Diffenrential gene expression analysis of a prostate cancer toy RNA-seq dataset using the state of the art methods for DE, enrichment, annotation and motif analysis.

Load the prostate cancer dataset, generate using the Recount package.

```
load("prostate_cancer.RData")
```

Extract protein coding genes:

```
ensembl <- useEnsembl(biomart = "ensembl",dataset = "hsapiens_gene_ensembl")
query1 <- getBM(attributes=c("ensembl_gene_id", "gene_biotype"),
                filters=c("ensembl_gene_id"),
                values=list(c(r_anno_df$gene_id)),
                mart = ensembl)
query1_protein_coding <- query1[which(query1$gene_biotype=="protein_coding"),]

r_anno_df = subset(r_anno_df, r_anno_df$gene_id %in% query1_protein_coding$ensembl_gene_id)
raw_counts_df = subset(raw_counts_df, rownames(raw_counts_df) %in% query1_protein_coding$ensembl_gene_i

cat(sprintf("Before selecting protein coding: %s \n After selecting protein coding: %s", nrow(query1), n
```

```
## Before selecting protein coding: 57034
##  After selecting protein coding: 19617
```

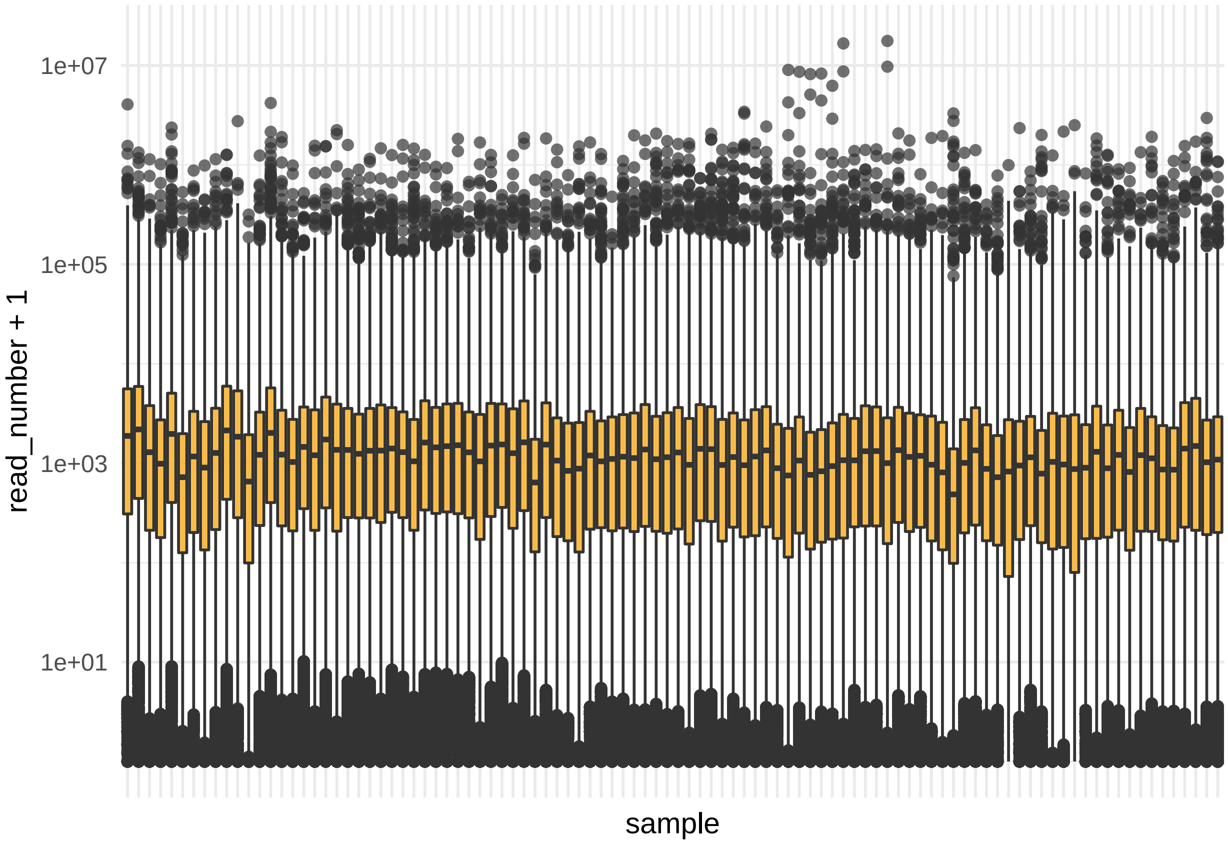
Filter to 20 counts at least in 1 tumor and 1 normal, then normalize the counts:

```
c_anno_case = subset(c_anno_df, c_anno_df$condition=="Case")
c_anno_control = subset(c_anno_df, c_anno_df$condition=="Control")

df_raw_case = raw_counts_df[,c(c_anno_case$sample)]
over_20_case = apply(df_raw_case, 1, function(x) any(x > 20))

df_raw_control = raw_counts_df[,c(c_anno_control$sample)]
over_20_control = apply(df_raw_control, 1, function(x) any(x > 20))

boolean = over_20_case & over_20_control
filtered_raw_df  = cbind(df_raw_case, df_raw_control, boolean)
c_anno_df = rbind(c_anno_case, c_anno_control)

filtered_raw_df = subset(filtered_raw_df, filtered_raw_df$boolean)
filtered_raw_df = filtered_raw_df[,-101]
rownames(r_anno_df) = r_anno_df$gene_id
filtered_anno_df = r_anno_df[rownames(filtered_raw_df),]

edge_dge = DGEList(counts=filtered_raw_df,group=c_anno_df$condition,samples=c_anno_df,genes=filtered_ann

edge_n <- calcNormFactors(edge_dge,method="TMM")
cpm_table <- as.data.frame(round(cpm(edge_n),2))

cat(sprintf("Before filtering: %s \n After filtering: %s", nrow(raw_counts_df), nrow(filtered_raw_df)))
```

```
## Before filtering: 19617
##  After filtering: 17291
```

