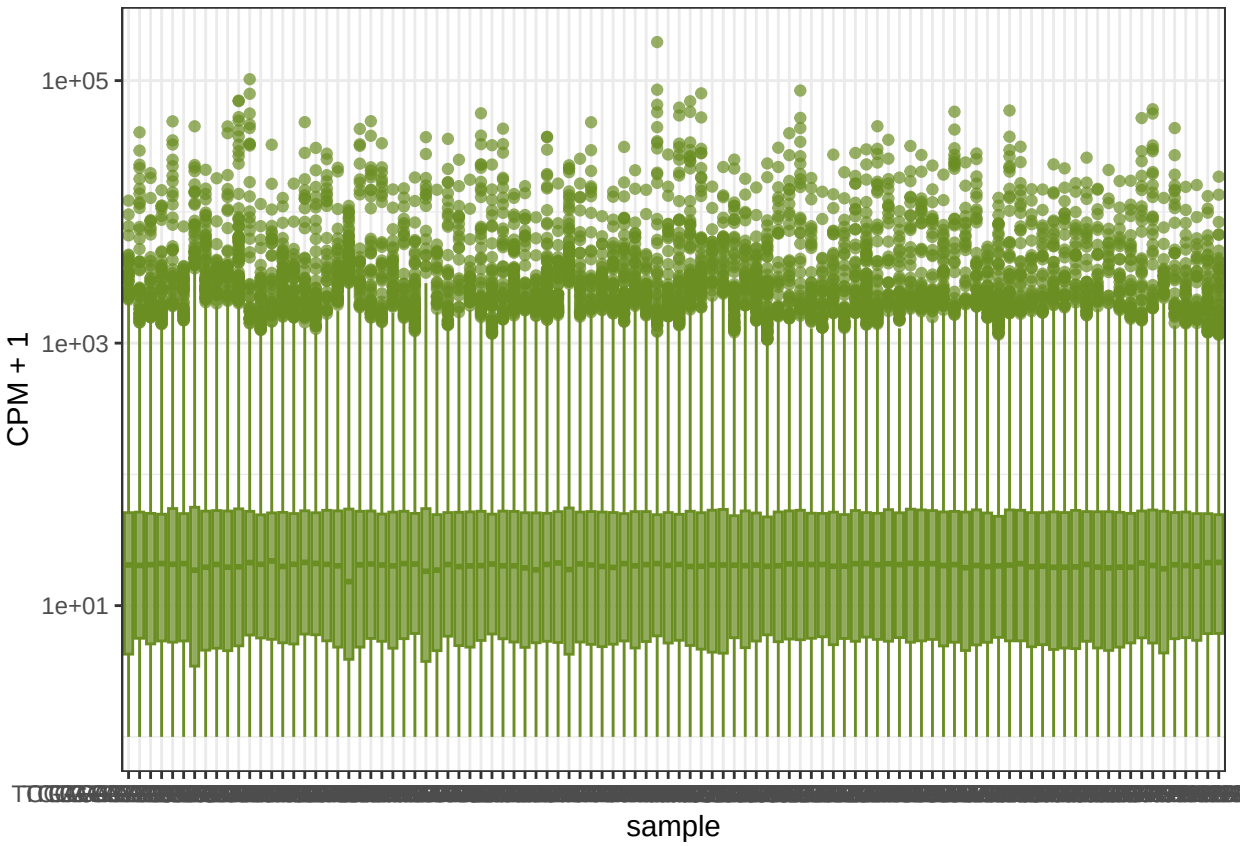
The two classes are well distinguishable according to the first two PCs, such that even a linear boundary in this space could provide a high accuracy separation. Similarly to the dendogram, we observe some case observations clustering with the control. It might be that a richer representation (more than 2 PCs) would be able to provide better separation. This could be assessed using specific models e.g. linear discriminant analysis.

Data normalization using TMM, conversion to CPM, boxplots of gene distributions

TMM allows for both intra- and inter sample-normalization by taking into account: - gene size - variability across genes - library size - number of genes expressed in each sample

```r
## DEG analysis with edgeR
suppressMessages(library("edgeR"))
# create a DGRList object
edge_c <- DGEList(counts=filter_counts_df,group=c_anno_df$condition,samples=c_anno_df,genes=filter_anno
# normalization with the edgeR package (TMM method)
edge_n <- calcNormFactors(edge_c,method="TMM")
# create a cpm table (normalized expression values)
```

```r
cpm_table <- as.data.frame(round(cpm(edge_n),2))
# look at the boxplot distribution of gene expression signals after normalization
long_cpm_df <- gather(cpm_table, key = "sample", value = "CPM")

ggplot(data=long_cpm_df,aes(sample,CPM+1)) +
   geom_boxplot(colour="olivedrab",fill="olivedrab",alpha=0.7)+
   theme_bw()+
   scale_y_log10()
```

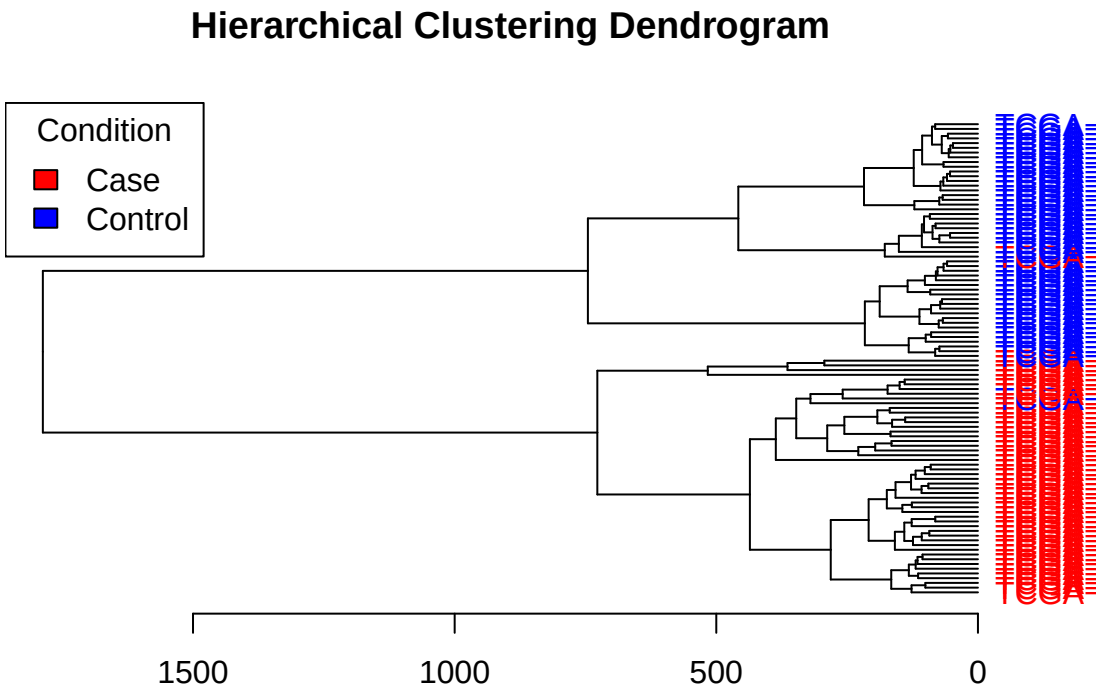


TMM and CPM normalization markedly improved the comparability of gene distributions across samples, allowing for better comparisons and DE analysis.

Check clustering of normalized data using the dendogram

```r
clu_data <- scale(t(cpm_table))
dd <- dist(clu_data, method = "euclidean")
hc <- hclust(dd, method = "ward.D")

dend <- as.dendrogram(hc)

dend_labels <- labels(dend)

match_idx <- match(dend_labels, c_anno_df$sample)
sample_conditions <- c_anno_df$condition[match_idx]
```

```r
sample_colors <- ifelse(sample_conditions == "case", "red", "blue")

labels_colors(dend) <- sample_colors

plot(dend, main = "Hierarchical Clustering Dendrogram", horiz = TRUE)

legend("topleft", legend = c("Case", "Control"), fill = c("red", "blue"), title = "Condition")
```

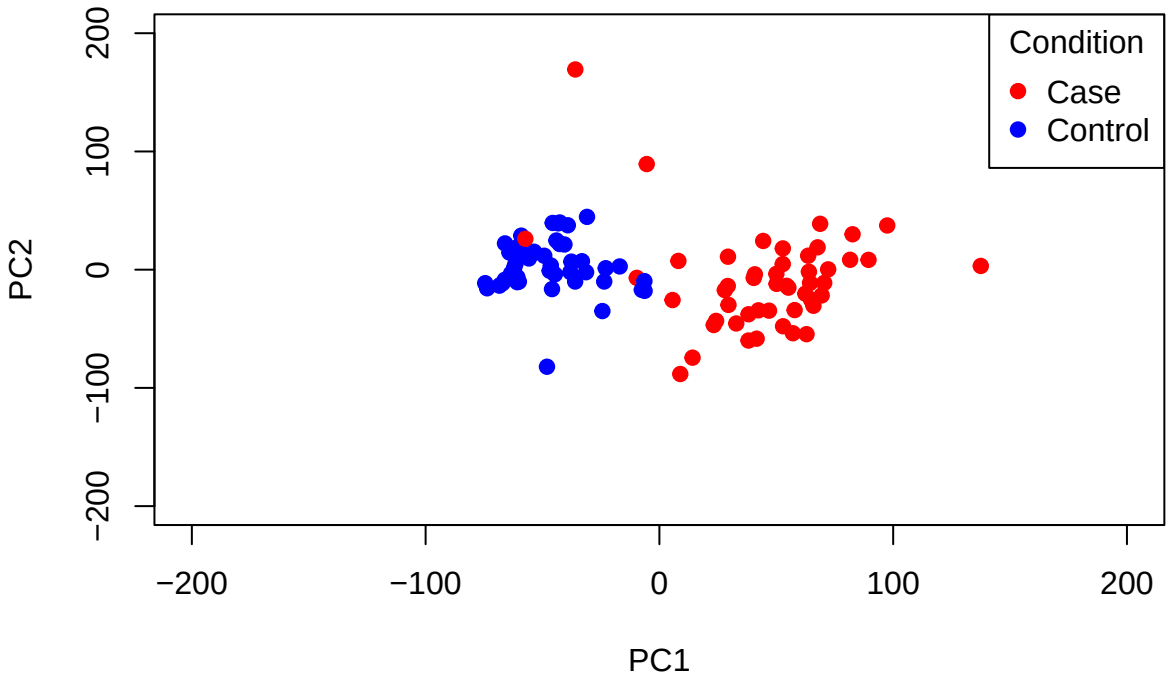


Normalization improved the separation of case and control in the dendogram, although a there is still a small amount of case samples clustering in the control.

Check clustering of normalized data using the dendogram

```r
data.matrix <- cpm_table

# Assign colors based on condition
color <- rep(NA, ncol(filter_counts_df))
color[which(c_anno_df$condition == "case")] <- "red"
color[which(c_anno_df$condition == "control")] <- "blue"

# Perform PCA
data.PC <- prcomp(t(data.matrix), scale. = TRUE)
```

```r
# Plot PCA
plot(data.PC$x[, 1:2], xlim = c(-200, 200), ylim = c(-200, 200), col = color, pch = 19, xlab = "PC1", yl

legend("topright", legend = c("Case", "Control"), col = c("red", "blue"), pch = 19, title = "Condition")
```



Although the boundary itself may not have improved in the first 2 components, after normalization, clusters look definitely more tight, probably reflecting the removal of technical sources of variation. Also in this case we can still observe the presence of mis-clustered observations. This effect might be real biological signal or be some technical artifact.

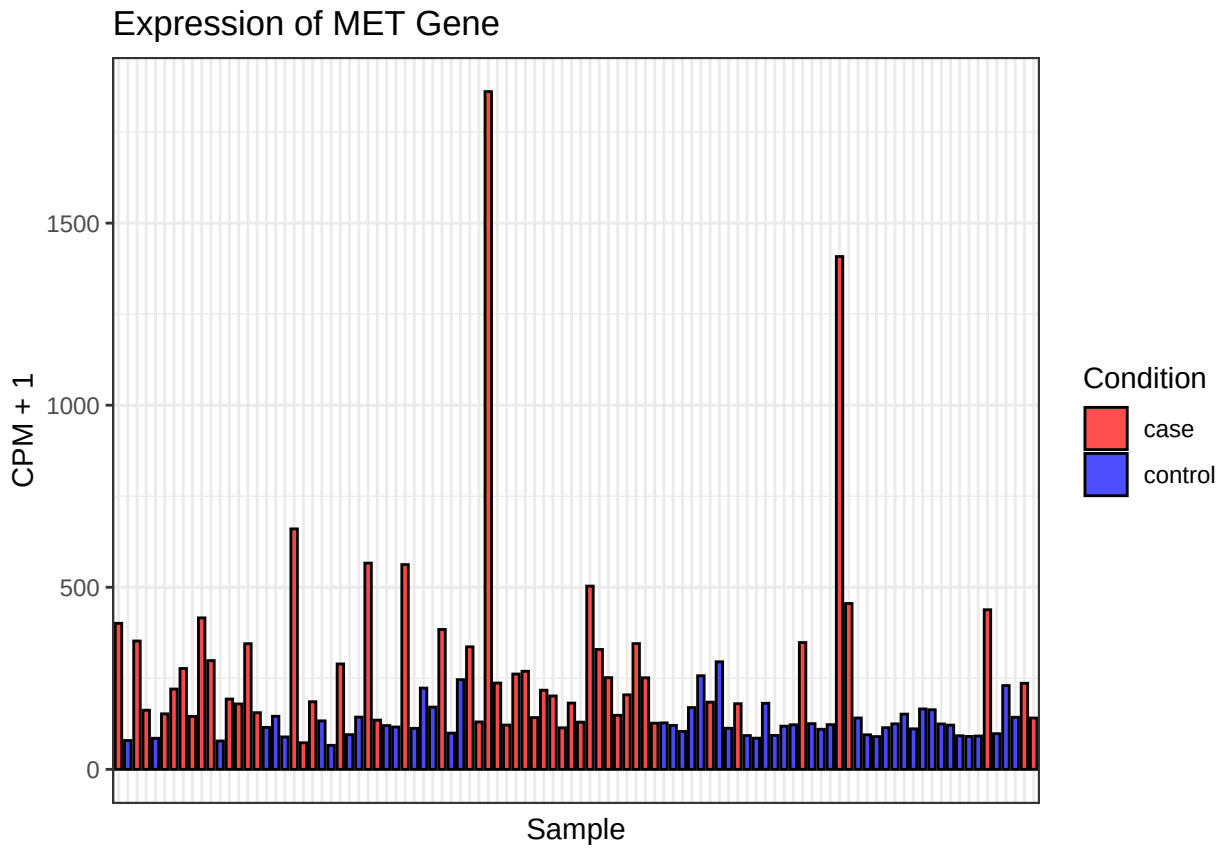
Check expression of marker gene of PRCC known to be upregulated with respect to control

```r
# check expression of marker gene of renal-cell papillary carcinoma MET
"Title: Comprehensive Molecular Characterization of Papillary Renal-Cell Carcinoma
Journal: The New England Journal of Medicine (2016)
Authors: Durinck, S., Stawiski, E. W., Pavía-Jiménez, A., Modrusan, Z., et al.
DOI: 10.1056/NEJMoa1505917"
```

```
## [1] "Title: Comprehensive Molecular Characterization of Papillary Renal-Cell Carcinoma\nJournal: The
```

```r
# Identify the marker gene for renal-cell papillary carcinoma
met <- cpm_table[which(rownames(cpm_table) == "ENSG00000105976"), ]
long_met_df <- gather(as.data.frame(met), key = "sample", value = "CPM")
```

```r
# Add a condition column to `long_met_df` based on `c_anno_df`
long_met_df <- long_met_df %>%
  left_join(c_anno_df, by = "sample")

# Assign colors based on condition
long_met_df$color <- ifelse(long_met_df$condition == "case", "blue", "red")

# Create the plot with colored bars and no sample names on x-axis
ggplot(data = long_met_df, aes(x = sample, y = CPM + 1, fill = condition)) +
  geom_bar(stat = "identity", colour = "black", width = 0.7, alpha = 0.7) +
  scale_fill_manual(values = c("case" = "red", "control" = "blue")) +
  theme_bw() +
  theme(axis.text.x = element_blank(),  # Remove sample names
        axis.ticks.x = element_blank()) +  # Remove x-axis ticks as well
  labs(fill = "Condition", title = "Expression of MET Gene", x = "Sample", y = "CPM + 1")
```



