Comparison of Differential Expression

Tolga Tabanli

2025-01-25

This chapter deals with how the results of DESeq analysis was visualized and used to graphically compare the differences between IDP expression and the overall expression patterns. I have used bar plots to distinguish the number of up and down-regulated genes, heatmaps for comparison of expression patterns and clusters in different genotypes and lastly gene enrichment analysis to find out the functions of the IDPs implicated in heat shock and possible broken down in knock-down and knock-out.

Libraries and Preparation

The used libraries give an overview of this chapter. IDPs were read from the RDS data. The relevant DESeq results from DESeq pipeline were read in and the label "ref" was used for saves and plots. The results were lastly filtered for significance.

```
library(tidyverse)
library(UpSetR)
library(gprofiler2)
library(ggpubr)
library(janitor)

idps <- readRDS("IDP decisions/commons_modes.Rds")

deseq_results <- readRDS("deseq_reference37.Rds") %>%
  map(as.data.frame)
ref <- 37

deseq_sig <- deseq_results %>%
  map(drop_na) %>%
  map(arrange, padj) %>%
  map(~ dplyr::select(.x, log2FoldChange, padj)) %>%
  map(~ filter(.x, padj < 0.05)) %>%
  map(~ filter(.x, abs(log2FoldChange) > 0.6)) # for lfc filtering
```

Up or Down?

To be able follow what the ratio of up- and down-regulated genes under different conditions was, the filtered DESeq results were labeled according to direction of regulation:

```
expr_direction <- deseq_sig %>%
  map(~ mutate(.x, direction = case_when(
    log2FoldChange < 0 ~ "Down",
    log2FoldChange > 0 ~ "Up"
  ))) %>%
  map(rownames_to_column) %>%
  map(~ mutate(.x, idp = case_when(
```

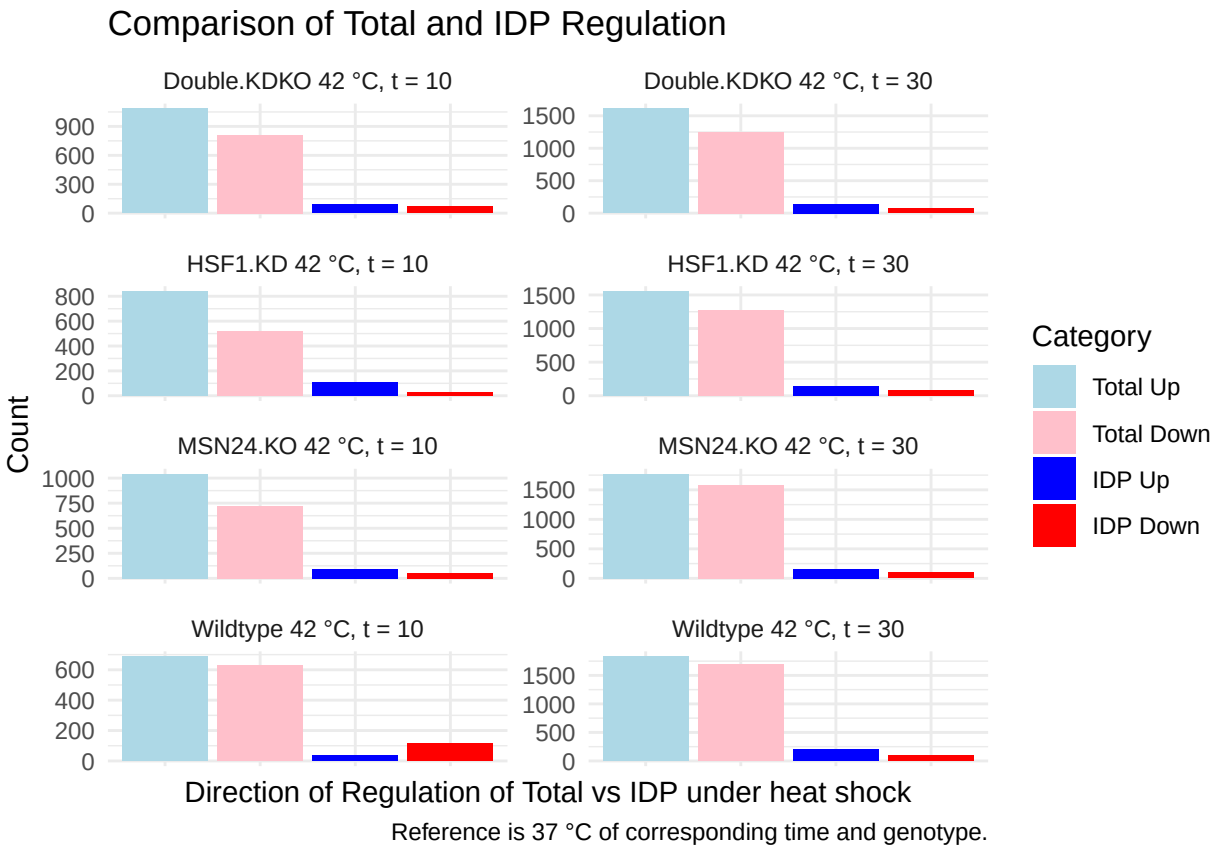
```
    rowname %in% idps ~ TRUE,
    !(rowname %in% idps) ~ FALSE
  )))
```