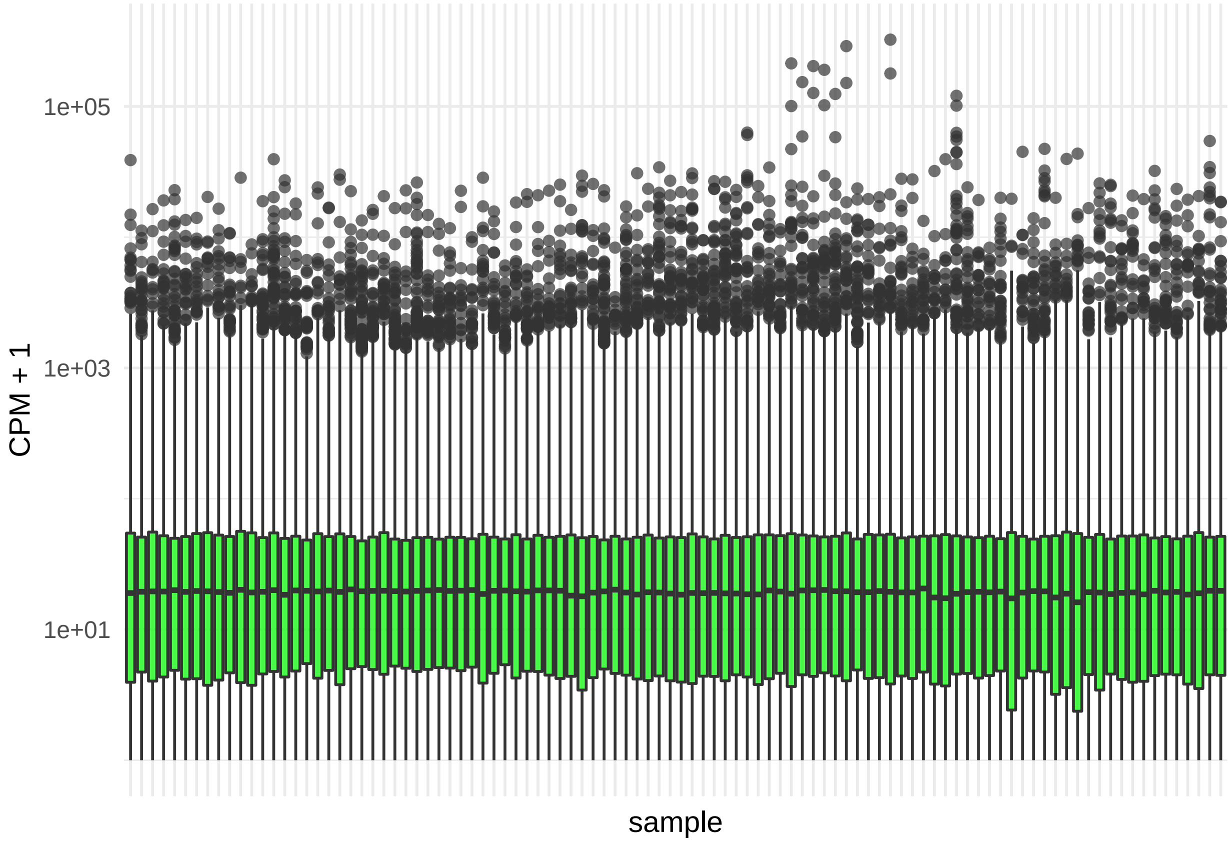
Before normalization:

```
long_counts_df <- gather(filtered_raw_df, key = "sample", value = "read_number")
ggplot(data=long_counts_df,aes(sample,read_number+1)) +
   geom_boxplot(fill="orange",alpha=0.7)+
   theme_minimal() +
   scale_x_discrete(labels=NULL) +
   scale_y_log10()
```