The MET gene is upregulated in renal papillary cell carcinoma; our preprocessed data appears to suggest the same, although some case observations have expression levels comparable to the control. It might indicate potential heterogeneity of the tumor subtype across individuals, or different stages, technical artifacts, etc.

```r
suppressMessages(library(pheatmap))

# Filter raw counts data retaining only genes with a raw count >20 in at least 5 Cases or 5 Control sam
filter_counts <- rowSums(edge_n$counts >= 20) >= 5
edge_filtered <- edge_n[filter_counts, ]
```

```r
# Define the experimental design matrix
design <- model.matrix(~0 + group, data = edge_filtered$samples)
colnames(design) <- levels(edge_filtered$samples$group)
rownames(design) <- edge_filtered$samples$sample

# Estimate dispersion and fit the model
edge_d <- estimateDisp(edge_filtered, design)
edge_f <- glmQLFit(edge_d, design)

# Define the contrast for differential expression analysis
contrast <- makeContrasts(case_vs_control = case - control, levels = design)

# Perform the differential expression test
edge_t <- glmQLFTest(edge_f, contrast = contrast)

# Extract the DEGs based on the specified thresholds
DEGs <- as.data.frame(topTags(edge_t, n = Inf))
DEGs$class <- "="
DEGs$class[which(DEGs$logCPM > 1 & DEGs$logFC > 1.5 & DEGs$FDR < 0.01)] <- "+"
DEGs$class[which(DEGs$logCPM > 1 & DEGs$logFC < -1.5 & DEGs$FDR < 0.01)] <- "-"
DEGs <- DEGs[order(DEGs$logFC, decreasing = TRUE), ]

# Check how many genes passed the filter
table(DEGs$class)
```

```
##
##     -     +     =
##   728  1182 14864
```

According to the thresholds of log FC, log CPM, and FDR:

728 genes have been found downregulated in case w.r.t. control

1182 genes have been found upregulated in case w.r.t. control 14864 have no significant different in GE levels across the 2 conditions

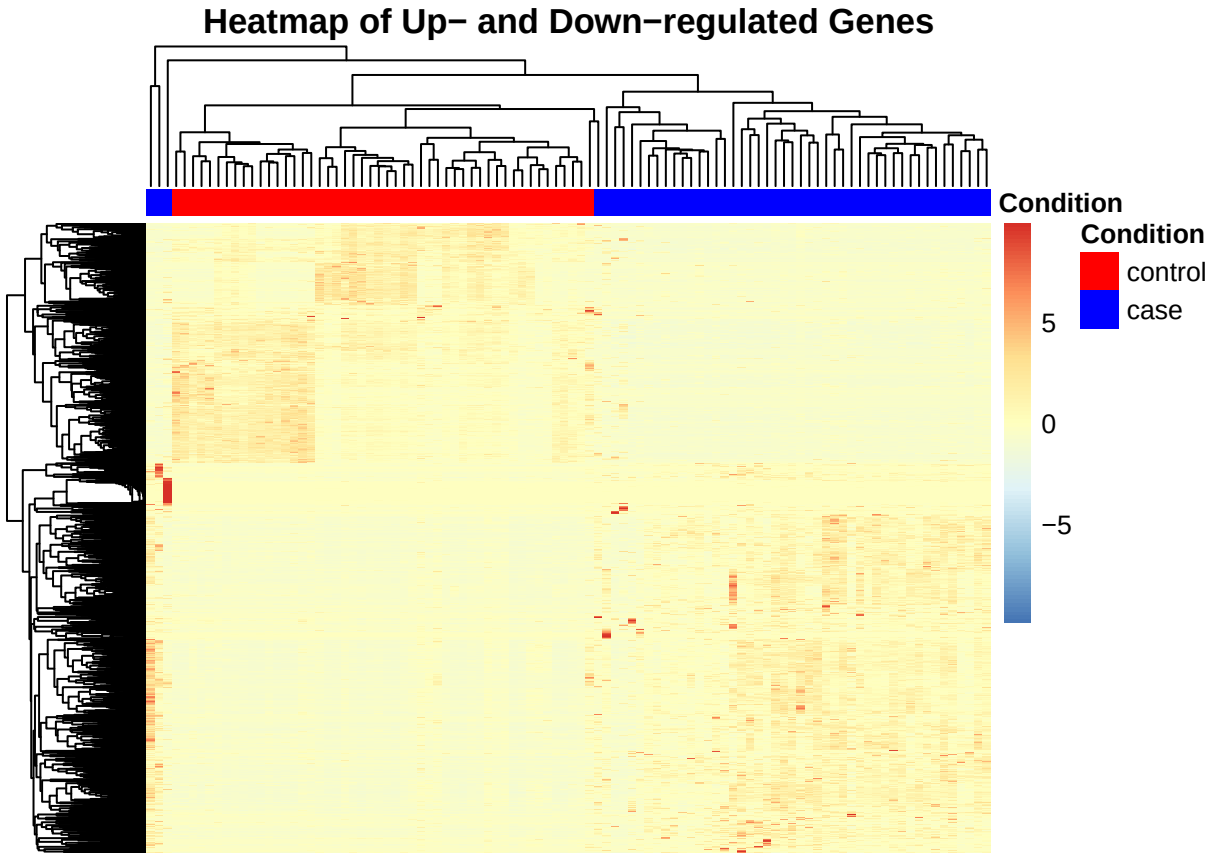
```r
# Create a Volcano plot to visualize the DEGs
volcano_plot <- ggplot(DEGs, aes(x = logFC, y = -log10(FDR))) +
  geom_point(aes(color = class), alpha = 0.6) +
  scale_color_manual(values = c("+" = "red", "-" = "blue", "=" = "grey70")) +
  theme_minimal() +
  labs(title = "Volcano Plot of Differentially Expressed Genes (Case vs Control)",
       x = "Log2 Fold Change",
       y = "-Log10 Adjusted P-value",
       color = "Regulation") +
  geom_vline(xintercept = c(-1.5, 1.5), linetype = "dashed", color = "black") +
  geom_hline(yintercept = -log10(0.01), linetype = "dashed", color = "black")

print(volcano_plot)
```

Thresholds and significant genes can be visualized in the volcano plot. A lot of genes are found DE. We could be more stringent and increase the magnitude of the fold change and-or FDR needed. Red: upregulated genes in case vs control Blue: downregulated genes in case vs control Grey observations in the center middle are a |logFC|< 1.5 Grey observations below the horizontal dashed line are not statically significant and also have small |logFC| Grey observations next to red and blue observations have logCPM <= 1

Create heatmap of DEGs

```
suppressMessages(library(pheatmap))

# Create a heatmap for the DEGs
filtered_DEGs <- DEGs[which(DEGs$class != "="), ]
deg_names <- rownames(filtered_DEGs)

# Extract the expression data for the filtered DEGs
deg_expr <- edge_filtered$counts[deg_names, ]

# Ensure the Condition column is a factor with levels in the correct order
annotation <- data.frame(Condition = factor(edge_filtered$samples$group, levels = c("control", "case")))
rownames(annotation) <- rownames(edge_filtered$samples)

# Define the colors manually for the annotations
annotation_colors <- list(
  Condition = c("control" = "red", "case" = "blue")
```

```
)

# Create the heatmap with explicit annotation colors
pheatmap(deg_expr,
         scale = "row",
         cluster_rows = TRUE,
         cluster_cols = TRUE,
         annotation_col = annotation,
         annotation_colors = annotation_colors,  # Apply the manually defined colors
         show_rownames = FALSE,
         show_colnames = FALSE,
         main = "Heatmap of Up- and Down-regulated Genes")
```



From the heatmap it clearly appears a distinction between case and control; in the first block a set of genes are downregulated w.r.t. the control, whereas in the second block genes are upregulated w.r.t. control. There appears to be individual variability within blocks, especially in the degree of upregulation in both groups with respect to the other group. We can see that the "mis-clustered" case observations in the control group have distinguishable expression levels, to some degree they look more similar to the case observations. This might due to an algorithmic limitation of the clustering algorithm rather than proper biological signal.

Save upregulated and downregulated genes for Enrichment analysis

```
# Export up-regulated and down-regulated DEGs to separate files
up_DEGs <- DEGs[which(DEGs$class == "+"), ]
```

```r
down_DEGs <- DEGs[which(DEGs$class == "-"), ]

#write.table(up_DEGs,file="up_DEGs.txt",row.names=F,col.names=T,sep="\t",quote=F)
#write.table(down_DEGs,file="down_DEGs.txt",row.names=F,col.names=T,sep="\t",quote=F)
#write.table(DEGs,file="DEGs.txt",row.names=F,col.names=T,sep="\t",quote=F)
```

```r
# Clear up memory for next steps
rm(list = ls())
```

Task 4

Perform gene set enrichment analysis using clusterProfiler R package. a Perform both GO (BP and MF) and WP analysis; b Report the top 10 enriched GO terms and the top 10 enriched WP pathways; resulting from both up- and down-regulated gene lists.

```r
suppressMessages(library(org.Hs.eg.db))
suppressMessages(library(biomaRt))
suppressMessages(library(clusterProfiler))
suppressMessages(library(enrichplot))
suppressMessages(library(ggnewscale))
suppressMessages(library(DOSE))
suppressMessages(library(pathview))
suppressMessages(library(tidyverse))

### Load results of DEG analysis
DEGs <- read.table("DEGs.txt", header = TRUE, sep = "\t", stringsAsFactors = FALSE)
table(DEGs$class)
```

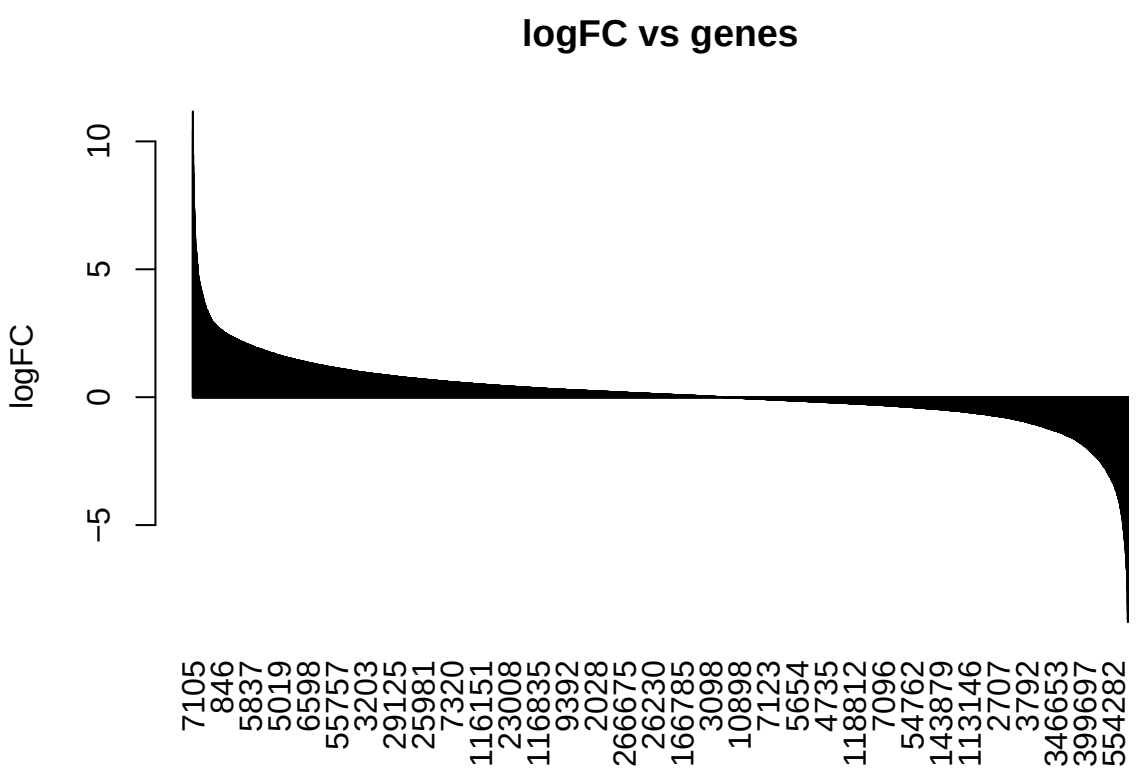
```
##
##     -     +     =
##   728  1182 14864
```

```r
# Load the up-regulated DEGs
up_DEGs <- read.table("up_DEGs.txt", header = TRUE, sep = "\t", stringsAsFactors = FALSE)

# Load the down-regulated DEGs
down_DEGs <- read.table("down_DEGs.txt", header = TRUE, sep = "\t", stringsAsFactors = FALSE)

### Use biomaRt to map Gene symbols, Entrez IDs, and Ensembl gene IDs
ensembl <- useEnsembl(biomart = "ensembl", dataset = "hsapiens_gene_ensembl")

# Function to map Ensembl IDs to Entrez IDs and merge with the DEGs data frame
add_entrez_ids <- function(df, ensembl) {
  # Retrieve Entrez Gene IDs
  convert <- getBM(attributes = c("ensembl_gene_id", "entrezgene_id"),
                   filters = "ensembl_gene_id",
                   values = df$ensembl_gene_id,
                   mart = ensembl)

  # Merge without duplicating columns
  df <- merge(df, convert, by = "ensembl_gene_id", all.x = TRUE)
```

```
  # Remove rows with NA Entrez IDs and duplicates
  df <- df[!is.na(df$entrezgene_id), ]
  df <- df[!duplicated(df$entrezgene_id), ]

  return(df)
}

# Add Entrez IDs to DEGs, up_DEGs, and down_DEGs
DEGs <- add_entrez_ids(DEGs, ensembl)
up_DEGs <- add_entrez_ids(up_DEGs, ensembl)
down_DEGs <- add_entrez_ids(down_DEGs, ensembl)

# Save the updated data frames (optional)
#write.table(DEGs, file = "DEGs_with_EntrezIDs.txt", row.names = FALSE, col.names = TRUE, sep = "\t", q
#write.table(up_DEGs, file = "up_DEGs_with_EntrezIDs.txt", row.names = FALSE, col.names = TRUE, sep = "
#write.table(down_DEGs, file = "down_DEGs_with_EntrezIDs.txt", row.names = FALSE, col.names = TRUE, sep

### GSEA analysis

## Create vector with ranks
ranks <- DEGs$logFC
ranks <- sort(ranks,decreasing = T)
names(ranks) <- DEGs$entrezgene_id
head(ranks)
```

```
##      7105     64102      8813     57147     55732      2268
## 11.170845 10.444198 10.018498  9.472736  9.470631  9.068154
```

```
# Create the barplot
barplot(sort(ranks, decreasing = TRUE),
        las = 3,              # Rotate the labels on the x-axis
        names.arg = NULL,     # Remove the default names below the bars
        ylab = "logFC",       # Label for the y-axis
        main = "logFC vs genes",  # Title of the plot
)# Label for the x-axis
```



The barplot representing the distribution of log FC ratio shows positive and negative peaks which correspond to respectively upregulated and downregulated genes in the case samples compared to the control.

```
# Function for plotting go enrichment (it has to be called many times)
plot_go_enrichment <- function(enrich_obj) {
  ## Visualize the top 10 enriched terms with a barplot
  print(barplot(enrich_obj,showCategory=10))

  ## Visualize the top 10 enriched terms with a dotplot
  print(dotplot(enrich_obj, showCategory=20))

  ## Visualize the genes associated with the top 2 enriched terms with a heatplot
  print(heatplot(enrich_obj,showCategory = 3))

  #### Visualize with a network view the connection between the top 10 enriched terms
  x2 <- pairwise_termsim(enrich_obj)
  print(emapplot(x2))
}
```

```
# GO Biological Process (BP) for up-regulated genes
go_bp_up <- enrichGO(gene = up_DEGs$external_gene_name,
                     OrgDb = org.Hs.eg.db,
                     keyType = 'SYMBOL',
                     ont = "BP",
                     pAdjustMethod = "BH",
                     pvalueCutoff = 0.05,
```

```
                    qvalueCutoff = 0.05) # p value cutoff

#plot_go_enrichment(go_bp_up)
```

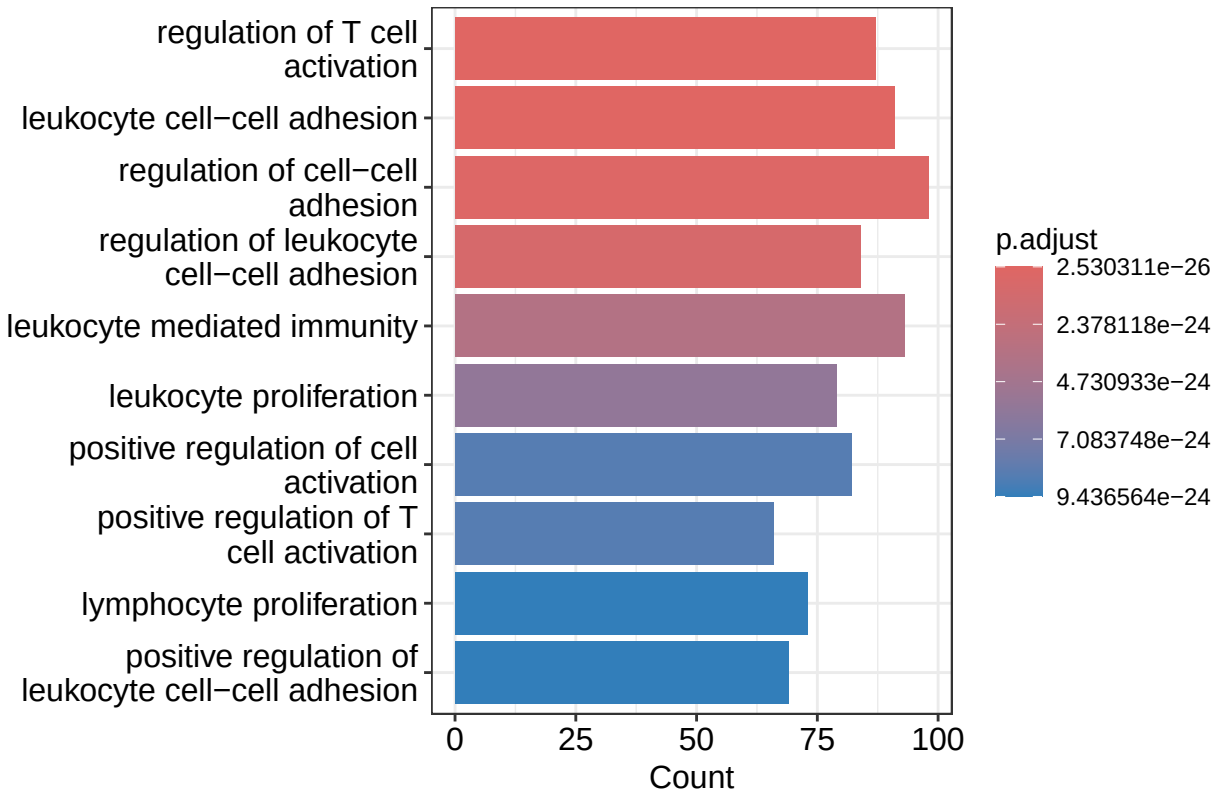
```
library(glue)
glue("Number of terms satisfying adjusted p. value cutoff: {dim(go_bp_up)[1]}")
```

```
## Number of terms satisfying adjusted p. value cutoff: 820
```

```
length(go_bp_up$p.adjust)
```