To compare IDPs' expression behaviour with the background (whole genome), I needed to use a function to convert the up/down labels to counts both for whole genome and for IDPs, a kind of tabularization. In short, what this does is 1) summarise counts for Up/Downregulated genes, 2) repeat for IDPs, and 3) return a new data frame consisting of these four groups.

```
convert_to_counts <- function(df) {
  n_total <- df %>%
    group_by(direction) %>%
    summarise(count = n(), .groups = "drop") %>%
    mutate(group = paste0("Total ", direction))

  n_idp <- df %>%
    filter(idp == TRUE) %>%
    group_by(direction) %>%
    summarise(count = n(), .groups = "drop") %>%
    mutate(group = paste0("IDP ", direction))

  bind_rows(n_total, n_idp)
}
```

Then, the data with information on regulation direction was passed through tabularization function described above and then plotted on a bar plot.

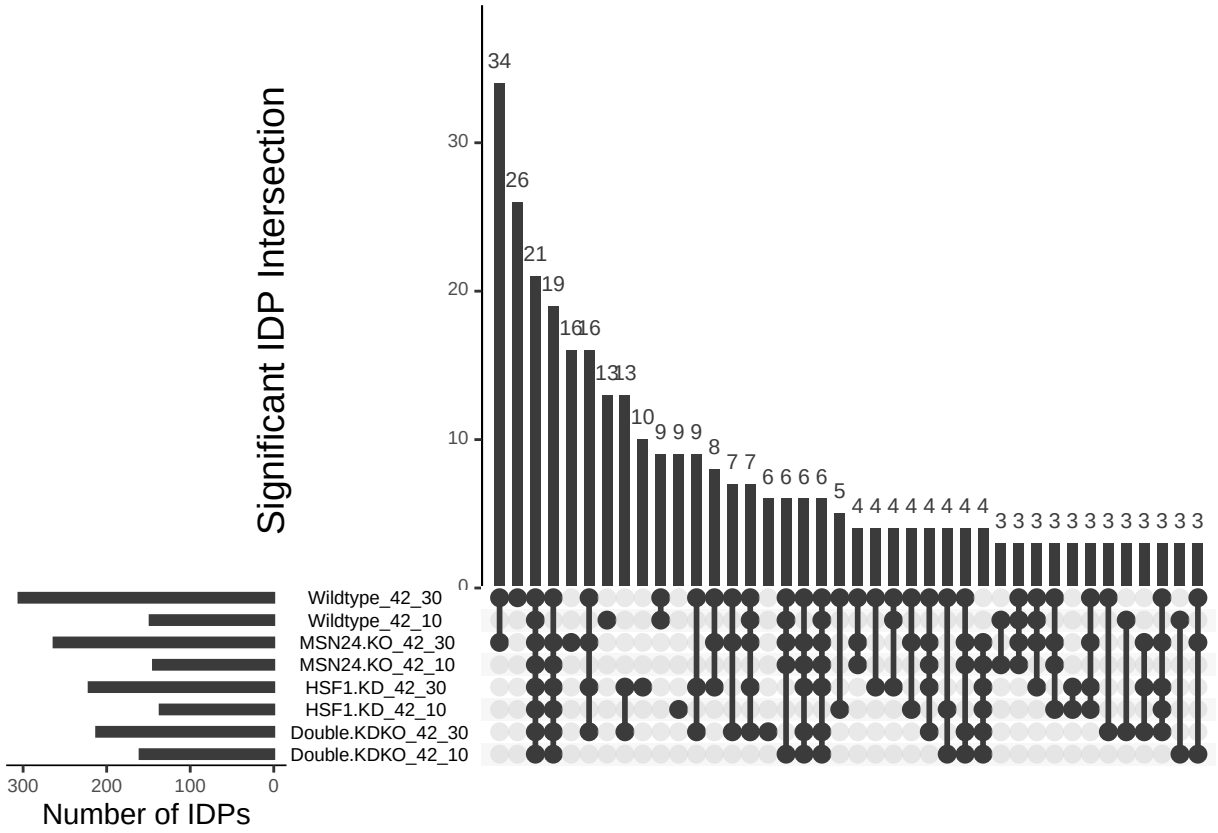
```
expr_direction %>%
  map_dfr(convert_to_counts, .id = "dataset") %>%
  # filter(str_starts(dataset, "Wildtype")) %>% # for creating Wild-type only
  mutate(group = factor(group, levels = c("Total Up", "Total Down",
                                          "IDP Up", "IDP Down"))) %>%
  ggplot(aes(x = group, y = count, fill = group)) +
  geom_bar(stat = "identity") +
  facet_wrap(~ dataset, ncol = 2, scales = "free",
             labeller = labeller(dataset = function(x) {
               gsub("(\\w+?)_(\\d+?)_(\\d+?)", "\\1 \\2 °C, t = \\3", x)
             })) +
  labs(
    title = "Comparison of Total and IDP Regulation",
    x = "Direction of Regulation of Total vs IDP under heat shock",
    caption = paste("Reference is", ref, "°C of corresponding time and genotype."),
    y = "Count",
    fill = "Category"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_blank()) +
  scale_fill_manual(values = c(
    "IDP Up" = "blue",
    "Total Up" = "lightblue",
    "IDP Down" = "red",
    "Total Down" = "pink"
  ))
```