After normalization - medians are aligned pretty well:

```
long_cpm_df <- gather(cpm_table, key = "sample", value = "CPM")
ggplot(data=long_cpm_df,aes(sample,CPM+1)) +
   geom_boxplot(fill="green",alpha=0.7)+
   theme_minimal() +
   scale_x_discrete(labels=NULL) +
   scale_y_log10()
```



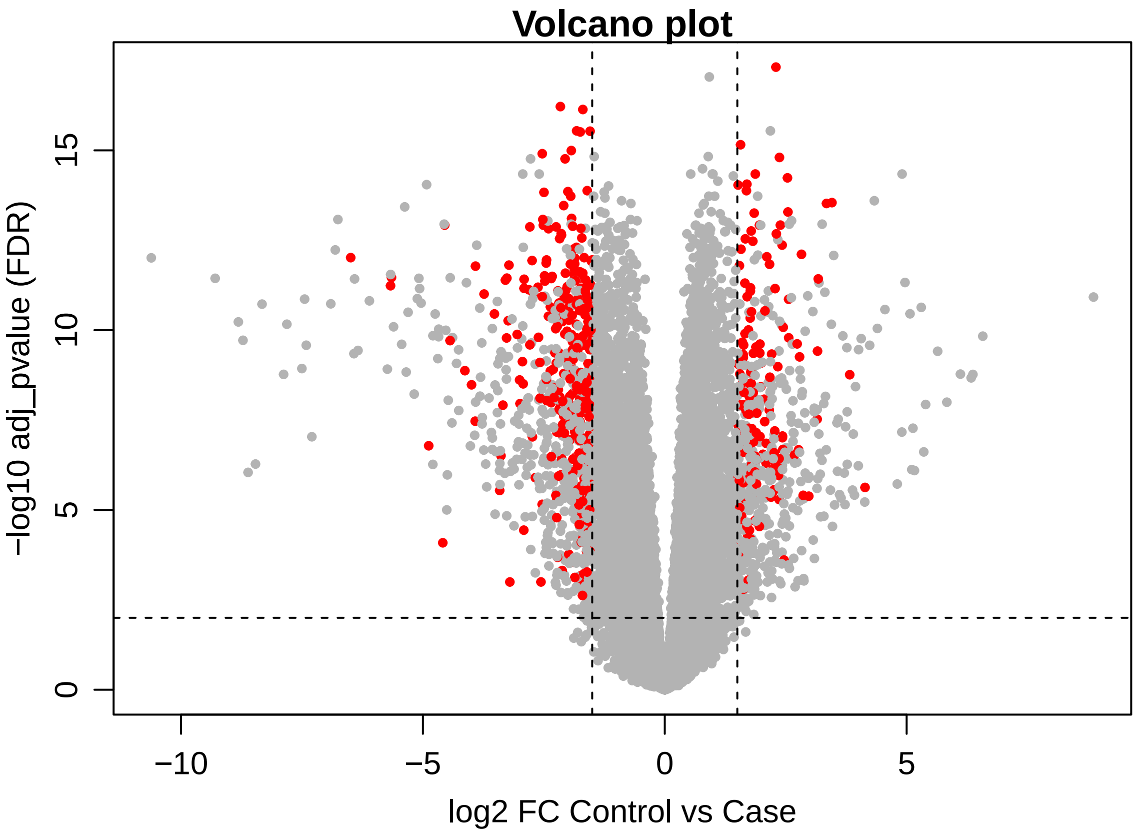
EdgeR analysis - thresholds for corrected p-val of 0.01 and fold change of 1.5 were used.

```
design <- model.matrix(~0+group, data=edge_dge$samples)
colnames(design) <- levels(edge_dge$samples$group)
rownames(design) <- edge_dge$samples$sample

edge_d <- estimateDisp(edge_n,design)
edge_f <- glmQLFit(edge_d,design)

contro <- makeContrasts("Case-Control",levels=design)

edgeRglmQLF <- function(mat=edge_f,contro,cpm_mat=edge_n,label="Cancer vs Control",sig_thr=1,sig_col="l
{
   degs <- glmQLFTest(edge_f,contrast=contro)$table[,-3]
   colnames(degs) <- c("log2_FC","log2_CPM","p_val")
   a_levels <- rownames(contro)[which(contro!=0)]
   a_samples <- which(cpm_mat$samples$group%in%a_levels)
   cpm_sele <- cpm(cpm_mat,log=T)[,a_samples]
   degs$log2_CPM <- apply(cpm_sele,1,function(x) mean(x))
   degs$p_adj <- p.adjust(degs$p_val, method ="BH")
   degs$class <- "="
   degs[which(degs[,sig_col]>=sig_thr & degs$log2_FC>=fc_thr & degs[,pval_col]<=pval_thr),"class"] <- "
   degs[which(degs[,sig_col]>=sig_thr & degs$log2_FC<=(-fc_thr) & degs[,pval_col]<=pval_thr),"class"] <
   degs$class <- as.factor(degs$class)
   degs$comp <- label
   degs$id <- rownames(degs)
   degs <- degs[,c("id","comp","log2_FC","log2_CPM","p_val","p_adj","class")]
   if(names=="TRUE"){
      newnames <- paste(label,colnames(degs),sep="_")
      colnames(degs) <- newnames
   }
   return(degs)
}
DEGs <- edgeRglmQLF(mat=edge_f, cpm_mat=edge_n, contro=contro, sig_thr=1, sig_col="log2_CPM", fc_thr=1.

summary(DEGs$class)
```

```
##     -     +     =
##   322   158 16811
```