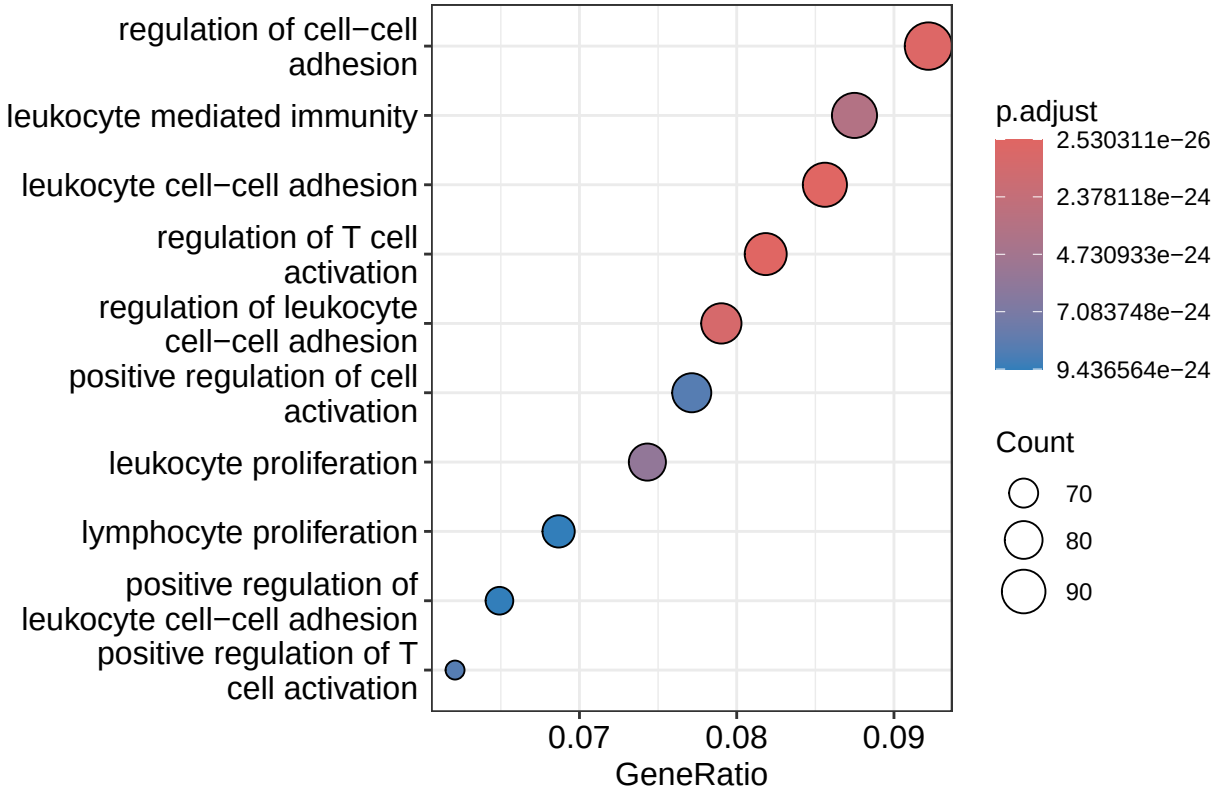
```
## [1] 820
```

```
print(barplot(go_bp_up,showCategory=10))
```



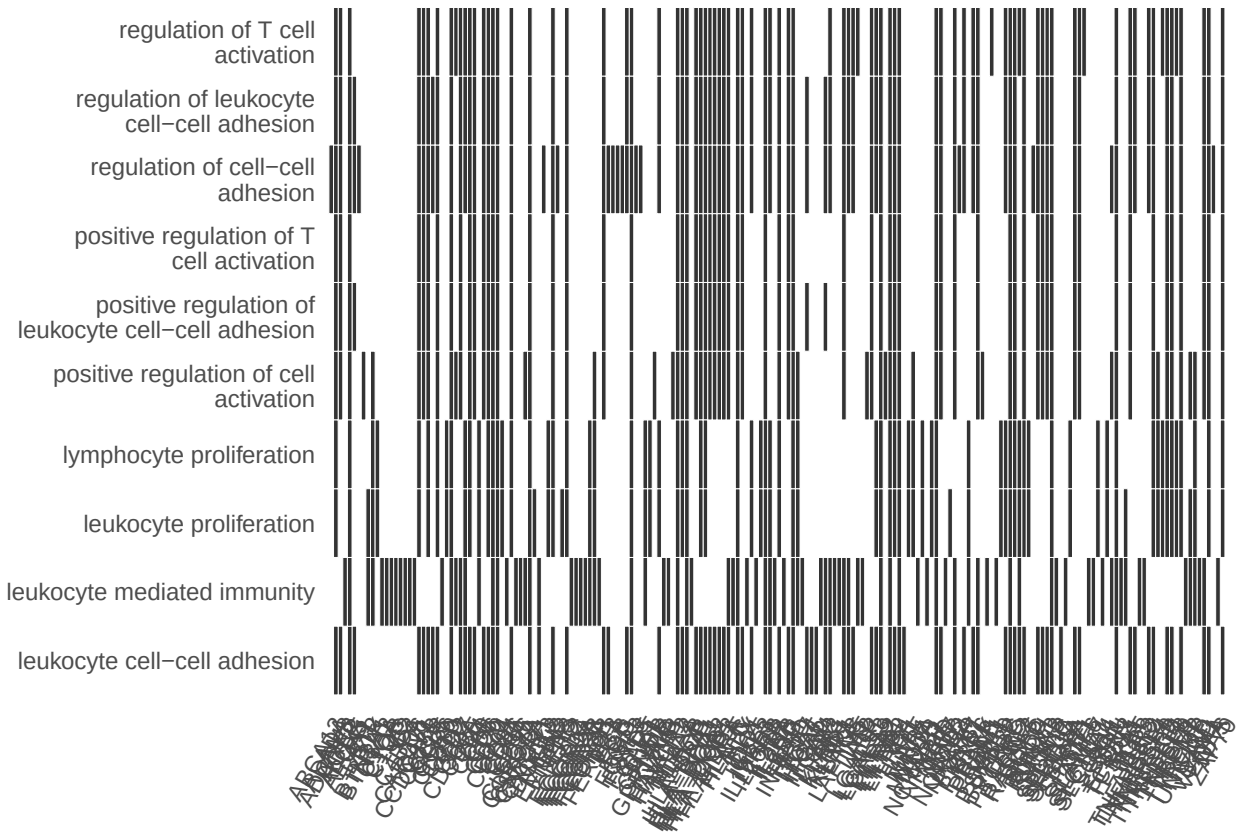
The top 10 enriched terms in the up-regulated DEGs are highly statistically significant according to the adjusted p value. These terms are concerned with immune system deregulation and cell motility, both of which are expected to be involved in cancer. The immune system is physiologically involved in killing cancer cells and controlling their expansion. In cancer, these protective mechanisms may be evaded, altered or made ineffective. Moreover, cancer cells up-regulate genes related to invasion, movement and adhesion, allowing for dislocation into the blood stream and causing metastasis.

```
print(dotplot(go_bp_up, showCategory=10))
```



The Gene Ratio represents the percentage of DEGs belonging to the term. In this case, for example, 10% of the up-regulated DEGs belong to the term "regulation of cell-cell adhesion". Here, larger counts tend to associate with larger gene ratio and lower adjusted p-value.

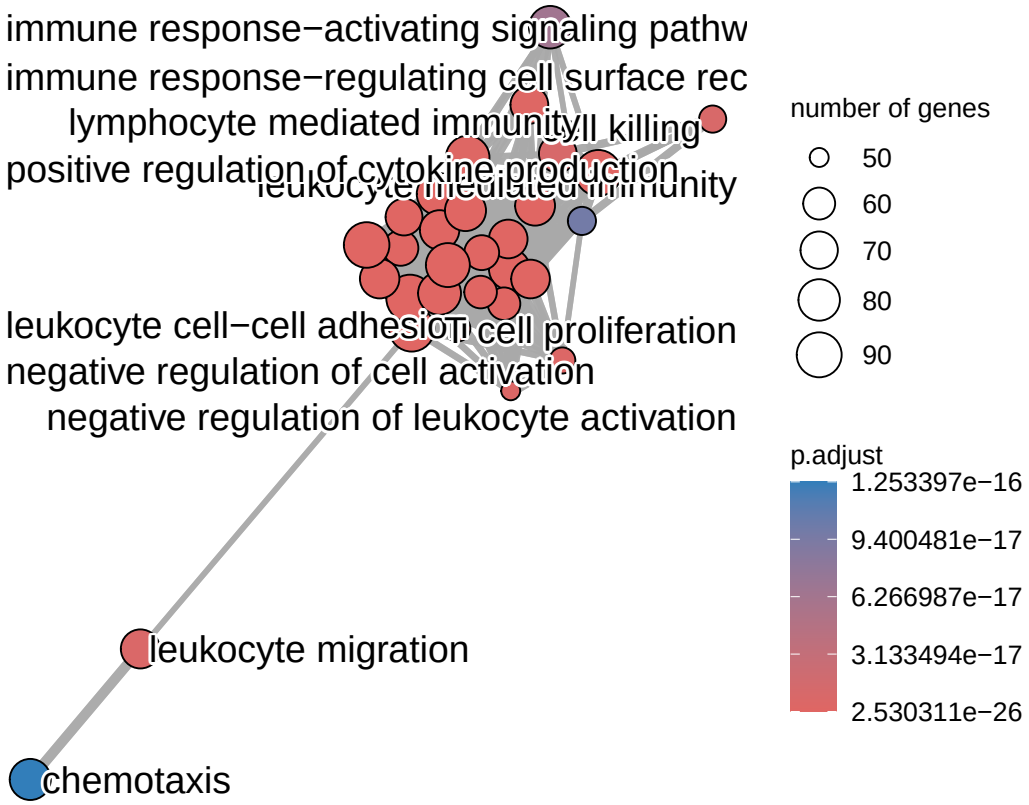
```
print(heatplot(go_bp_up,showCategory = 10))
```



This plot enables to see the overlap in genes among different terms, that is the redundancy. The largest groups of redundant genes involve 1) immune cell adhesion, movement and activation and 2) immune cell proliferation, suggesting a different genetic program for these two biological processes.

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

```
## Warning: ggrepel: 18 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



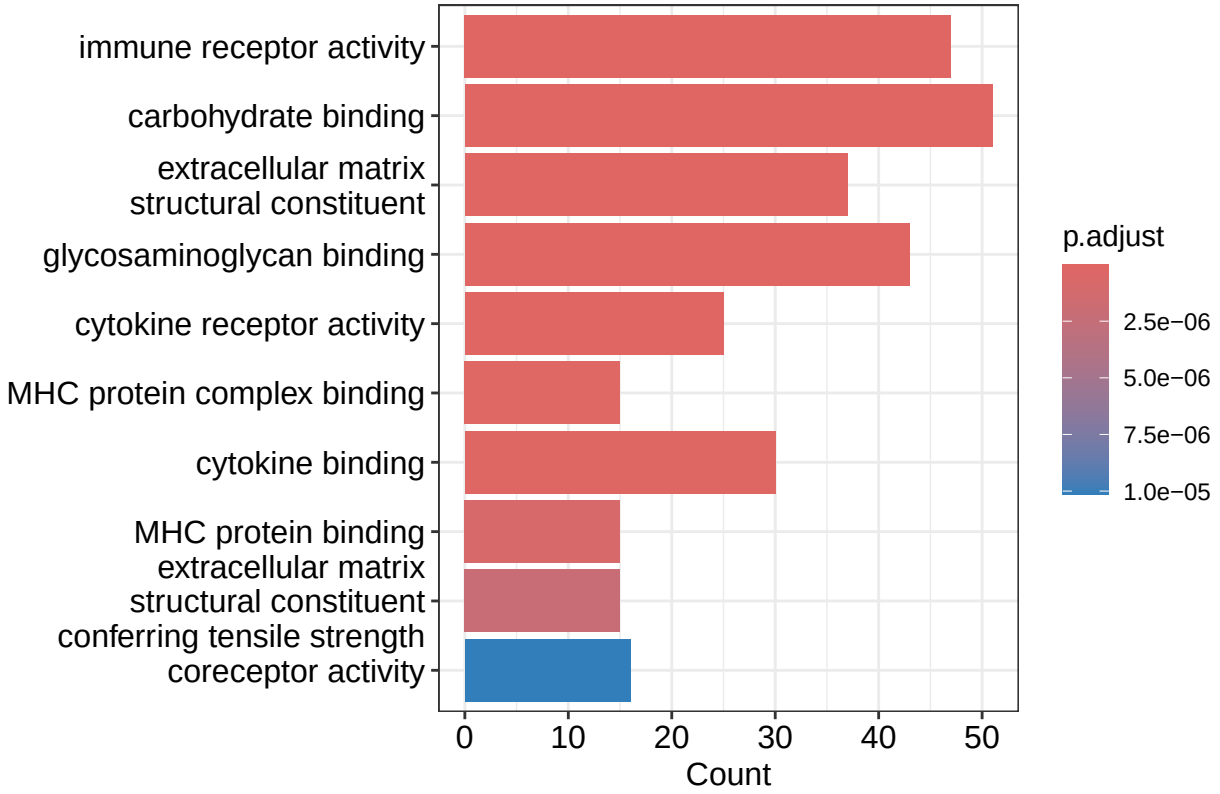
Here we observe the undirected graph, where edge sizes reflect overlap in genes between pairs of terms, and the visualization is optimized to identify functional clusters. For example, here we can see a smaller cluster related to cell migration, and a larger cluster more related to the immune response.

The remaining plots of this type will be left mostly uncommented unless interesting or different information is observed.