UpSet

To see if there is an overlap between the significant IDPs from the results of different conditions, I used UpSet plot. The row names are the gene names. Text scale parameters indicate the font sizes used in the plot.

```
deseq_sig_idps <- deseq_sig %>%
  map(row.names) %>%
  map(intersect, idps)

upset(fromList(deseq_sig_idps), sets = names(deseq_sig_idps),
      keep.order = T,
      nsets = length(deseq_sig_idps),
      point.size = 3, line.size = 1,
      mainbar.y.label = "Significant IDP Intersection",
      sets.x.label = "Number of IDPs",
      order.by = c("freq"),
      text.scale = c(1.7, 1, 1.3, 1, 1, 1.3))
```



A similar UpSet plot was created for the intersection on only wild-type. Here is given only the set for the input to upset():

```
updown_wt <- expr_direction %>%
  keep(str_starts(names(.), "Wildtype")) %>%
  map(filter, idp) %>%
  map(~ mutate(.x, direction = factor(.x$direction, levels = c("Up", "Down")))) %>%
  map(~ split(.x$rowname, .x$direction)) %>%
  flatten() %>%
  set_names(c("Wildtype_42_10 Up", "Wildtype_42_10 Down",
              "Wildtype_42_30 Up", "Wildtype_42_30 Down"))
```