Volcano plot:

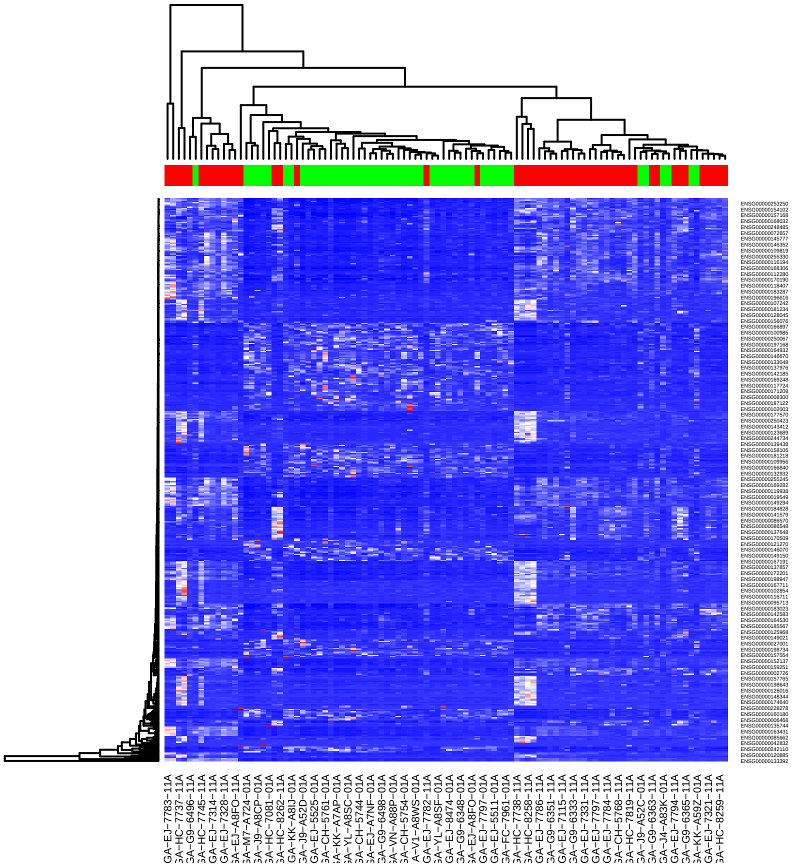
```
input_df <- DEGs
xlabel <- "log2 FC Control vs Case"
ylabel <- "-log10 adj_pvalue (FDR)"
par(fig=c(0,1,0,1), mar=c(4,4,1,2), mgp=c(2, 0.75, 0))
plot(input_df$log2_FC, -log(input_df$p_adj,base=10),xlab=xlabel, ylab=ylabel,
     col=ifelse(input_df$class=="=","grey70","red"), pch=20, frame.plot=TRUE, cex=0.8, main="Volcano pl
```



```
## integer(0)
```