```
# GO Molecular Function (MF) for up-regulated genes
go_mf_up <- enrichGO(gene = up_DEGs$external_gene_name,
                     OrgDb = org.Hs.eg.db,
                     keyType = 'SYMBOL',
                     ont = "MF",
                     pAdjustMethod = "BH",
                     pvalueCutoff = 0.05,
                     qvalueCutoff = 0.05)

glue("Number of terms satisfying adjusted p. value cutoff: {dim(go_mf_up)[1]}")
```

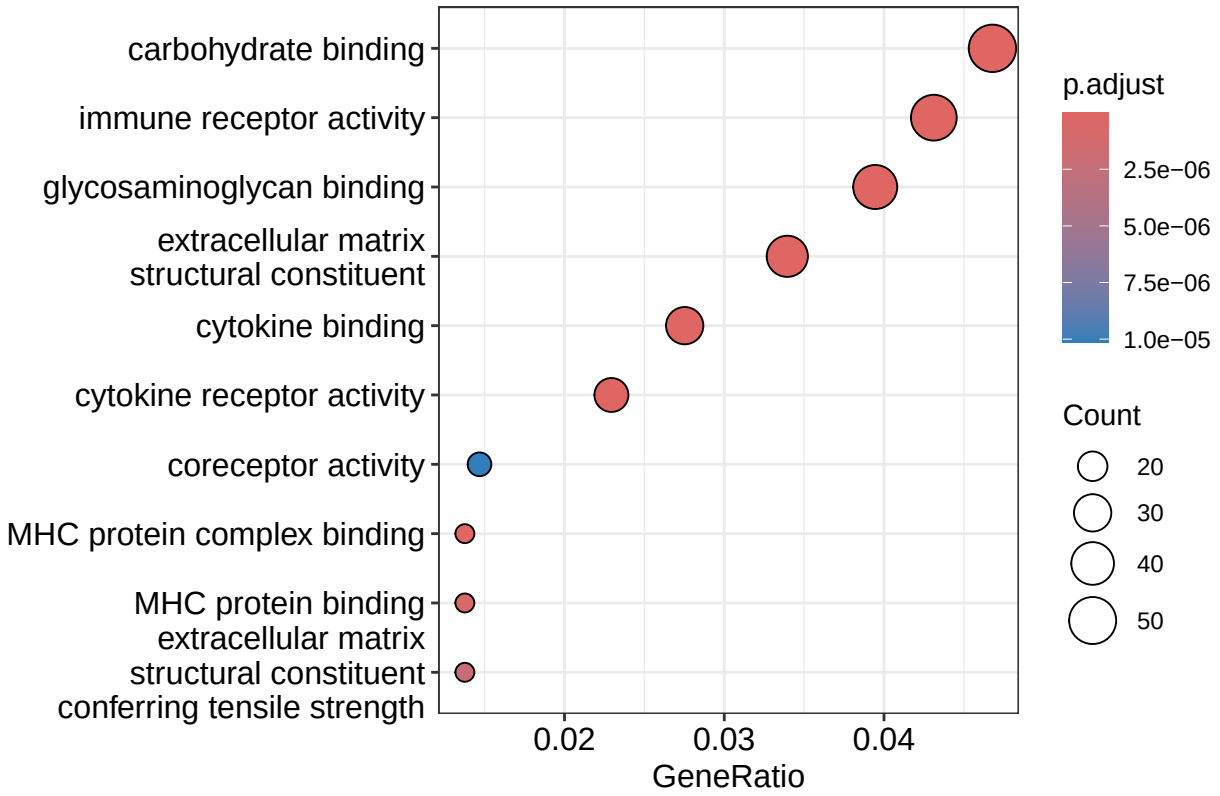
```
## Number of terms satisfying adjusted p. value cutoff: 86
```

```
print(barplot(go_mf_up,showCategory=10))
```

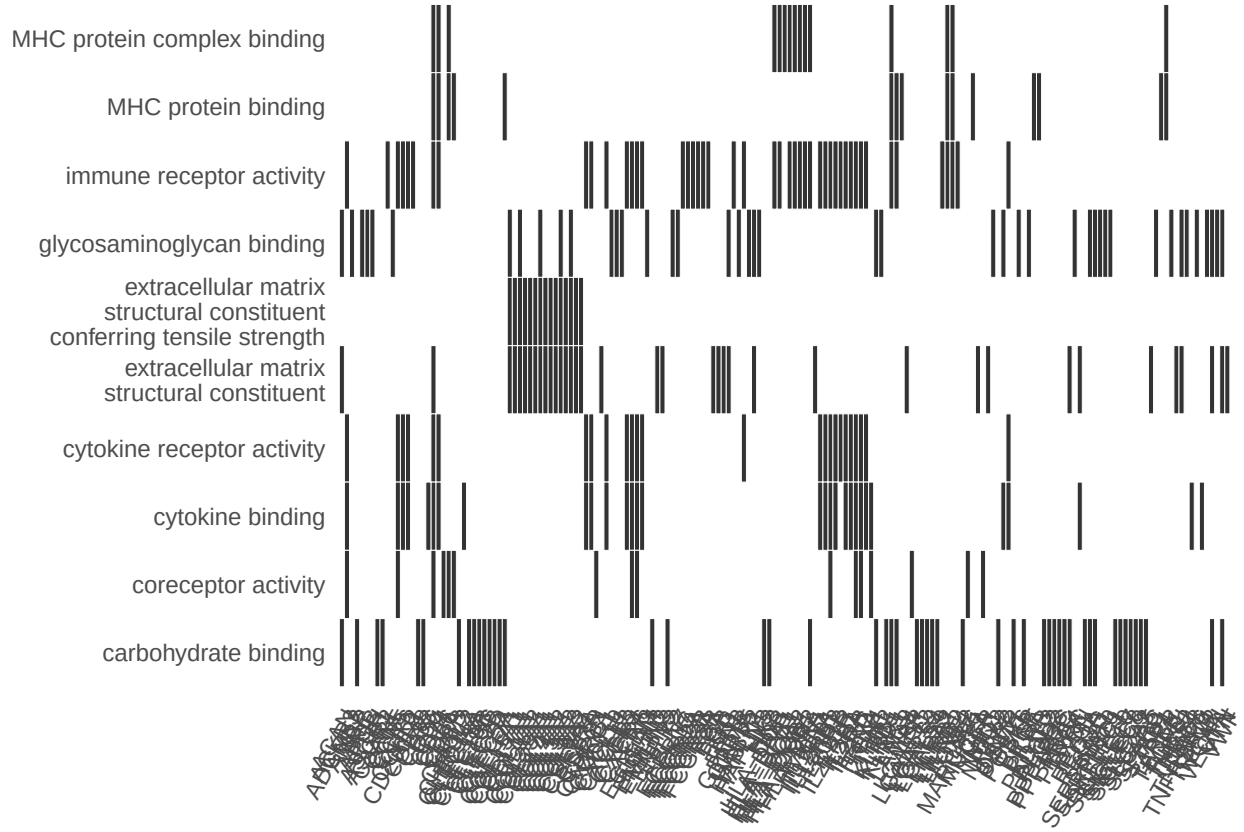


Given that we're using molecular function (MF) instead of biological process (BP) the term names are more specific and granular. Also in this case we see that the most deregulated patterns involve cell motility and the immune system.

```
print(dotplot(go_mf_up, showCategory=10))
```

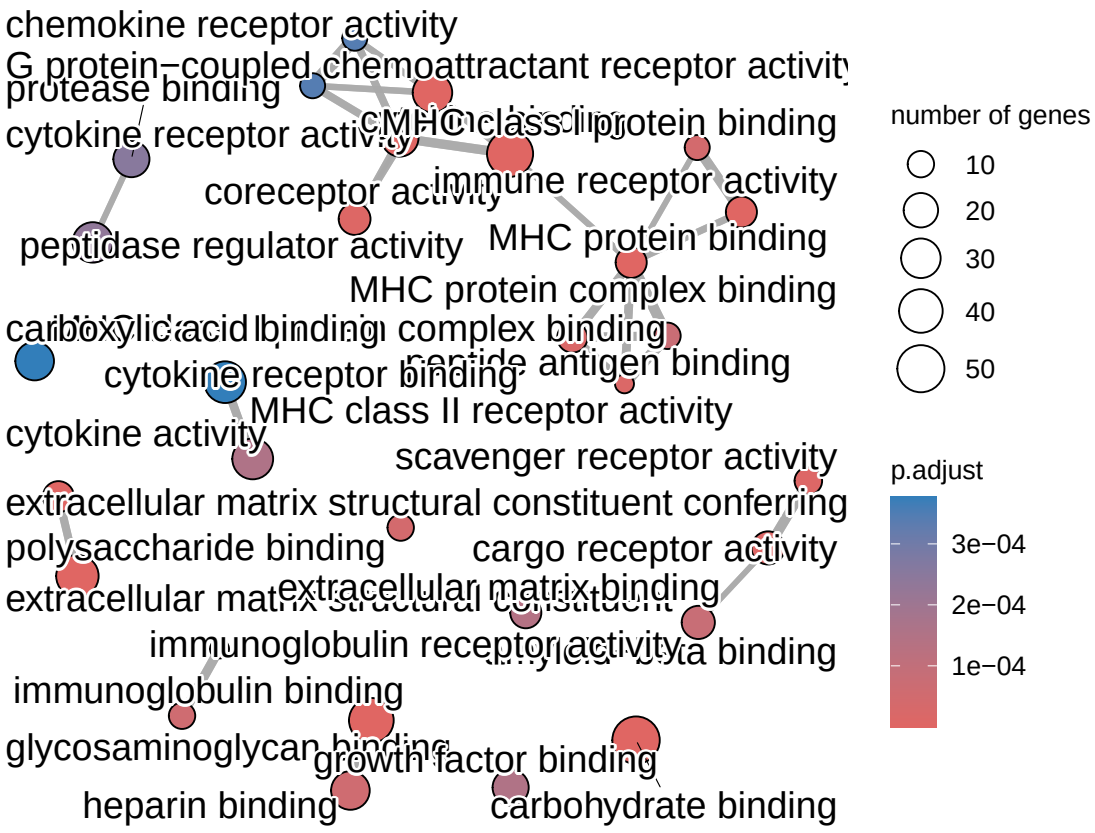



```
print(heatplot(go_mf_up,showCategory = 10))
```



We observe less degree of redundancy at the level of molecular function. This is expected since we're lower in the ontological hierarchy, where specialized functions tend to have less shared genes.

```
#### Visualize with a network view the connection between the top 10 enriched terms
x2 <- pairwise_termsim(go_mf_up)
emapplot(x2)
```



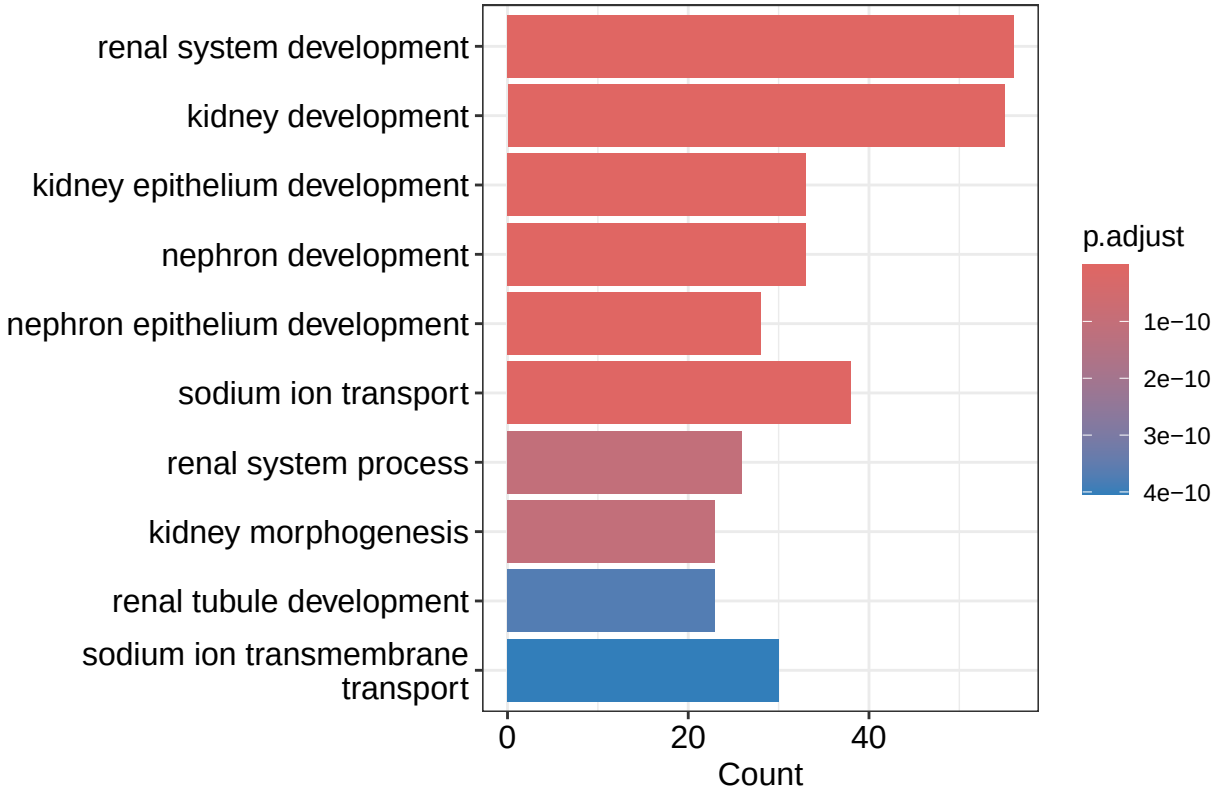
Less redundancy in terms also leads to a sparser graph in terms of connections and smaller neighborhoods. Indeed, many small clusters can be observed.

We now focus on downregulated genes.

```
# similarly for downregulated genes
go_bp_down <- enrichGO(gene = down_DEGs$external_gene_name,
                       OrgDb = org.Hs.eg.db,
                       keyType = 'SYMBOL',
                       ont = "BP",
                       pAdjustMethod = "BH",
                       pvalueCutoff = 0.05,
                       qvalueCutoff = 0.05)
glue("Number of terms satisfying adjusted p. value cutoff: {dim(go_bp_down)[1]}")
```

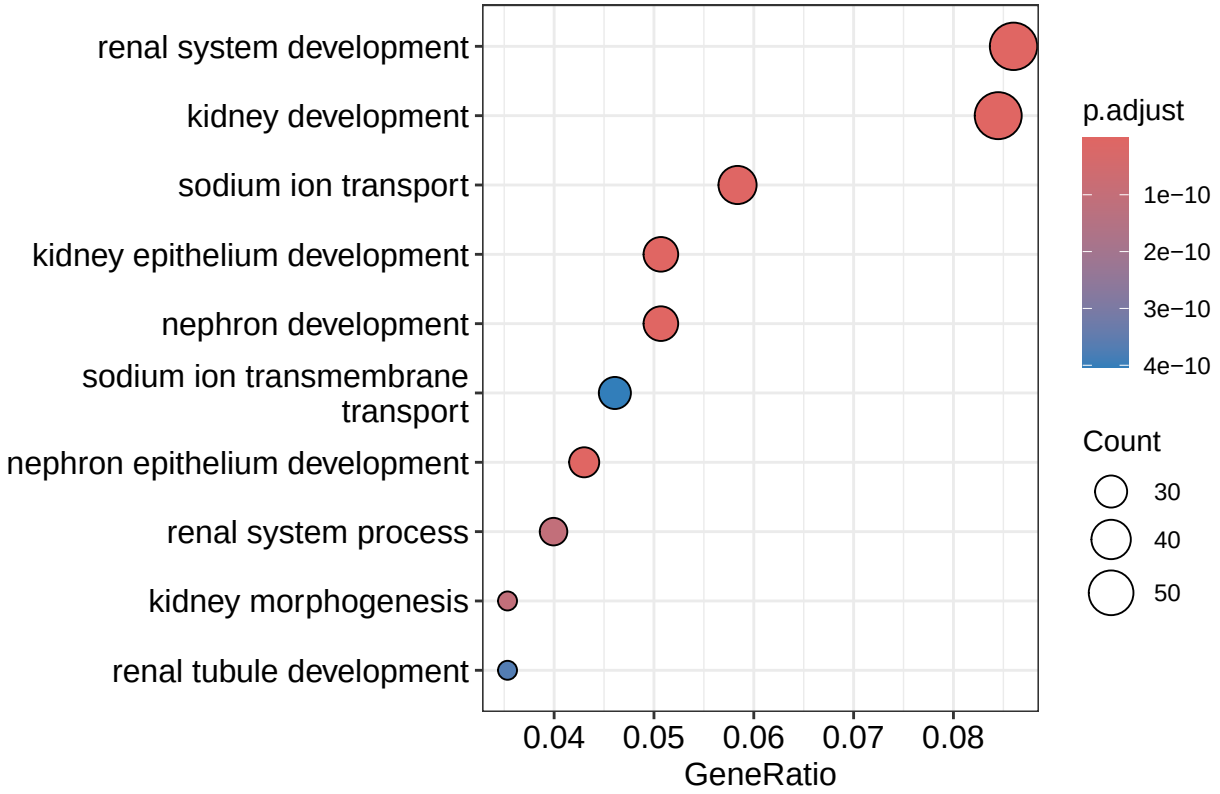
```
## Number of terms satisfying adjusted p. value cutoff: 318
```

```
print(barplot(go_bp_down,showCategory=10))
```



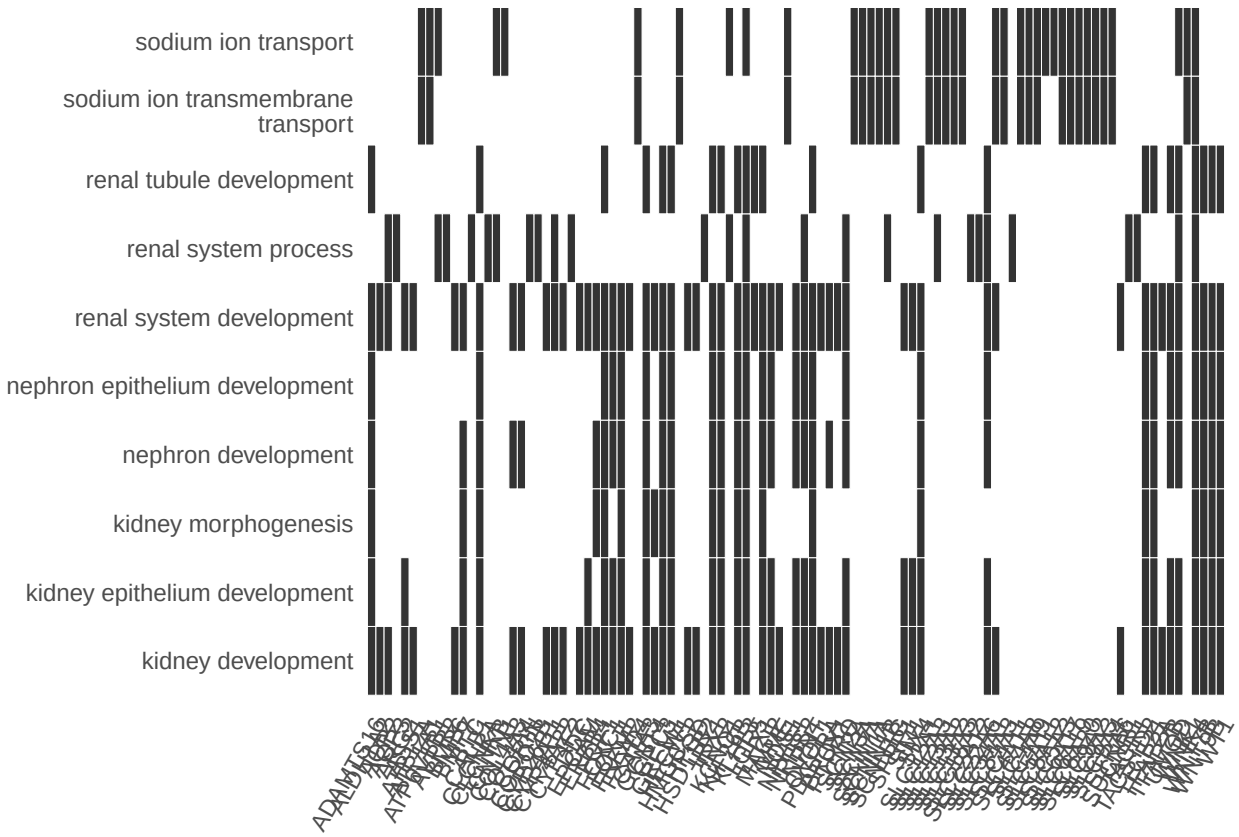
The dominant biological processes that are down-regulated in the case are involved in development and transport.