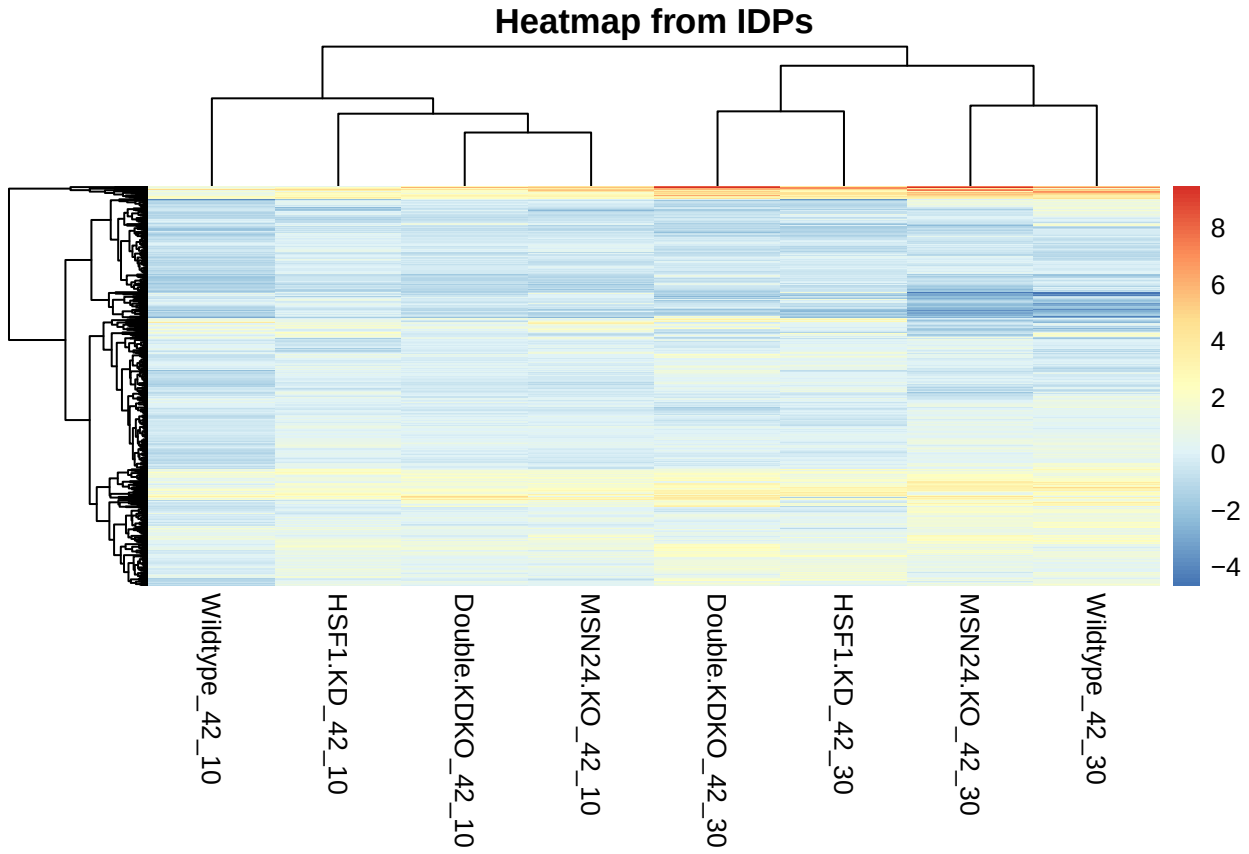
Heatmaps

Heatmaps were used the compare a possible pattern of expression that was hypothesized to change between all genes and IDPs. The data preparation step included:

```
library(pheatmap)
library(tidyverse)

# PREPARE
idps <- readRDS("IDP decisions/commons_modes.Rds")
heat_matrix <- data.frame(rowname = idps) # For IDP step
deseq <- readRDS("deseq_reference37.Rds") %>%
  map(data.frame) %>%
  map(rownames_to_column) %>%
  map(select, rowname, log2FoldChange)
```