Heatmap: red labels are cases, green are controls.

```
cols <- c(rep("green",50),rep("red",50))
pal <- c("blue","white","red")
pal <- colorRampPalette(pal)(50)
heatmap(as.matrix(cpm_table[which(rownames(cpm_table)%in%DEGs$id[which(DEGs$class!="=")]),]),
        ColSideColors = cols,cexCol = 0.5,margins = c(4,4),col=pal,cexRow = 0.2)
```



Gene set enrichment analysis of up-regulated and down-regulated genes separately using the Gene ontology:

```
upDEGs <- filtered_anno_df[rownames(DEGs %>% filter(class == "+")),]
downDEGs <- filtered_anno_df[rownames(DEGs %>% filter(class == "-")),]

#GO - perform separately on BP and MF, then merge and sort by p.adjust
up_ego_BP <- enrichGO(gene = upDEGs$symbol,
                   OrgDb = org.Hs.eg.db,
                   keyType = 'SYMBOL',
                   ont = "BP",
                   pAdjustMethod = "BH",
                   pvalueCutoff = 0.05,
                   qvalueCutoff = 0.05)
up_ego_MF <- enrichGO(gene = upDEGs$symbol,
                   OrgDb = org.Hs.eg.db,
                   keyType = 'SYMBOL',
                   ont = "MF",
                   pAdjustMethod = "BH",
                   pvalueCutoff = 0.05,
                   qvalueCutoff = 0.05)
up_GO = rbind(up_ego_BP@result, up_ego_MF@result)
up_GO = up_GO[order(up_GO$p.adjust),]


down_ego_BP <- enrichGO(gene = downDEGs$symbol,
                   OrgDb = org.Hs.eg.db,
                   keyType = 'SYMBOL',
                   ont = "BP",
                   pAdjustMethod = "BH",
                   pvalueCutoff = 0.05,
                   qvalueCutoff = 0.05)
down_ego_MF <- enrichGO(gene = downDEGs$symbol,
                   OrgDb = org.Hs.eg.db,
                   keyType = 'SYMBOL',
                   ont = "MF",
                   pAdjustMethod = "BH",
                   pvalueCutoff = 0.05,
                   qvalueCutoff = 0.05)
down_GO = rbind(down_ego_BP@result, down_ego_MF@result)
down_GO = down_GO[order(down_GO$p.adjust),]

#KEGG
# get ENTREZ IDs
up_entrez_ids <- getBM(attributes=c("ensembl_gene_id", "entrezgene_id"),
                filters=c("ensembl_gene_id"),
                values=list(c(upDEGs$gene_id)),
                mart = ensembl)
# remove rows containing NA values
up_entrez_ids = na.omit(up_entrez_ids)
# one Ensembl ID can be associated to multiple ENTREZ IDs - take only one association
up_entrez_ids = up_entrez_ids %>% distinct(ensembl_gene_id, .keep_all = TRUE)
# KEGG enrichment
up_ekegg <- enrichKEGG(gene = up_entrez_ids$entrezgene_id,
                    organism = 'human',
                    pvalueCutoff = 0.05,
                    qvalueCutoff = 0.05)
up_ekegg = up_ekegg@result
# sort
up_ekegg = up_ekegg[order(up_ekegg$p.adjust),]


# repeat for down-regulated
down_entrez_ids <- getBM(attributes=c("ensembl_gene_id", "entrezgene_id"),
                filters=c("ensembl_gene_id"),
                values=list(c(downDEGs$gene_id)),
                mart = ensembl)
down_entrez_ids = na.omit(down_entrez_ids)
down_entrez_ids = down_entrez_ids %>% distinct(ensembl_gene_id, .keep_all = TRUE)
down_ekegg <- enrichKEGG(gene = down_entrez_ids$entrezgene_id,
                    organism = 'human',
                    pvalueCutoff = 0.05,
                    qvalueCutoff = 0.05)
down_ekegg = down_ekegg@result
down_ekegg = down_ekegg[order(down_ekegg$p.adjust),]
```