```
print(dotplot(go_bp_down, showCategory=10))
```



A couple of terms show significantly larger gene ratios (fraction of down DEGs falling in the corresponding biological process). These are involved in the genetic program of kidney development. Although samples were not extracted in developing organisms, developmental pathways are extensively used by cancer cells to increase in number. Almost 10% of the down-regulated DEGs are associated with development. Even though these are indicated as "down-regulated", careful analysis should be made before making assumptions on the direction. For example, we also observe development-associated processes in the up-regulated biological processes (e.g. cell proliferation). This could be interpreted as certain parts of development are down-regulated, while others are up-regulated. In other words, these processes are deregulated or rewired.

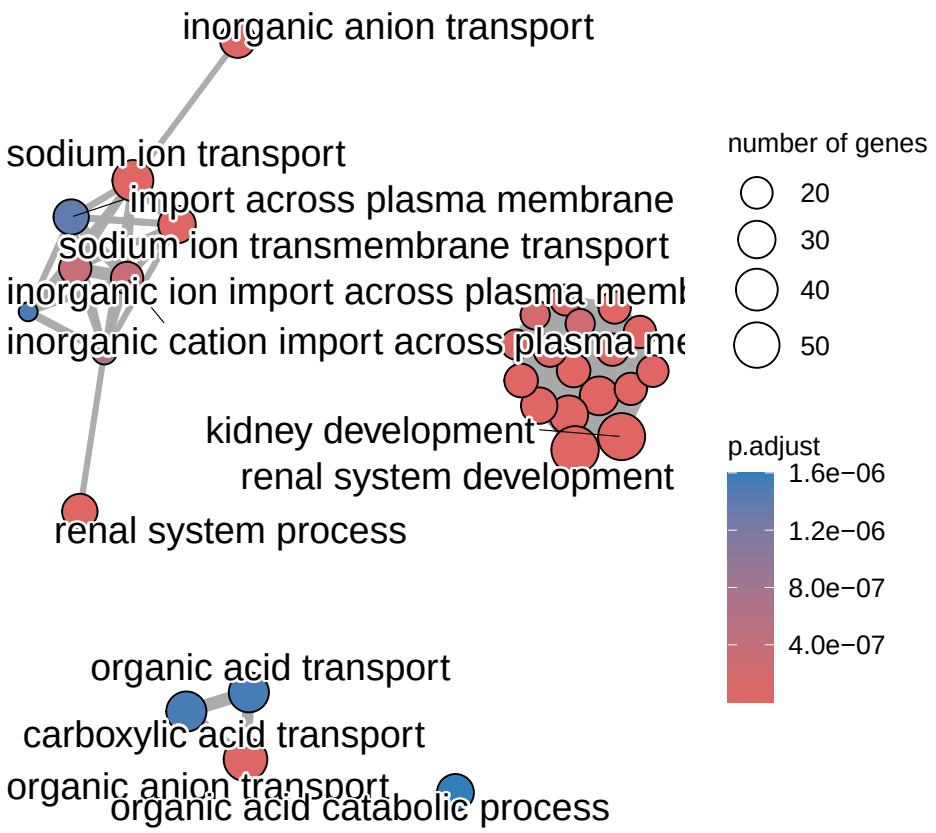
```
print(heatplot(go_bp_down,showCategory = 10))
```



We can see that transport and development processes tend to have separate sets of overlapping genes.

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

```
## Warning: ggrepel: 17 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

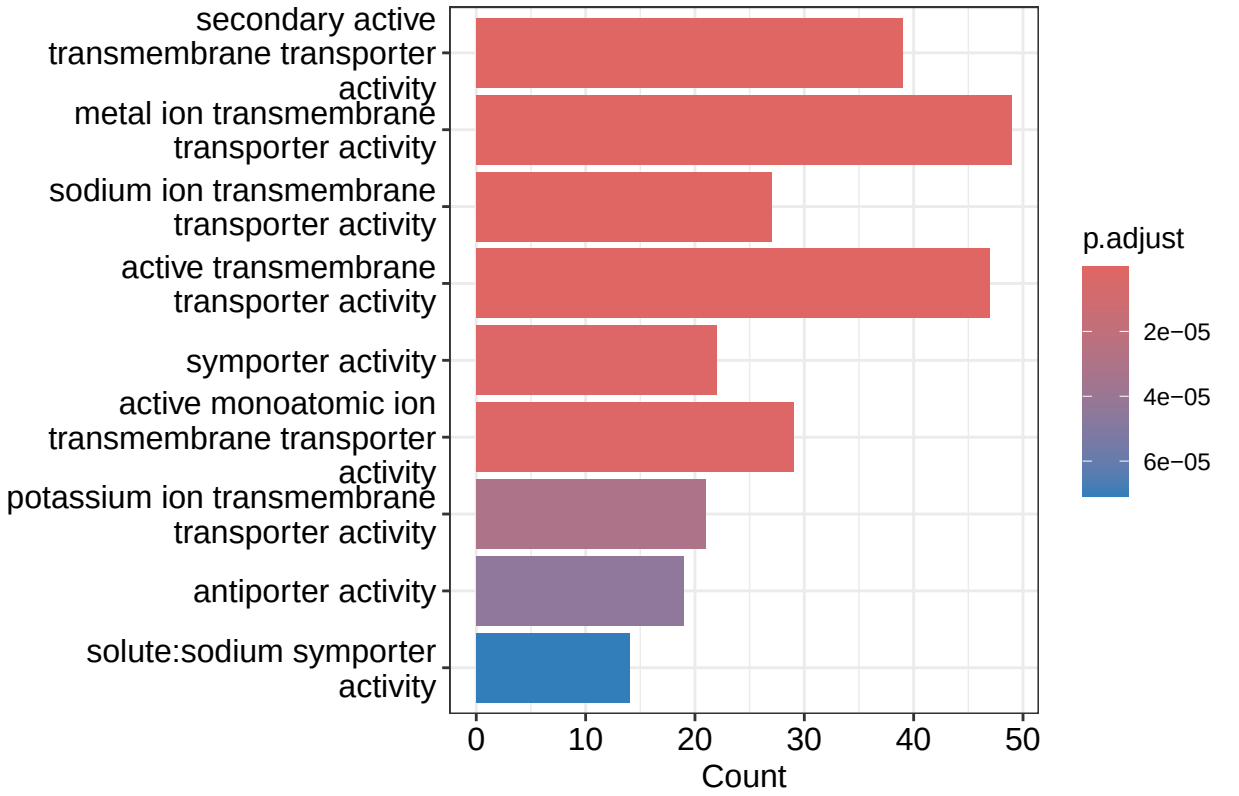


This non-connected graph shows 3 separate clusters. One involves development, and the other 2 involve transport.

```
# GO Molecular Function (MF) for down-regulated genes
go_mf_down <- enrichGO(gene = down_DEGs$external_gene_name,
                       OrgDb = org.Hs.eg.db,
                       keyType = 'SYMBOL',
                       ont = "MF",
                       pAdjustMethod = "BH",
                       pvalueCutoff = 0.05,
                       qvalueCutoff = 0.05)

glue("Number of terms satisfying adjusted p. value cutoff: {dim(go_bp_down)[1]}")
```

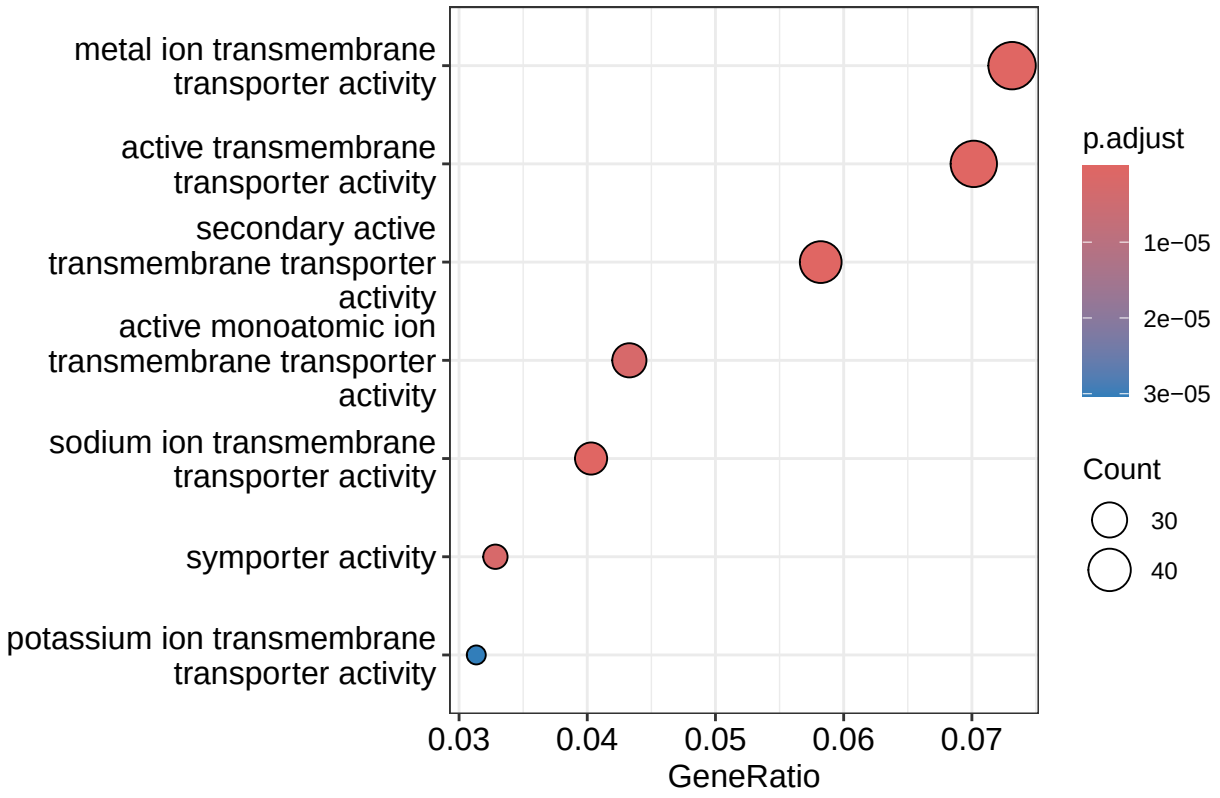
```
## Number of terms satisfying adjusted p. value cutoff: 318
```

```
print(barplot(go_mf_down,showCategory=9))
```

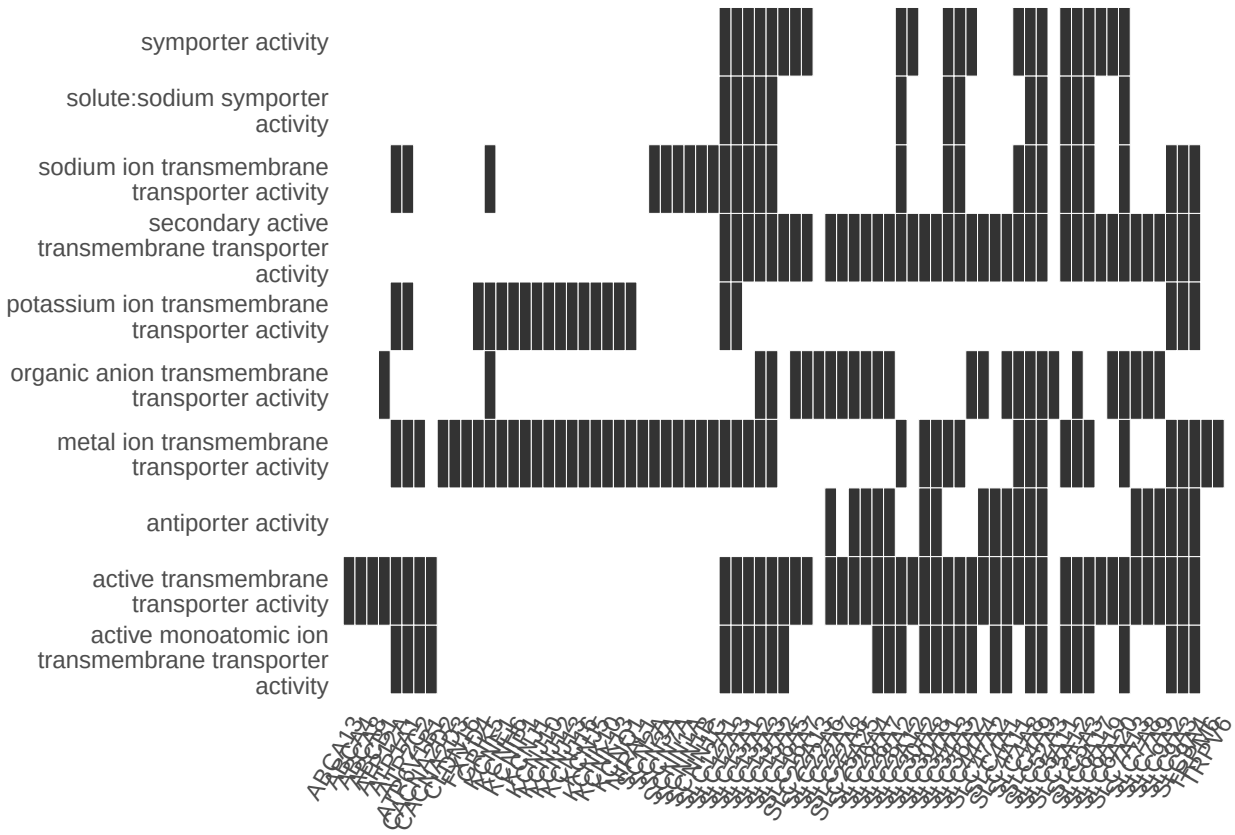


While top 10 down-regulated biological processes (according to adjusted p-value) are mostly concerned with development, the top downregulated molecular functions deal mostly with transport. To which extent these molecular functions are related to developmental processes was not investigated here.

```
print(dotplot(go_mf_down, showCategory=7))
```

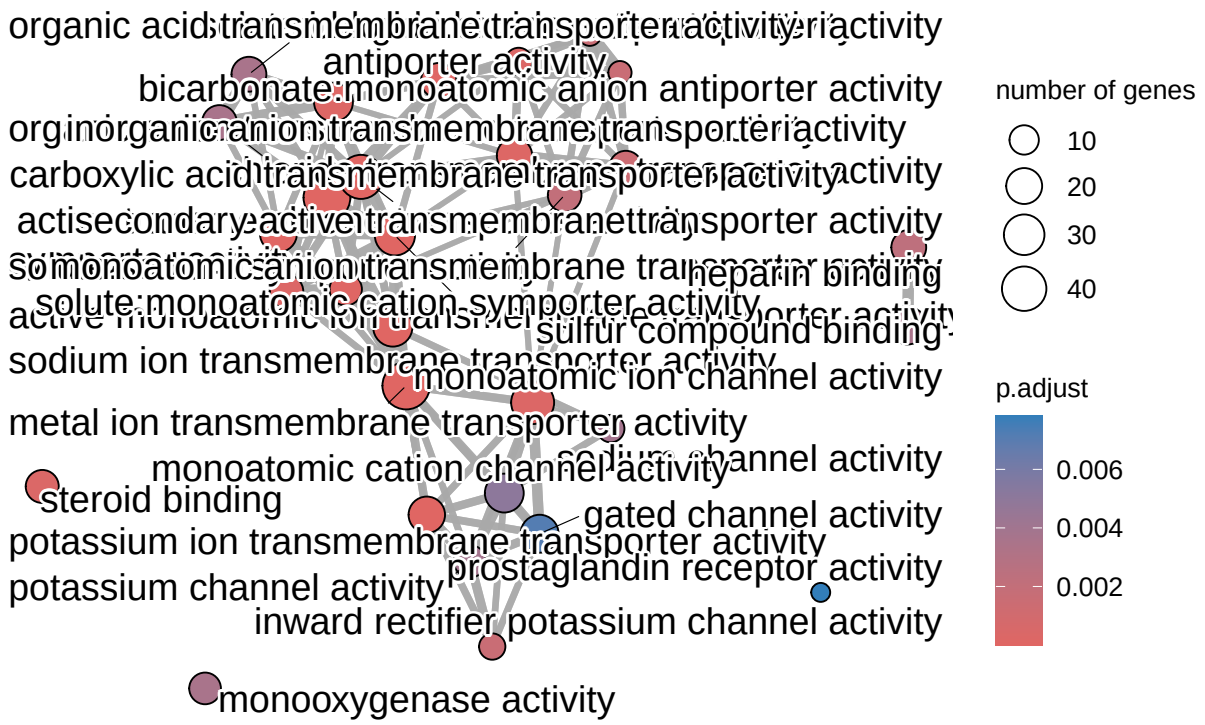



```
print(heatplot(go_mf_down,showCategory = 10))
```



Although there is substantial overlapping of genes among transport molecular functions, we also observe some degree of heterogeneity in gene modules that are used.