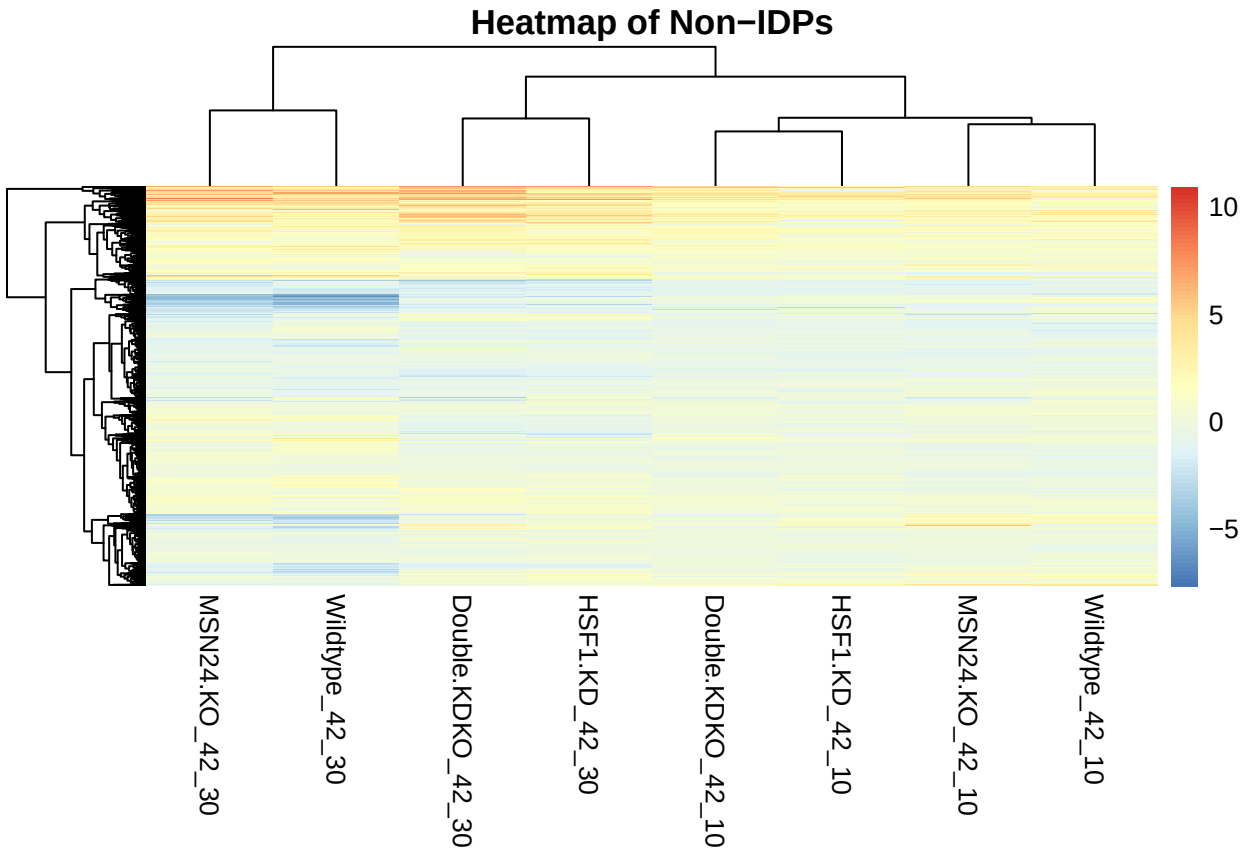
Next, matrices of log2 fold changes were created. For IDP case, an empty matrix with row names as IDPs were used as join partner. As heatmap cannot accept NA values in the matrix, NA-rows in both matrices were dropped. (The NA values are naturally generated in DESeq in cases such as zero or low counts, extreme variability or other statistical test mechanisms.)

```
# Heatmap of IDPs
joined_matrix <- deseq %>%
  imap(~ setNames(.x, gsub("log2FoldChange", .y, names(.x)))) %>%
  map(right_join, heat_matrix) %>% # joins here with IDPs
  purrr::reduce(full_join) %>%
  drop_na() %>%
  column_to_rownames() %>%
  as.matrix()

pheatmap(joined_matrix, show_rownames = F,
         main = "Heatmap from IDPs")
```




```
# Heatmap of Non-IDPs
joined_matrix <- deseq %>%
  map(filter, !(rowname %in% idps)) %>%
  imap(~ setNames(.x, gsub("log2FoldChange", .y, names(.x)))) %>%
  purrr::reduce(full_join) %>%
  drop_na() %>%
  column_to_rownames() %>%
  as.matrix()

pheatmap(joined_matrix, show_rownames = F,
```

```
        main = "Heatmap of Non-IDPs")
```



It should be noted that the heatmaps did not take *significance* into account, meaning that they were not filtered according to p-value or LFC and the color labels mirror all the LFC values of genes that had a non-NA value in each condition.

Enrichment Analysis

Functional enrichment analysis was done with help of g:Profiler interface gprofiler2. Dotplot was implemented using ggplot to facilitate query management, as a direct conversion of gost() results to other packages such enrich was not successful (multiquery requests would cause problems, another option would be to vertically add results of single queries).