Top 10 GO enrichment results (Molecular functions and Biological processes sub-ontologies combined) for up and down-regulated genes:

```
up_GO[1:10, c(2,6)]
```

```
##                                                  Description   p.adjust
## GO:0043090                             amino acid import 0.03095844
## GO:1904062 regulation of cation transmembrane transport 0.03095844
## GO:0034368             protein-lipid complex remodeling 0.03095844
## GO:0034369       plasma lipoprotein particle remodeling 0.03095844
## GO:0099177       regulation of trans-synaptic signaling 0.03095844
## GO:0034367        protein-containing complex remodeling 0.03095844
## GO:0010872     regulation of cholesterol esterification 0.04291681
## GO:0033555  multicellular organismal response to stress 0.04615581
## GO:0034374  low-density lipoprotein particle remodeling 0.04881314
## GO:0089718     amino acid import across plasma membrane 0.04881314
```

```
down_GO[1:10, c(2,6)]
```

```
##                                                                 Description     p.adjust
## GO:1901681                                     sulfur compound binding 1.059468e-06
## GO:0004601                                         peroxidase activity 1.865937e-06
## GO:0016684 oxidoreductase activity, acting on peroxide as acceptor 2.691107e-06
## GO:0006936                                          muscle contraction 3.718331e-06
## GO:0003012                                       muscle system process 4.065173e-06
## GO:0008201                                             heparin binding 1.021242e-05
## GO:0005539                                   glycosaminoglycan binding 2.088007e-05
## GO:1990748                                     cellular detoxification 3.485123e-05
## GO:0006575          cellular modified amino acid metabolic process 3.514103e-05
## GO:0043295                                         glutathione binding 3.975699e-05
```

Top 10 KEGG enrichment results for up and down-regulated genes:

```
up_ekegg[1:10, c(2,6)]
```

```
##                                      Description  p.adjust
## hsa04145                               Phagosome 0.8310134
## hsa05034                              Alcoholism 0.8310134
## hsa05202 Transcriptional misregulation in cancer 0.8310134
## hsa04512                ECM-receptor interaction 0.8310134
## hsa02010                        ABC transporters 0.8310134
## hsa00565                  Ether lipid metabolism 0.8310134
## hsa04979                  Cholesterol metabolism 0.8310134
## hsa05144                                 Malaria 0.8310134
## hsa05165          Human papillomavirus infection 0.8310134
## hsa04110                              Cell cycle 0.8310134
```

```
down_ekegg[1:10, c(2,6)]
```

```
##                                              Description   p.adjust
## hsa00982            Drug metabolism - cytochrome P450 0.00113259
## hsa00480                       Glutathione metabolism 0.00839557
## hsa04726                         Serotonergic synapse 0.01162980
## hsa05204        Chemical carcinogenesis - DNA adducts 0.01162980
## hsa04510                               Focal adhesion 0.01162980
## hsa00980 Metabolism of xenobiotics by cytochrome P450 0.01824726
## hsa04270           Vascular smooth muscle contraction 0.02081637
## hsa00590                  Arachidonic acid metabolism 0.02081637
## hsa04915                   Estrogen signaling pathway 0.02081637
## hsa04970                           Salivary secretion 0.02948306
```

Using the list of up-regulated genes, none of the KEGG pathways was significantly enriched (adjusted p-value >=0.84). For this reason I visualized the **Estrogen signaling pathway** which had a significant adjusted p-value when using the list of down-regulated genes. PNG located in the folder

```
logFC <- (DEGs %>% filter(class == "-"))$log2_FC
names(logFC) <- down_entrez_ids$entrezgene_id
pathview(gene.data = logFC,
         pathway.id = "hsa04915",
         species = "human")
```

Motif enrichment analysis for **up-regulated** genes' upstream regions. Top 10 motif enrichment scores are shown below:

```
upstream_seqs <- biomaRt::getSequence(id = upDEGs$gene_id, type="ensembl_gene_id", seqType="gene_flank"
row.names(upstream_seqs) <- upstream_seqs$ensembl_gene_id

data(PWMLogn.hg19.MotifDb.Hsap)
res = motifEnrichment(DNAStringSet(upstream_seqs$gene_flank),PWMLogn.hg19.MotifDb.Hsap, score = "affini
report = groupReport(res)
report_df = as.data.frame(report)
report_df[1:10,c(2,4,5)]
```

```
##    target raw.score      p.value
## 1   CEBPB  2.671451 1.040707e-19
## 2   PGAM2  9.652294 1.076358e-19
## 3    MAFK  2.830945 2.156909e-19
## 4    ODC1  3.049552 2.896012e-19
## 5     NNT  2.334489 5.099784e-19
## 6   TFAP4  3.968342 6.931475e-19
## 7  NANOS1  2.725734 7.386397e-19
## 8   ZMAT2  2.418168 7.825662e-19
## 9     JUN  3.176321 1.522346e-18
## 10    NRL  1.496928 1.798141e-18
```