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



While the visualization is not optimal, we can still observe what seems to be 3 relatively large clusters, representing the heterogeneity in the transport and binding processes.

Wikipathways analysis

```
# WikiPathways enrichment for up-regulated genes
wp_up <- 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"...
```

```
# hsa04640 corresponds to organismal systems; its the gene set with highest p value

# WikiPathways enrichment for up-regulated genes
wp_down <- enrichKEGG(gene = down_DEGs$entrezgene_id,
                    organism = 'human',
                    pvalueCutoff = 0.05,
                    qvalueCutoff = 0.1)
```

Tabular format: gene ontology enrichment results

Top 10 GO BP terms for up-regulated genes

```
head(go_bp_up, 10)
```

```
##                    ID                                        Description
## GO:0050863 GO:0050863                      regulation of T cell activation
## GO:0007159 GO:0007159                         leukocyte cell-cell adhesion
## GO:0022407 GO:0022407                    regulation of cell-cell adhesion
## GO:1903037 GO:1903037          regulation of leukocyte cell-cell adhesion
## GO:0002443 GO:0002443                          leukocyte mediated immunity
## GO:0070661 GO:0070661                              leukocyte proliferation
## GO:0050867 GO:0050867              positive regulation of cell activation
## GO:0050870 GO:0050870            positive regulation of T cell activation
## GO:0046651 GO:0046651                              lymphocyte proliferation
## GO:1903039 GO:1903039 positive regulation of leukocyte cell-cell adhesion
##            GeneRatio   BgRatio       pvalue     p.adjust       qvalue
## GO:0050863   87/1063 382/18888 4.992721e-30 2.530311e-26 1.942957e-26
## GO:0007159   91/1063 419/18888 1.045732e-29 2.649884e-26 2.034774e-26
## GO:0022407   98/1063 496/18888 1.790041e-28 3.023975e-25 2.322028e-25
## GO:1903037   84/1063 382/18888 7.057392e-28 8.941715e-25 6.866099e-25
## GO:0002443   93/1063 469/18888 3.554669e-27 3.603012e-24 2.766655e-24
## GO:0070661   79/1063 352/18888 6.847621e-27 5.783957e-24 4.441343e-24
## GO:0050867   82/1063 380/18888 1.212154e-26 8.554258e-24 6.568582e-24
## GO:0050870   66/1063 253/18888 1.350317e-26 8.554258e-24 6.568582e-24
## GO:0046651   73/1063 308/18888 1.768021e-26 9.436564e-24 7.246080e-24
## GO:1903039   69/1063 277/18888 1.861990e-26 9.436564e-24 7.246080e-24
##
## GO:0050863                                                                                                CD4/CD6/CD74/RUNX3/TNF
## GO:0007159                                                                                    ITGAL/CD4/ALOX5/CD6/CD74/RUNX3/FOXP3/PAG1/PTPRC/LA
## GO:0022407 CD4/ALOX5/CD6/CD74/RUNX3/FOXP3/SPI1/FSTL3/PAG1/PTPRC/FXYD5/LAG3/IL12RB1/LGALS1/SIRPB1/TNF
## GO:1903037                                                                                                      CD4/ALO
## GO:0002443                                                                                                BTK/TYROBP/WAS/C
## GO:0070661
## GO:0050867
## GO:0050870
## GO:0046651
## GO:1903039
##            Count
## GO:0050863    87
## GO:0007159    91
## GO:0022407    98
## GO:1903037    84
## GO:0002443    93
## GO:0070661    79
## GO:0050867    82
## GO:0050870    66
## GO:0046651    73
## GO:1903039    69
```

Top 10 GO MF terms for up-regulated genes

```r
head(go_mf_up, 10)
```

```
##                    ID
## GO:0140375 GO:0140375
## GO:0030246 GO:0030246
## GO:0005201 GO:0005201
## GO:0005539 GO:0005539
## GO:0004896 GO:0004896
## GO:0023023 GO:0023023
## GO:0019955 GO:0019955
## GO:0042287 GO:0042287
## GO:0030020 GO:0030020
## GO:0015026 GO:0015026
##                                                                  Description
## GO:0140375                                                immune receptor activity
## GO:0030246                                                   carbohydrate binding
## GO:0005201                              extracellular matrix structural constituent
## GO:0005539                                            glycosaminoglycan binding
## GO:0004896                                           cytokine receptor activity
## GO:0023023                                          MHC protein complex binding
## GO:0019955                                                  cytokine binding
## GO:0042287                                                MHC protein binding
## GO:0030020 extracellular matrix structural constituent conferring tensile strength
## GO:0015026                                                coreceptor activity
##            GeneRatio   BgRatio       pvalue     p.adjust       qvalue
## GO:0140375   47/1090 151/18522 5.066135e-22 4.721638e-19 3.978249e-19
## GO:0030246   51/1090 279/18522 3.565947e-13 1.661731e-10 1.400103e-10
## GO:0005201   37/1090 166/18522 1.407187e-12 4.371662e-10 3.683374e-10
## GO:0005539   43/1090 239/18522 4.373603e-11 1.019050e-08 8.586074e-09
## GO:0004896   25/1090  96/18522 1.789623e-10 3.335857e-08 2.810650e-08
## GO:0023023   15/1090  37/18522 8.791427e-10 1.365602e-07 1.150597e-07
## GO:0019955   30/1090 145/18522 1.247750e-09 1.661289e-07 1.399731e-07
## GO:0042287   15/1090  42/18522 7.010859e-09 8.167650e-07 6.881711e-07
## GO:0030020   15/1090  45/18522 2.073404e-08 2.147125e-06 1.809075e-06
## GO:0015026   16/1090  57/18522 1.087799e-07 1.013829e-05 8.542088e-06
##
## GO:0140375                  CD4/CD74/FCGR2B/IL12RB2/IL12RB1/IL2RB/IL21R/LILRB1/EBI3/IL10RA/CCR6/CSF3
## GO:0030246 VCAN/ITIH1/SLC2A3/CHI3L2/PFKP/CLEC2D/SIGLEC1/LGALS1/PYGL/SIGLEC8/CD33/NPTX2/PLOD3/AMBP/VT
## GO:0005201
## GO:0005539                                                                TENM1/STAB1/VCAN/COL
## GO:0004896
## GO:0023023
## GO:0019955
## GO:0042287
## GO:0030020
## GO:0015026
##            Count
## GO:0140375    47
## GO:0030246    51
## GO:0005201    37
## GO:0005539    43
## GO:0004896    25
## GO:0023023    15
```

```
## GO:0019955    30
## GO:0042287    15
## GO:0030020    15
## GO:0015026    16
```

Top 10 GO BP terms for down-regulated genes

```
head(go_bp_down, 10)
```

```
##                    ID                         Description GeneRatio   BgRatio
## GO:0072001 GO:0072001           renal system development    56/651 329/18888
## GO:0001822 GO:0001822                 kidney development    55/651 319/18888
## GO:0072073 GO:0072073      kidney epithelium development    33/651 152/18888
## GO:0072006 GO:0072006                nephron development    33/651 158/18888
## GO:0072009 GO:0072009     nephron epithelium development    28/651 121/18888
## GO:0006814 GO:0006814               sodium ion transport    38/651 244/18888
## GO:0003014 GO:0003014               renal system process    26/651 127/18888
## GO:0060993 GO:0060993               kidney morphogenesis    23/651  98/18888
## GO:0061326 GO:0061326           renal tubule development    23/651 104/18888
## GO:0035725 GO:0035725 sodium ion transmembrane transport    30/651 182/18888
##                  pvalue     p.adjust       qvalue
## GO:0072001 1.840270e-23 5.215070e-20 4.421241e-20
## GO:0001822 2.445519e-23 5.215070e-20 4.421241e-20
## GO:0072073 1.405562e-17 1.998241e-14 1.694072e-14
## GO:0072006 4.842476e-17 5.163290e-14 4.377343e-14
## GO:0072009 7.096224e-16 6.053079e-13 5.131691e-13
## GO:0006814 5.643228e-15 4.011395e-12 3.400787e-12
## GO:0003014 1.723788e-13 1.050279e-10 8.904078e-11
## GO:0060993 2.026592e-13 1.080427e-10 9.159664e-11
## GO:0061326 7.801780e-13 3.697177e-10 3.134399e-10
## GO:0035725 9.471972e-13 4.039796e-10 3.424866e-10
##
## GO:0072001 CYP26B1/TFAP2B/FOXC1/COL4A4/ARG2/SIX4/BMP7/CALB1/SFRP1/GATA3/SIM1/FGF1/NPHS2/PROX1/TCF21/
## GO:0001822      CYP26B1/TFAP2B/FOXC1/COL4A4/ARG2/SIX4/BMP7/CALB1/SFRP1/GATA3/SIM1/FGF1/NPHS2/PROX1/T
## GO:0072073
## GO:0072006
## GO:0072009
## GO:0006814                                                                                HECW1/SLC13A2/CNT
## GO:0003014
## GO:0060993
## GO:0061326
## GO:0035725
##            Count
## GO:0072001    56
## GO:0001822    55
## GO:0072073    33
## GO:0072006    33
## GO:0072009    28
## GO:0006814    38
## GO:0003014    26
## GO:0060993    23
## GO:0061326    23
## GO:0035725    30
```

Tabular format: gene ontology enrichment results

Top 10 GO MF terms for down-regulated genes

```
head(go_mf_down, 10)
```

```
##                    ID                                          Description
## GO:0015291 GO:0015291      secondary active transmembrane transporter activity
## GO:0046873 GO:0046873            metal ion transmembrane transporter activity
## GO:0015081 GO:0015081           sodium ion transmembrane transporter activity
## GO:0022804 GO:0022804               active transmembrane transporter activity
## GO:0015293 GO:0015293                                    symporter activity
## GO:0022853 GO:0022853 active monoatomic ion transmembrane transporter activity
## GO:0015079 GO:0015079        potassium ion transmembrane transporter activity
## GO:0015297 GO:0015297                                   antiporter activity
## GO:0015370 GO:0015370                     solute:sodium symporter activity
## GO:0008514 GO:0008514        organic anion transmembrane transporter activity
##            GeneRatio   BgRatio       pvalue     p.adjust       qvalue
## GO:0015291    39/670 285/18522 8.339034e-13 6.554481e-10 5.556430e-10
## GO:0046873    49/670 437/18522 2.025880e-12 7.961710e-10 6.749380e-10
## GO:0015081    27/670 157/18522 1.413102e-11 3.702327e-09 3.138573e-09
## GO:0022804    47/670 454/18522 9.382250e-11 1.843612e-08 1.562885e-08
## GO:0015293    22/670 147/18522 1.595979e-08 2.508879e-06 2.126852e-06
## GO:0022853    29/670 244/18522 1.934691e-08 2.534446e-06 2.148526e-06
## GO:0015079    21/670 158/18522 2.697702e-07 3.029134e-05 2.567888e-05
## GO:0015297    19/670 136/18522 4.554103e-07 4.474406e-05 3.793088e-05
## GO:0015370    14/670  79/18522 8.109625e-07 7.082406e-05 6.003968e-05
## GO:0008514    26/670 258/18522 2.566690e-06 2.002441e-04 1.697529e-04
##
## GO:0015291                                                  SLC4A1/SLC13A2/SLC7A9/SLC4A8/SLC9A3
## GO:0046873           SLC13A2/CACNA2D2/SLC4A8/KCNQ1/ATP2C2/SLC9A3/SLC12A3/SLC12A1/ATP12A/SLC13A1/SLC4A
## GO:0015081
## GO:0022804 SLC4A1/SLC13A2/SLC7A9/SLC4A8/ATP2C2/SLC9A3/SLC12A3/SLC12A1/ATP12A/SLC13A1/ABCB1/SLC4A11/SL
## GO:0015293
## GO:0022853
## GO:0015079
## GO:0015297
## GO:0015370
## GO:0008514
##            Count
## GO:0015291    39
## GO:0046873    49
## GO:0015081    27
## GO:0022804    47
## GO:0015293    22
## GO:0022853    29
## GO:0015079    21
## GO:0015297    19
## GO:0015370    14
## GO:0008514    26
```

Top 10 WikiPathways for up-regulated genes

```
head(wp_up, 10)
```

```
##                                     category
## hsa04640                  Organismal Systems
## hsa04060 Environmental Information Processing
## hsa04145                  Cellular Processes
## hsa04610                  Organismal Systems
## hsa05150                      Human Diseases
## hsa04514 Environmental Information Processing
## hsa04061 Environmental Information Processing
## hsa04658                  Organismal Systems
## hsa05330                      Human Diseases
## hsa04940                      Human Diseases
##                                 subcategory       ID
## hsa04640                       Immune system hsa04640
## hsa04060 Signaling molecules and interaction hsa04060
## hsa04145           Transport and catabolism hsa04145
## hsa04610                       Immune system hsa04610
## hsa05150      Infectious disease: bacterial hsa05150
## hsa04514 Signaling molecules and interaction hsa04514
## hsa04061 Signaling molecules and interaction hsa04061
## hsa04658                       Immune system hsa04658
## hsa05330                     Immune disease hsa05330
## hsa04940    Endocrine and metabolic disease hsa04940
##                                                                  Description
## hsa04640                                              Hematopoietic cell lineage
## hsa04060                               Cytokine-cytokine receptor interaction
## hsa04145                                                           Phagosome
## hsa04610                                  Complement and coagulation cascades
## hsa05150                                      Staphylococcus aureus infection
## hsa04514                                            Cell adhesion molecules
## hsa04061 Viral protein interaction with cytokine and cytokine receptor
## hsa04658                                     Th1 and Th2 cell differentiation
## hsa05330                                                 Allograft rejection
## hsa04940                                          Type I diabetes mellitus
##          GeneRatio  BgRatio       pvalue     p.adjust       qvalue
## hsa04640    31/617  99/8844 3.423940e-13 1.064845e-10 8.758079e-11
## hsa04060    56/617 298/8844 4.229514e-12 5.940673e-10 4.886050e-10
## hsa04145    38/617 157/8844 5.730553e-12 5.940673e-10 4.886050e-10
## hsa04610    27/617  88/8844 2.008396e-11 1.561528e-09 1.284316e-09
## hsa05150    27/617 100/8844 5.152265e-10 3.204709e-08 2.635790e-08
## hsa04514    34/617 157/8844 1.839326e-09 9.533841e-08 7.841338e-08
## hsa04061    26/617 100/8844 2.625064e-09 1.166278e-07 9.592338e-08
## hsa04658    24/617  92/8844 1.005467e-08 3.908752e-07 3.214848e-07
## hsa05330    15/617  38/8844 1.306866e-08 4.515949e-07 3.714252e-07
## hsa04940    15/617  43/8844 9.214383e-08 2.865673e-06 2.356942e-06
##
## hsa04640
## hsa04060 9235/920/3604/84957/4804/3595/3594/3560/958/10673/50615/10148/944/3587/1235/6364/1441/7852/3
## hsa04145
## hsa04610
## hsa05150
## hsa04514
```

```
## hsa04061
## hsa04658
## hsa05330
## hsa04940
##          Count
## hsa04640    31
## hsa04060    56
## hsa04145    38
## hsa04610    27
## hsa05150    27
## hsa04514    34
## hsa04061    26
## hsa04658    24
## hsa05330    15
## hsa04940    15
```