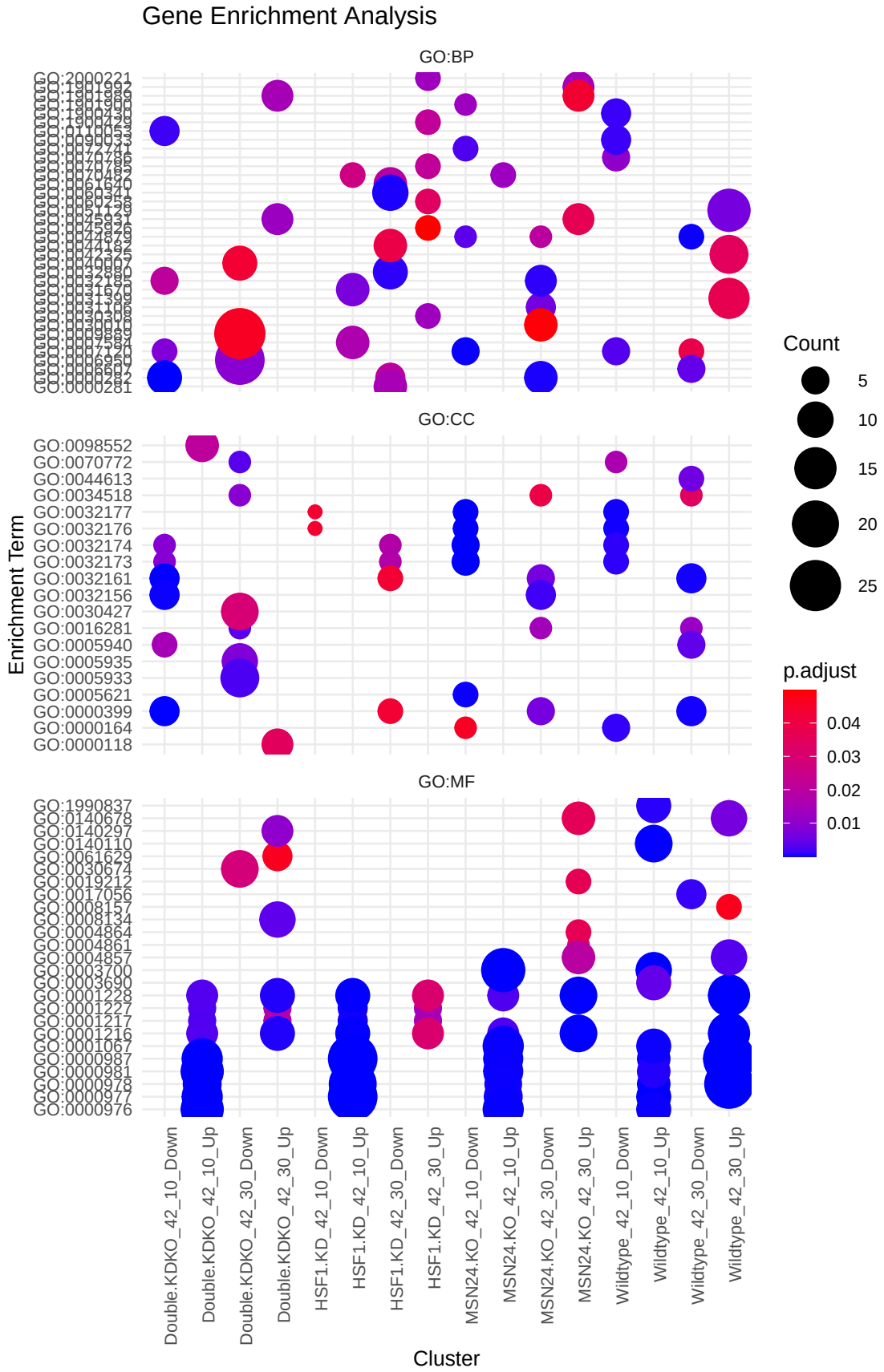
First, a function was defined the underlying dotplot data which are the enrichment data for up and down regulation (reg) in each condition (subdeseq), resulting in 16 sets.

```
dotplot_updown <- function(sub_deseq, reg) {
  xdirection <- sub_deseq %>%
    filter(direction == reg & idp) %>%
    pull(rowname)
  multi_gp = gost(query = xdirection,
                  organism = "scerevisiae",
                  multi_query = FALSE, evcodes = TRUE)
  gp_mod = multi_gp$result[,c("query", "source", "term_id",
                              "term_name", "p_value", "query_size",
                              "intersection_size", "term_size",
                              "effective_domain_size", "intersection")]
```

```
  gp_mod$GeneRatio = gp_mod$intersection_size / gp_mod$query_size
  gp_mod$BgRatio = paste0(gp_mod$term_size, "/", gp_mod$effective_domain_size)
  names(gp_mod) = c("Cluster", "Category", "ID", "Description", "p.adjust",
                    "query_size", "Count", "term_size", "effective_domain_size",
                    "geneID", "GeneRatio", "BgRatio")
  gp_mod$geneID = gsub(",", "/", gp_mod$geneID)
  return(gp_mod)
}
```