Empirical distribution for the **CEBPB** transcription factor.

```
tf = report_df$target[1]

tf_motif = subset(MotifDb, organism=='Hsapiens' & geneSymbol==tf)
PWM = toPWM(as.list(tf_motif))
ecdf = motifEcdf(PWM,organism = "hg19",quick=TRUE)
thresholds = lapply(ecdf,function(x) quantile(x,0.995))
```

Pattern-matching using all 19 PWMs of **CEBPB** binding sites in the flanking regions of up-regulated genes. 10 genes who's promoter regions have the highest count of scores above the threshold for any PWM are shown below:

```
scores = as.data.frame(motifScores(DNAStringSet(upstream_seqs$gene_flank),PWM,raw.score=FALSE,cutoff=thr
row.names(scores) = row.names(upstream_seqs)
scores$sum = c(apply(scores, 1, sum))
scores$symbol = lapply(upDEGs$symbol, function(x) toString(x))
scores = scores[order(-scores$sum),]

CEBPB_regulated_genes = subset(scores, scores$sum > 0)
CEBPB_regulated_genes[1:10,c(21,20)]
```

```
##                 symbol sum
## ENSG00000151012   CST1 172
## ENSG00000157388  ACSM1 148
## ENSG00000106819 CAMKK2 143
## ENSG00000167332   NEK5 142
## ENSG00000125780   TGM3 137
## ENSG00000204970  TUBB3 132
## ENSG00000188848    ERG 127
## ENSG00000112742 B3GAT1 120
## ENSG00000086205 EEF1A2 117
## ENSG00000166006  MEX3A 116
```

```
cat(sprintf("%s out of %s up-regulated genes have at least one region in their promoters with a binding
```

```
## 146 out of 158 up-regulated genes have at least one region in their promoters with a binding score ov
```

Upload files containing Ensembl IDs for up and down-regulated genes separately to String DB and download the generated PPI files. Mapping files from String DB were used to name the nodes. These files can be found in project directory.

```
# Export lists of ids of up and down regulated genes
#write.table(upDEGs$gene_id,file="upDEGs.txt" , quote=FALSE, row.names=FALSE, col.names=FALSE)
#write.table(downDEGs$gene_id,file="downDEGs.txt" , quote=FALSE, row.names=FALSE, col.names=FALSE)

# up-regulated PPI
up_links = read.delim("upDEGs_PPI.tsv")
up_mappings = read.csv("upDEGs_mapping.tsv", header=TRUE, sep="\t")
up_nodes = unique(up_mappings[,4])
up_net <- graph_from_data_frame(d=up_links,vertices=up_nodes,directed=FALSE)

up_c <- components(up_net)
```

```
up_largest_c = induced_subgraph(up_net, which(up_c$membership == which.max(up_c$csize)) )


#down-regulated PPI
down_links = read.delim("downDEGs_PPI.tsv")
down_mappings = read.csv("downDEGs_mapping.tsv", header=TRUE, sep="\t")
down_nodes = unique(down_mappings[,4])
down_net = graph_from_data_frame(d=down_links, vertices=down_nodes, directed=FALSE)

down_c = components(down_net)
down_largest_c = induced_subgraph(down_net, which(down_c$membership == which.max(down_c$csize)) )
```

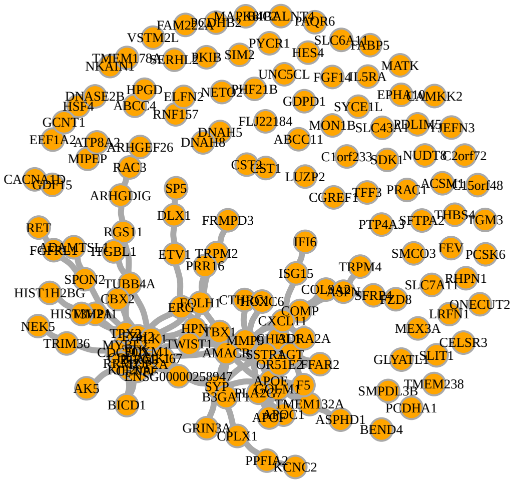
Up-regulated genes PPI:

```
cat(sprintf("Using the up-regulated genes, the largest connected component contains %s out of %s elemen
```

```
## Using the up-regulated genes, the largest connected component contains 77 out of 153 elements.
```