Top 10 WikiPathways for down-regulated genes

```
head(wp_down, 10)
```

```
##                                    category
## hsa04974                  Organismal Systems
## hsa04978                  Organismal Systems
## hsa04960                  Organismal Systems
## hsa00330                         Metabolism
## hsa04928                  Organismal Systems
## hsa04514 Environmental Information Processing
## hsa00280                         Metabolism
## hsa03320                  Organismal Systems
## hsa04971                  Organismal Systems
## hsa04530                  Cellular Processes
##                                 subcategory       ID
## hsa04974                   Digestive system hsa04974
## hsa04978                   Digestive system hsa04978
## hsa04960                   Excretory system hsa04960
## hsa00330               Amino acid metabolism hsa00330
## hsa04928                   Endocrine system hsa04928
## hsa04514 Signaling molecules and interaction hsa04514
## hsa00280               Amino acid metabolism hsa00280
## hsa03320                   Endocrine system hsa03320
## hsa04971                   Digestive system hsa04971
## hsa04530     Cellular community - eukaryotes hsa04530
##                                                 Description GeneRatio  BgRatio
## hsa04974                   Protein digestion and absorption    16/339 105/8844
## hsa04978                                Mineral absorption    11/339  61/8844
## hsa04960          Aldosterone-regulated sodium reabsorption     8/339  38/8844
## hsa00330                     Arginine and proline metabolism     9/339  50/8844
## hsa04928 Parathyroid hormone synthesis, secretion and action    14/339 115/8844
## hsa04514                           Cell adhesion molecules    16/339 157/8844
## hsa00280         Valine, leucine and isoleucine degradation     8/339  48/8844
## hsa03320                            PPAR signaling pathway    10/339  76/8844
## hsa04971                            Gastric acid secretion    10/339  76/8844
## hsa04530                                   Tight junction    16/339 170/8844
```

```
##                pvalue     p.adjust       qvalue
## hsa04974 2.136856e-06 0.0006047303 0.0005420867
## hsa04978 1.670649e-05 0.0023639685 0.0021190865
## hsa04960 7.657130e-05 0.0068092755 0.0061039070
## hsa00330 1.001511e-04 0.0068092755 0.0061039070
## hsa04928 1.203052e-04 0.0068092755 0.0061039070
## hsa04514 3.295053e-04 0.0155416650 0.0139317138
## hsa00280 4.229935e-04 0.0171010235 0.0153295394
## hsa03320 6.017420e-04 0.0189214429 0.0169613827
## hsa04971 6.017420e-04 0.0189214429 0.0169613827
## hsa04530 7.976553e-04 0.0225736451 0.0202352556
##                                                                                                                  g
## hsa04974           11136/1298/3784/6550/1286/23428/3769/7512/136227/476/1285/340024/153201/1287/4311/
## hsa04978                                        6550/7421/140803/4495/6569/476/55503/340024/4494/142680/
## hsa04960                                                                6337/53828/4306/3758/476/6340/6338/
## hsa00330                                                             384/4842/1610/79814/219/8659/2628/112817/5
## hsa04928                            1591/846/65125/6559/2625/1594/7421/65266/6569/9365/4929/5745/50964/14
## hsa04514     6401/5010/1272/999/1001/10686/79679/9071/152404/9073/23562/22854/23114/149461/100506658/
## hsa00280                                                                                  30/4329/3158/219/1629/3033/1
## hsa03320                                                          30/5106/5105/3158/79966/2168/23305/8309/2710/
## hsa04971                                                       3784/54207/91807/115111/3758/3773/476/3766/810/38
## hsa04530 5010/54566/27134/10686/84952/4628/9071/91862/57530/153562/9073/5521/23562/149461/79861/10050
##          Count
## hsa04974    16
## hsa04978    11
## hsa04960     8
## hsa00330     9
## hsa04928    14
## hsa04514    16
## hsa00280     8
## hsa03320    10
## hsa04971    10
## hsa04530    16
```

Task 5

Use the pathview R package to visualize one pathway you find enriched using the up-regulated gene list.

In this case I use the first up-regulated pathway: hsa04640 Organismal Systems Immune system

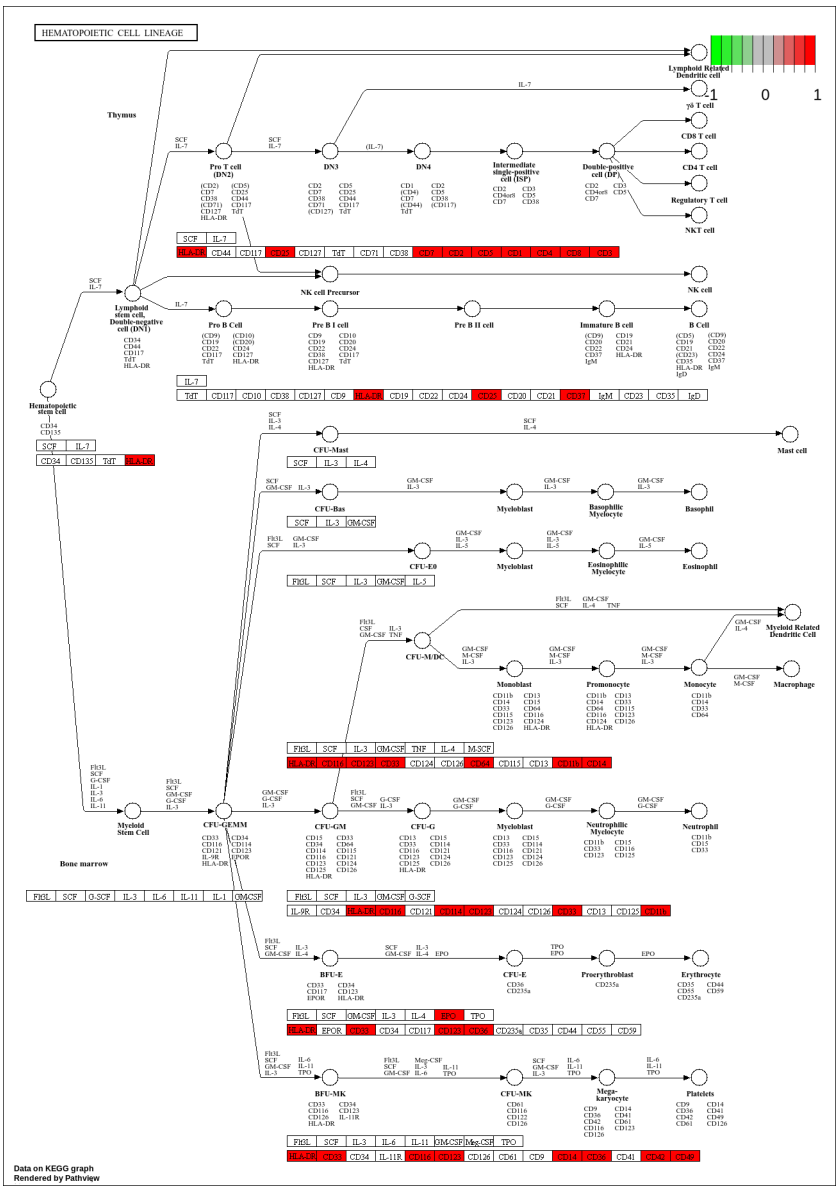
```
### Plot the pathway with DEG genes
logFC <- up_DEGs$logFC
names(logFC) <- up_DEGs$entrezgene_id

# Render the pathway, which saves the image to a file
pv.out <- pathview(gene.data = logFC,
                   pathway.id = "hsa04640",
                   species = "human",
                   kegg.native = TRUE)
```

```
## 'select()' returned 1:1 mapping between keys and columns
```

```
## Info: Working in directory /home/leonardo/Desktop/git/BioinformaticsResourcesProject
```

```
## Info: Writing image file hsa04640.pathview.png
```

```r
# Include the saved image in the R Markdown document
# The path is usually in the current working directory and named based on the pathway ID
image_path <- paste0("hsa04640.pathview.png")
knitr::include_graphics(image_path)
```



This image represents the pathway of the hematopoietic lineage expressed as a directed tree. Nodes are cell types and edges are directed, indicating the lineage relationship across nodes (e.g. A -> B , cell type A gives rise to cell type B). An optional attribute of the edge is the name of the gene (or genes) promoting or regulating in some way the transition. Node attributes may be marker genes of that cell type. Up-regulated genes are viewed in red. Had we used all DEGs we may have potentially found also downregulated genes in the same pathway. While this is named as a pathway, it is different from a biochemical pathway because the latter also shows individual reactions, enzymes, molecular reactants and products. This is a pathway in a higher level sense.

```r
# Clear up memory for next steps
rm(list = ls())
```

Task 6

Identify which transcription factors (TFs) have enriched scores in the promoters of all up-regulated (or down-regulated if you prefer) genes. a use a window of 500 nucleotides upstream each gene.

WARNING: since bioMaRt's getSequence function does not work in my environment (using R on conda in Ubuntu 22), the biomaRt website interface was raising errors for that particular query, and the UCSC table browser was yielding too many results, I decided to use a different procedure to get the upstream sequences. Quality of downstream results may or may not have been altered by this custom procedure.

```r
### Load results of DEG analysis
DEGs <- read.table("DEGs.txt", header = TRUE, sep = "\t", stringsAsFactors = FALSE)
table(DEGs$class)
```

```
##
##     -     +     =
##   728  1182 14864
```

```r
# Load the up-regulated DEGs
up_DEGs <- read.table("up_DEGs.txt", header = TRUE, sep = "\t", stringsAsFactors = FALSE)

# Load libraries
suppressMessages(library(MotifDb))
suppressMessages(library(seqLogo))
suppressMessages(library(PWMEnrich))
suppressMessages(library(PWMEnrich.Hsapiens.background))
suppressMessages(library(BSgenome.Hsapiens.UCSC.hg19))  # Load the genome build you are using
suppressMessages(library(biomaRt))
suppressMessages(library(glue))

# Initialize the Ensembl Biomart
ensembl <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")

# Retrieve 500bp upstream sequences for the up-regulated genes
up_gene_ids <- up_DEGs$ensembl_gene_id

### GetSequence function NOT working on this environment (or other environments on ubuntu 22). Therefor

# Query biomart for gene information including strand and position
promoter_info <- getBM(
  attributes = c('ensembl_gene_id', 'external_gene_name', 'chromosome_name', 'start_position', 'end_pos
  filters = 'ensembl_gene_id',
  values = up_gene_ids,
  mart = ensembl
)

# Ensure chromosome names have the correct format
promoter_info$chromosome_name <- paste0("chr", promoter_info$chromosome_name)
```

```r
# Adjust the coordinates for upstream sequences
promoter_info$upstream_start <- ifelse(
  promoter_info$strand == 1,
  promoter_info$start_position - 500,  # For forward strand: 500 bp upstream
  promoter_info$end_position + 1       # For reverse strand: 1 bp downstream (start counting)
)

promoter_info$upstream_end <- ifelse(
  promoter_info$strand == 1,
  promoter_info$start_position - 1,    # Up to the start of the gene
  promoter_info$end_position + 500     # 500 bp downstream for reverse strand
)

# Ensure upstream_start is not less than 1
promoter_info$upstream_start <- pmax(promoter_info$upstream_start, 1)

# Get chromosome lengths to avoid exceeding them
valid_chromosome_lengths <- seqlengths(BSgenome.Hsapiens.UCSC.hg19)

# Ensure upstream_end does not exceed chromosome lengths
promoter_info$upstream_end <- pmin(promoter_info$upstream_end,
                                   valid_chromosome_lengths[promoter_info$chromosome_name])

# Further check: Ensure start is always <= end
valid_indices <- promoter_info$upstream_start <= promoter_info$upstream_end
promoter_info <- promoter_info[valid_indices, ]

# Extract sequences using mapply
up_promoter_sequences <- mapply(function(chr, start, end) {
  getSeq(BSgenome.Hsapiens.UCSC.hg19, chr, start=start, end=end)
},
promoter_info$chromosome_name, promoter_info$upstream_start, promoter_info$upstream_end)

# Convert to DNAStringSet format
seq_up <- DNAStringSet(up_promoter_sequences)

# Assign the gene names as the names of the sequences
names(seq_up) <- promoter_info$external_gene_name

# Filter out sequences with empty gene names
valid_gene_names <- promoter_info$external_gene_name != ""

# Apply the filter to both seq_up and promoter_info
seq_up_filtered <- seq_up[valid_gene_names]
promoter_info_filtered <- promoter_info[valid_gene_names, ]

# Reassign gene names to the filtered sequences
names(seq_up_filtered) <- promoter_info_filtered$external_gene_name

# Debugging step to confirm that no empty gene names remain
if (any(names(seq_up_filtered) == "")) {
  warning("There are still empty values in the gene names after filtering.")
} else {
```

```
  message("All sequences now have valid gene names.")
}
```

```
## All sequences now have valid gene names.
```

```
# Identify sequences that contain only "N"
is_all_N <- vapply(seq_up_filtered, function(x) {
  freq <- alphabetFrequency(x)
  sum(freq[c("A", "C", "G", "T")]) == 0 && freq["N"] > 0
}, logical(1))

# Remove sequences that are entirely "N"
seq_up_filtered <- seq_up_filtered[!is_all_N]

# Also remove corresponding entries from promoter_info_filtered if needed
promoter_info_filtered <- promoter_info_filtered[!is_all_N, ]

# Subset motifs from MotifDb to human TFs
mdb.human <- subset(MotifDb, organism == 'Hsapiens')

# Convert motifs to PWMs
pwms <- sapply(as.list(mdb.human), toPWM)

# Load background model as seen in class
data(PWMLogn.hg19.MotifDb.Hsap)

# Perform motif enrichment analysis on the 500bp upstream regions
#res = motifEnrichment(seq_up_filtered,PWMLogn.hg19.MotifDb.Hsap,score = "affinity")

# Save the res_up object to a file
#save(res, file = "res_up_results.RData")

#Load pre-computed results (takes >1 hour without parallelization)
load("res_up_results.RData")

report = sequenceReport(res, 1)
# Only top 10 significant TFs to have a better visualization
plot(report[report$p.value < 0.05][1:10], fontsize=7, id.fontsize=6)
```

Rank	Target	PWM	Motif ID	Raw score	P−value
1	UW.Motif.0380		UW.Motif.0380	71.6	0.000575
2	POU2F3		Hsapiens−jolma2013−POU2F3	52.9	0.00201
3	FOXB1		Hsapiens−jolma2013−FOXB1−3	208	0.00211
4	POU2F2		Hsapiens−jolma2013−POU2F2	54.4	0.00221
5	UW.Motif.0653		UW.Motif.0653	37.8	0.0023
6	POU5F1P1		Hsapiens−jolma2013−POU5F1P1	49.9	0.00234
7	POU3F4		Hsapiens−jolma2013−POU3F4	92.1	0.0026
8	UW.Motif.0419		UW.Motif.0419	51.8	0.00305
9	UW.Motif.0630		UW.Motif.0630	31.6	0.00328
10	UW.Motif.0049		UW.Motif.0049	26.4	0.00378

Note: Only the first 10 were shown for clarity

With this procedure, we took the sequences starting 500 upstream the start of genes, likely representing promoter regions, and we computed the score using total binding affinity (sum of maximum scores of the sliding frame) between each of them and each PWM of the human genome registered in MotifDb. The likelihood of observing the obtained score or greater scores by chance is compared with background models and this is reflected in the p value. It appears that the adjusted p.value is not provided with this function. In this case we're more interested in ranking, therefore we omit the adjusted p.value step, but more rigorous analyses would need to consider it. For the next tasks, I chose the motif with greatest p-value with a known gene symbol related to it, in this case Pou2f3.