Finally, the function was called for each condition in a for loop, with plots saved and then arranged in a grid. Importantly, I sort the gost() result according to increasing term_size and decreasing Count, to get more specific terms that are shared by as most of the genes as possible. Here, only the GO-IDs are plotted due to size issues. While concentrating on Wildtypes, the filter that was commented out was also carried out.

```
dot_data <- list()
expr_names <- names(expr_direction)
for (cond in seq_along(expr_direction)*2) {
  label <- expr_names[[cond/2]]
  dot_data[[cond - 1]] <- dotplot_updown(expr_direction[[cond/2]], "Up") %>%
    mutate(Cluster = paste0(label, "_", "Up"))
  dot_data[[cond]] <- dotplot_updown(expr_direction[[cond/2]], "Down") %>%
    mutate(Cluster = paste0(label, "_", "Down"))
}

dot_data %>%
  map(filter, Category %in% c("GO:MF", "GO:BP", "GO:CC")) %>%
  map(arrange, term_size, desc(Count)) %>%
  map(slice_head, n = 10) %>%
  bind_rows() %>%
  #filter(str_starts(Cluster, "Wildtype")) %>%
  ggplot(aes(x = Cluster, y = ID, size = Count, color = p.adjust)) +
  geom_point() +
  scale_size_continuous(range = c(3, 12)) +
  scale_color_gradient(low = "blue", high = "red") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle=90, hjust=1, vjust=1)) +
  labs(x = "Cluster", y = "Enrichment Term", title = "Gene Enrichment Analysis") +
  facet_wrap(Category ~ ., scales = "free_y", ncol = 1)
```


Gene Enrichment Analysis
GO:BP
GO:CC
GO:MF
Enrichment Term
GO:0098552
GO:0070772
GO:0044613
GO:0034518
GO:0032177
GO:0032176
GO:0032174
GO:0032173
GO:0032161
GO:0032156
GO:0030427
GO:0016281
GO:0005940
GO:0005935
GO:0005933
GO:0005621
GO:0000399
GO:0000164
GO:0000118
GO:1990837
GO:0140678
GO:0140297
GO:0140110
GO:0061629
GO:0030674
GO:0019212
GO:0017056
GO:0008157
GO:0008134
GO:0004864
GO:0004861
GO:0004857
GO:0003700
GO:0003690
GO:0001228
GO:0001227
GO:0001217
GO:0001216
GO:0001067
GO:0000987
GO:0000981
GO:0000978
GO:0000977
GO:0000976
Double.KDKO_42_10_Down
Double.KDKO_42_10_Up
Double.KDKO_42_30_Down
Double.KDKO_42_30_Up
HSF1.KD_42_10_Down
HSF1.KD_42_10_Up
HSF1.KD_42_30_Down
HSF1.KD_42_30_Up
MSN24.KO_42_10_Down
MSN24.KO_42_10_Up
MSN24.KO_42_30_Down
MSN24.KO_42_30_Up
Wildtype_42_10_Down
Wildtype_42_10_Up
Wildtype_42_30_Down
Wildtype_42_30_Up
Cluster
Count
5
10
15
20
25
p.adjust
0.04
0.03
0.02
0.01