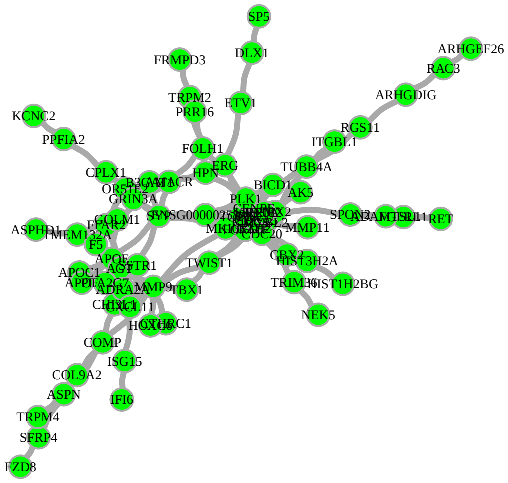
```
plot(up_net,
     edge.width=3,
     vertex.color="orange",
     vertex.size=10,
     vertex.frame.color="darkgray",
     vertex.label.color="black",
     vertex.label.cex=0.4,
     edge.curved=0.2, main="Complete up-regulated PPI")
```

Complete up−regulated PPI



```r
plot(up_largest_c,
     edge.width=3,
     vertex.color="green",
     vertex.size=10,
     vertex.frame.color="darkgray",
     vertex.label.color="black",
     vertex.label.cex=0.4,
     edge.curved=0.2, main="Largest connected component in up-regulated genes")
```

Largest connected component in up−regulated genes



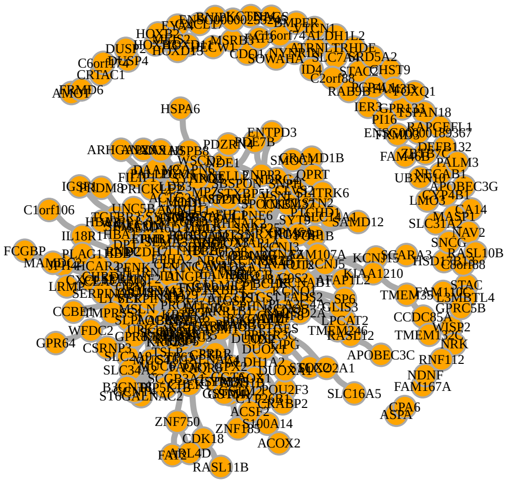
Down-regulated genes PPI:

```
cat(sprintf("Using the down-regulated genes, the largest connected component contains %s out of %s eleme
```

```
## Using the down-regulated genes, the largest connected component contains 257 out of 331 elements.
```

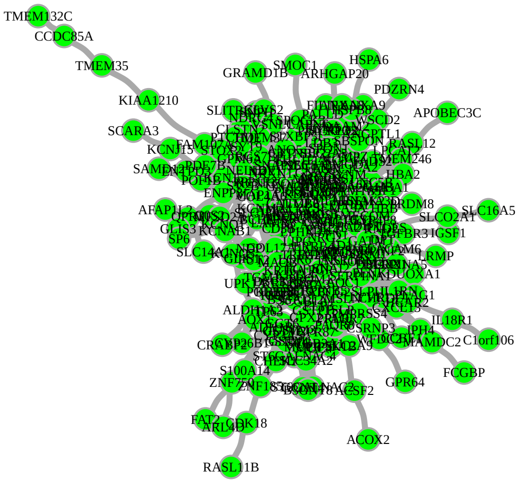
```
plot(down_net,
     edge.width=3,
     vertex.color="orange",
     vertex.size=10,
     vertex.frame.color="darkgray",
     vertex.label.color="black",
     vertex.label.cex=0.4,
     edge.curved=0.2, main="Complete down-regulated PPI")
```

Complete down−regulated PPI



```
plot(down_largest_c,
     edge.width=3,
     vertex.color="green",
     vertex.size=10,
     vertex.frame.color="darkgray",
     vertex.label.color="black",
     vertex.label.cex=0.4,
     edge.curved=0.2, main="Largest connected component in down-regulated genes")
```

Largest connected component in down−regulated genes


TMEM132C
CCDC85A
TMEM35
KIAA1210
SCARA3
GRAMD1B
SMOC1
ARHGAP20
HSPA6
PDZRN4
APOBEC3C
WSCD2
RASL12
HBA2
PRDM8
SLC16A5
SLCO2A1
IGSF1
LRMP
DUOXA1
IL18R1
C1orf106
MAMDC2
FCGBP
GPR64
ACOX2
CSF2
AFAP1L2
GLIS3
SP6
KCNAB1
ENPP
UPK
ALDH1A3
CRABP2
S100A14
ZNF750
FAT2
ARL4D
CDK18
RASL11B
SAMD
KCNH5