The letters in the PWM column represent the probability of observing a particular base at a specific position, with a scaling that takes into account the information content. This is called a sequence logo. The larger the letter, the more likely it can be found across the aligned sequences associated to a particular motif, the larger the information content according to the author's definition.

```
significant_motives <- sum(report$p.value < 0.05)
glue("Motives with p value less than 0.05: {significant_motives} ")
```

```
## Motives with p value less than 0.05: 126
```

Task 7

Select one among the top enriched TFs, compute the empirical distributions of scores for all PWMs that you find in MotifDB for the selected TF and determine for all of them the distribution (log2) threshold cutoff at 99.75% (relax the threshold if needed).

```r
# To load it later:
mdb.human = subset(MotifDb, organism=='Hsapiens' & geneSymbol=="POU2F3")
PWM = toPWM(as.list(mdb.human))
names(PWM) = sapply(names(PWM), function(x) strsplit(x,"-")[[1]][3])
names(PWM) <- make.unique(names(PWM))
#compute empirical distribution of scores for all associated PWMs, using "random" or "independent" geno
ecdf = motifEcdf(PWM,organism = "hg19",quick=TRUE)
```

```
## Starting scanning with motifs from 1 to 4

## Scanning all sequences with motif 1 / 4

## Scanning all sequences with motif 2 / 4

## Scanning all sequences with motif 3 / 4

## Scanning all sequences with motif 4 / 4
```

```r
threshold <- log2(quantile(ecdf$'POU2F3',0.9975))
threshold
```

```
##   99.75%
## 5.017342
```

In this step, we first query all POU2F3-associated PWMs in the human genome. We then compute an empirical distribution of scores obtained from random sequences in the human genome when the score is computed using the PWMs. This is useful to measure a background probability of observing a score under the null hypothesis assumption which states the independence between a query sequence and a given motif which is represented by the PWM. In this case we're only focusing on the POU3F3 motifs, which have been found enriched among the upstream sequences of the up-regulated genes. Once the distribution of scores from random sequences is generated, we select a threshold based on the quantiles and log scale it. In this case the threshold is 5.017342 at the 99.75th quantile, which is the value that is larger than 99.75% of observations

Task 8

Identify which up-regulated (or down-regulated depending on the choice you made at point 7) genes have a region in their promoter (defined as previously) with binding scores above the computed thresholds for any of the previously selected PWMs. Use pattern matching as done during the course;

```r
# For each upstream sequence, compute the score
scores = suppressMessages(motifScores(seq_up_filtered,PWM,raw.score=TRUE))
```

```r
# Compare log-transformed scores to the log2 threshold
genes_above_threshold <- vector()  # Initialize an empty vector to store gene names

for (i in seq_along(scores)) {
  # Extract the log-transformed scores for POU2F3
  pou2f3_scores <- scores[[i]][, 'POU2F3']
```

```
  # Check if any of these log scores exceed the log2 threshold
  if (any(pou2f3_scores > threshold, na.rm = TRUE)) {
    genes_above_threshold <- c(genes_above_threshold, names(scores)[i])
  }
}

# Display the list of gene names that exceed the threshold
print(head(genes_above_threshold))
```

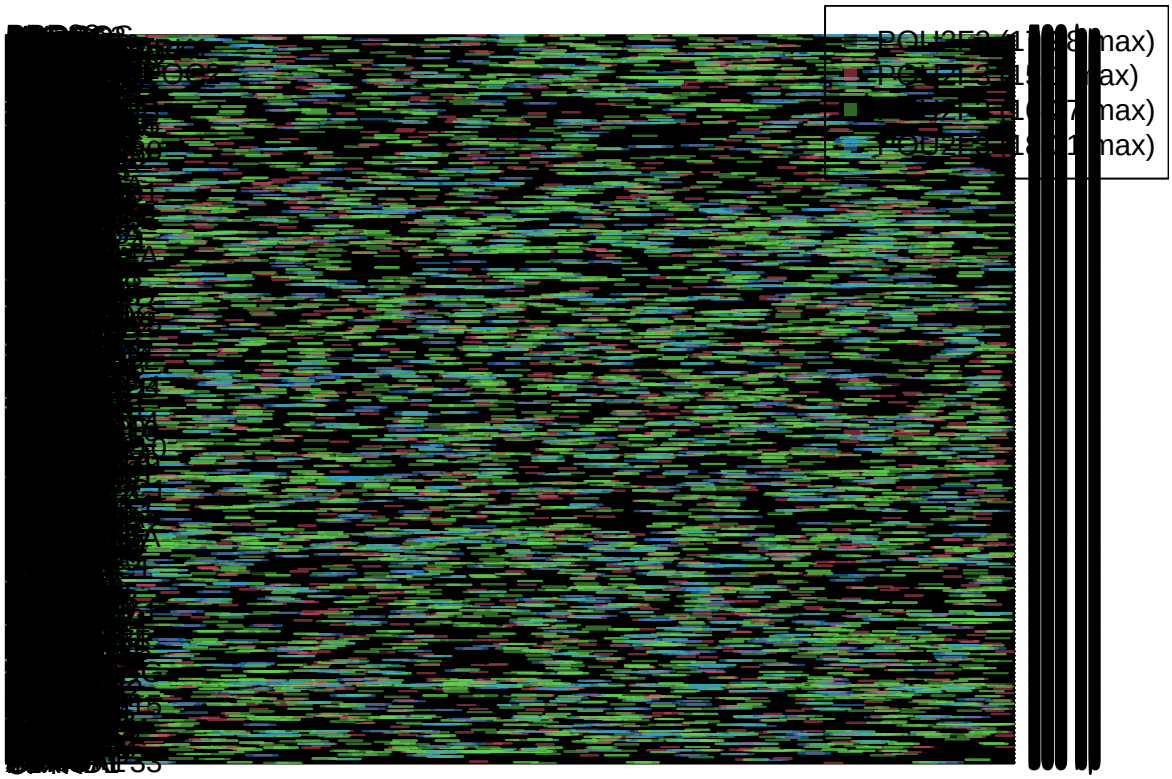
```
## [1] "CFTR"     "PLXND1"   "CAMKK1"   "ARHGAP33" "MCUB"     "PON1"
```

```
# Display the number of genes that exceed the threshold
print(length(genes_above_threshold))
```

```
## [1] 1074
```

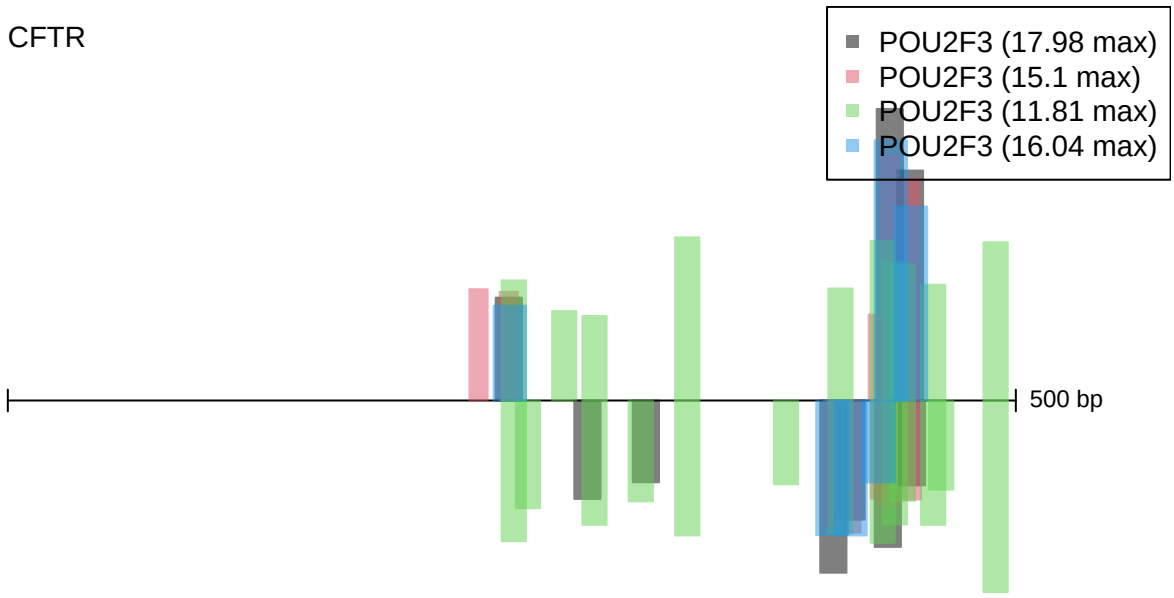
As we can see, 1074 out of 1182 genes associated to the upstream sequences have been found significantly enriched with the POU2F3 motif. It seems unlikely that a single motif is present in so many promoter sequences. Here, only the first 6 are printed out. Possible reasons: 1) The procedure is actually correct and the list contains mostly real positives 2) The way the upstream sequences were obtained is incorrect 3) The random model of scores is not appropriate 4) The threshold used is too small for this particular query. This might be the case since given that our up regulated DEGs are very redundant and connected, they might be overall a lot more similar to that pattern compared to random sequences.

```
plotMotifScores(scores,motif.names = "POU2F3",legend.space = 0.1,cutoff=threshold)
```



Indeed, too much signal results in a bad motif score plot. Focusing on one particular gene:

```
plotMotifScores(scores[1],motif.names = "POU2F3",legend.space = 0.1,cutoff=threshold)
```



Here we are observing the upstream sequence of the gene CFTR. The locations where colored bars appear correspond to regions where the score was exceeding the threshold, which suggest possible binding sites of POU2F3. The colors are associated with the database from which the PWM was obtained (4 in total).

```
# Clear up memory for next steps
rm(list = ls())
```

Task 9

Use STRING database to find PPI interactions among differentially expressed genes and export the network in TSV format.

```
# Performed with STRING's web interface
```

```
# Clear up memory for final step
rm(list = ls())
```

Task 10

Import the network in R and using igraph package identify and plot the largest connected component.

```r
suppressMessages(library(clusterProfiler))
suppressMessages(library(org.Hs.eg.db))
suppressMessages(library(enrichplot))
suppressMessages(library(ggnewscale))
suppressMessages(library(igraph))
options(ggrepel.max.overlaps = Inf)

# Read the network file
links <- read.delim("string_interactions.tsv")

#write.table(up_DEGs$external_gene_name, file = "up_DEGs_list.txt", row.names = FALSE, col.names = FALS
up_DEGs <- read.table("up_DEGs.txt", header = TRUE, sep = "\t", stringsAsFactors = FALSE)

## Create nodes annotations using biomaRt
ensembl <- useMart(biomart="ensembl",dataset="hsapiens_gene_ensembl")
nodes <-
  getBM(attributes=c("external_gene_name","ensembl_gene_id","description","gene_biotype","start_positio
        filters=c("ensembl_gene_id"),
        values=up_DEGs[,1],
        mart = ensembl)

nodes <- nodes[!duplicated(nodes$external_gene_name), ]
any(duplicated(nodes$external_gene_name))  # Should return FALSE
```

```
## [1] FALSE
```

```r
# Extract unique gene names from links dataframe
unique_genes_in_links <- unique(c(links$X.node1, links$node2))

# Extract the gene names from the nodes dataframe
genes_in_nodes <- unique(nodes$external_gene_name)

# Identify genes present in links but not in nodes
missing_genes <- setdiff(unique_genes_in_links, genes_in_nodes)

# Print the missing genes
print(missing_genes)
```

```
## [1] "C3orf67" "C4orf47" "CCDC84"  "COL18A1" "ODF3B"   "PKD2L1"
```

```r
cat("Number of missing genes: ", length(missing_genes), "\n")
```

```
## Number of missing genes:  6
```

```r
# Filter the links to exclude rows with missing genes
links_filtered <- links[links$X.node1 %in% genes_in_nodes & links$node2 %in% genes_in_nodes, ]

# Create the graph with the filtered links
net <- graph_from_data_frame(d = links_filtered, vertices = nodes, directed = FALSE)

# Check the graph structure
print(net)
```

```
## IGRAPH bf9fe44 UN-- 1166 28472 --
## + attr: name (v/c), ensembl_gene_id (v/c), description (v/c),
## | gene_biotype (v/c), start_position (v/n), end_position (v/n),
## | chromosome_name (v/c), strand (v/n), node1_string_id (e/c),
## | node2_string_id (e/c), neighborhood_on_chromosome (e/n), gene_fusion
## | (e/n), phylogenetic_cooccurrence (e/n), homology (e/n), coexpression
## | (e/n), experimentally_determined_interaction (e/n),
## | database_annotated (e/n), automated_textmining (e/n), combined_score
## | (e/n)
## + edges from bf9fe44 (vertex names):
## [1] APOH --A1BG GZMB --A1BG APOA1--A1BG TTR  --A1BG A1BG --C3   A1BG --ORM1
## + ... omitted several edges
```

```
# Calculate the connected components of the network
c <- components(net)

# Identify the largest component by size
largest_component_id <- which.max(c$csize)

# Get the vertex IDs that belong to the largest component
largest_component_vertices <- which(c$membership == largest_component_id)

largest_component_net <- induced_subgraph(net, vids = largest_component_vertices)

# Check if the largest component is actually a connected component
is_largest_component_connected <- is_connected(largest_component_net)

# Print the result
if (is_largest_component_connected) {
  cat("The largest component is a connected component.\n")
} else {
  cat("The largest component is NOT a connected component.\n")
}
```

```
## The largest component is a connected component.
```

```
# Print or inspect the largest component
class(largest_component_net)
```

```
## [1] "igraph"
```

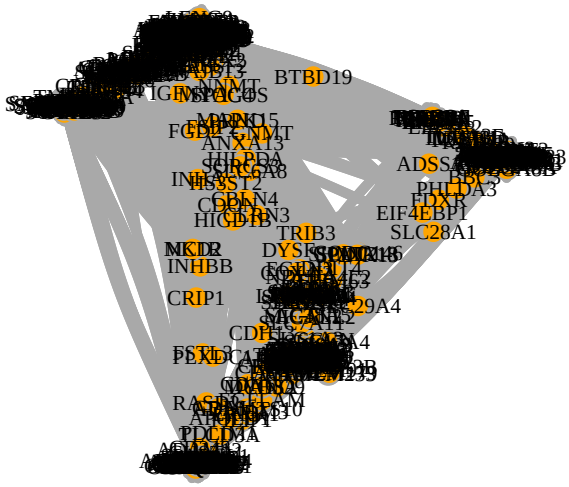
```
print(largest_component_net)
```

```
## IGRAPH aecc048 UN-- 1041 28456 --
## + attr: name (v/c), ensembl_gene_id (v/c), description (v/c),
## | gene_biotype (v/c), start_position (v/n), end_position (v/n),
## | chromosome_name (v/c), strand (v/n), node1_string_id (e/c),
## | node2_string_id (e/c), neighborhood_on_chromosome (e/n), gene_fusion
## | (e/n), phylogenetic_cooccurrence (e/n), homology (e/n), coexpression
## | (e/n), experimentally_determined_interaction (e/n),
## | database_annotated (e/n), automated_textmining (e/n), combined_score
## | (e/n)
```

```
## + edges from aecc048 (vertex names):
## [1] APOH --A1BG GZMB --A1BG APOA1--A1BG TTR  --A1BG A1BG --C3   A1BG --ORM1
## + ... omitted several edges
```

From the report of the largest component net (igraph object) we can see that such component contains 1041 genes out of 1182 total up-regulated DEGs. I'm not familiar with the typical numbers but it strikes as remarkable to have a connected component with 88% of the tested genes in it, considering that such a graph structure involves edges among all node pairs in the net. This might make more likely the unusual results obtained via pattern matching.

```
plot(largest_component_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)
```



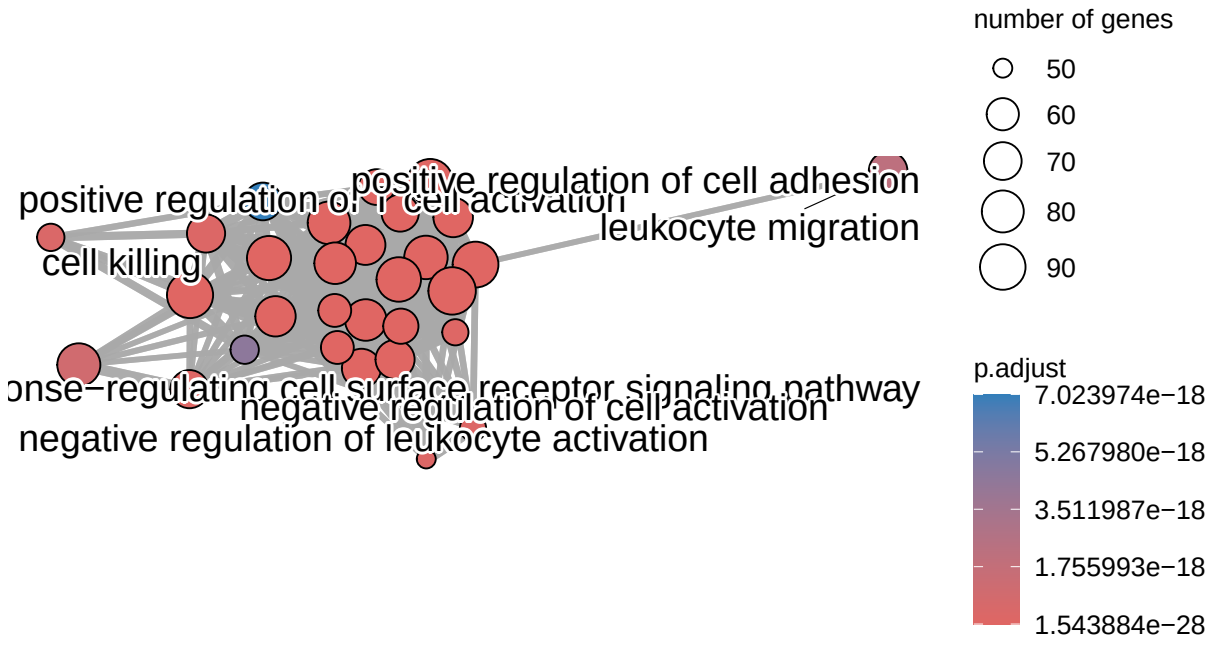
This shows the largest connected component at the level of genes. Here, all nodes (genes) are connected to each other. The spatial disposition of nodes is defined by a layout algorithm which is not here described.

We also perform gene set enrichment as seen in class, this time on the largest component.

```
## Gene-set enrichment analysis of the larger connected component
ego_BP <- enrichGO(gene = V(largest_component_net)$name,
```

```
                    OrgDb = org.Hs.eg.db,
                    keyType = 'SYMBOL',
                    ont = "BP",
                    pAdjustMethod = "BH",
                    pvalueCutoff = 0.05,
                    qvalueCutoff = 0.05)

x <- pairwise_termsim(ego_BP)
emapplot(x)
```

```
## Warning: ggrepel: 23 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



This is very similar to the enrichment on the full DEGs, since only 140 genes are missing from the original. Here nodes are terms, each containing a set of genes, and edges quantify the degree of overlap in the genes contained by pairs of nodes, reflecting an interconnection in biological function. The overall structure is still compact, although not fully connected since, as mentioned, edge size is determined by the overlap.

The end