DESeq Call with IDPs

Tolga Tabanli

2025-01-25

DESeq Call with IDPs

This chapter summarizes how the experiment samples were mapped and the DESeq was called accordingly. It also includes the creation of volcano plots and the highlighting of the inferred intrinsically disordered proteins.

Library Calls and Sample Preparations

```r
library(DESeq2)
library(tidyverse)
library(ggpubr)
library(grid)

colData <- read_tsv("sample_mapping.tsv", col_names = T) %>%
  mutate(across(everything(), as.factor)) %>%
  column_to_rownames(var = "sample")

heatshock_counts <- read_tsv(here::here("complex_yeast_heatshock.tsv"), col_names = T) %>%
  column_to_rownames(var = "gene_id") %>%
  relocate(rownames(colData))

knockouts <- colData %>%
  select(knockout) %>%
  distinct() %>%
  pull() %>%
  relevel("Wildtype")

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

Function Definitions for Multiple DESeq and Plotting

Firstly, the function that calls DESeq with given parameters is defined. This function takes the genotype, temperature and time information and carrious out DESeq with reference to temperature = 25 and time = 0 of the respective genotype.

Importantly, this function uses a global assignment to store DESeq results in a list outside of the function's scope. This way was preferred, because the function's return value is used to create the grid plot.

```r
multiple_deseq <- function(countData, colData, knock, temp, t) {
  subColData <- colData %>%
    filter(knockout == knock) %>%
    filter((temperature == 25 & time == 0) | temperature == temp)
  subsample <- subColData %>%
    filter(time %in% c(0, t))
  subcounts <- heatshock_counts %>% select(rownames(subsample))

  dds <- DESeqDataSetFromMatrix(countData = subcounts,
                                colData = subsample,
                                design = ~ time)
  deseq <- DESeq(dds, quiet = TRUE)
  res <- results(deseq)

  deseq_results[[paste0(knock,"_",temp,"_",t)]] <<- res

  return(plot_res(res, knock, temp, t))
}
```

Next function is used for creating an aesthetic grid plot that brings together 16 different volcano plots.

```r
plot_res <- function(res, knock, temp, t) {
  res <- res %>%
    as.data.frame() %>%
    drop_na()
  significant_idps <- res$padj < 0.05 & rownames(res) %in% idps

  grob <- grobTree(textGrob(
    paste("Significant IDP ratio:\n",
          round(sum(significant_idps) / length(idps), digits = 2)),
    x=0.5,  y=0.6))

  p <- ggplot(data = res,
              aes(x = log2FoldChange, y = -log10(pvalue))) +
    scale_color_manual(values = c("#0400ff", "#747880"),
                       labels = c("Significant IDPs", "Others"),
                       breaks = c(TRUE, FALSE)) +
    geom_point(data = res %>% filter(!significant_idps), aes(colour = FALSE)) +
    geom_point(data = res %>% filter(significant_idps), aes(colour = TRUE)) +
    annotation_custom(grob) +
    labs(title = paste(knock, temp, "C, t =", t),
         color = "Significance")
  return(p)
}
```

Multiple-DESeq Call

To analyze data, first the parameter space was created for the comparisons.

```r
times <- c(10, 30)
temps <- c(37, 42)
knockouts <- colData %>%
  dplyr::select(knockout) %>%
  unique() %>%
  pull()

combs <- expand.grid(time = times, temperature = temps, knockout = knockouts)
plots <- list()
```

Next, we generate the DESeq results. While the actual results are being stored in a global list, the individual plots are stored in "plots".

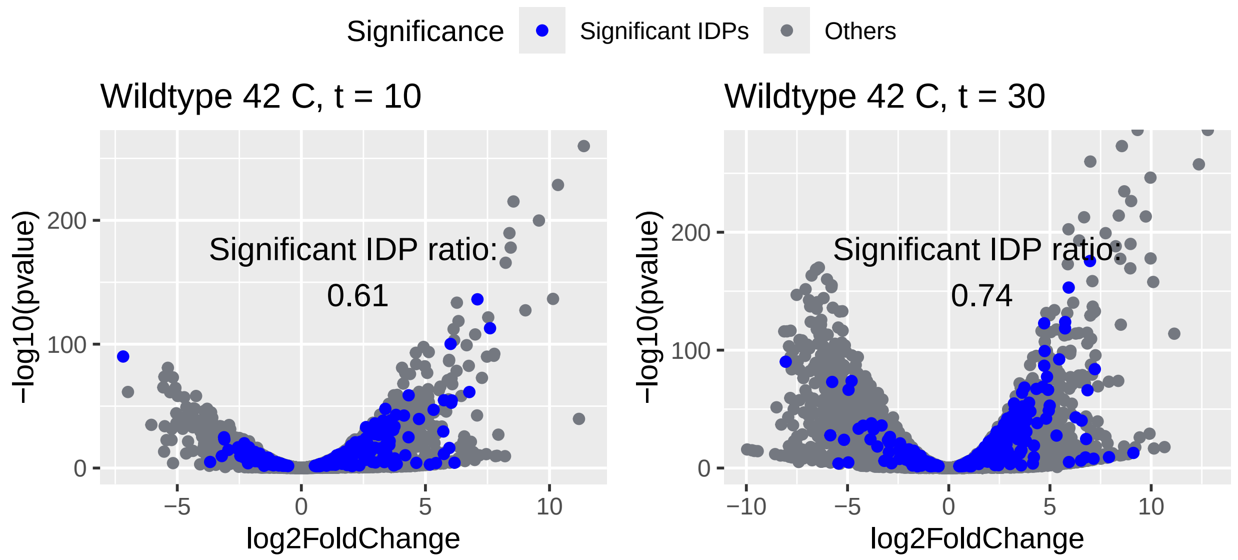
```r
for (row in 1:nrow(combs)) {
  plots[[row]] <- multiple_deseq(heatshock_counts,
                                 colData,
                                 combs[row,"knockout"],
                                 combs[row,"temperature"],
                                 combs[row, "time"])
}
```

Then, the plot was saved using ggarrange for fusion:

```r
ggarrange(plotlist = plots, ncol = 2, nrow = 8, common.legend = T)
```

The same was repeated just to compare the effect of heat shock (42 °C) on IDP expression, using wildtypes only.

```r
# For Heat shock DESeq with reference to wildtype (37) using only wildtypes
multiple_deseq_wt <- function(countData, colData, knock, temp, t) {
  subColData <- colData %>%
    filter(knockout == knock & time == t) %>%
    filter(temperature == 37 | temperature == temp) # REFERENCE AS 37

  subcounts <- heatshock_counts %>% select(rownames(subColData))

  dds <- DESeqDataSetFromMatrix(countData = subcounts,
                                colData = subsample,
                                design = ~ temperature)
  deseq <- DESeq(dds, quiet = TRUE)
  res <- results(deseq)

  deseq_results[[paste0(knock,"_",temp,"_",t)]] <<- res

  return(plot_res(res, knock, temp, t))
}

# reset results
deseq_results <- list()

times <- c(10, 30)
temps <- 42
knockouts <- "Wildtype"

combs <- expand.grid(time = times, temperature = temps, knockout = knockouts)
plots <- list()

for (row in 1:nrow(combs)) {
  plots[[row]] <- multiple_deseq(heatshock_counts,
                                 colData,
                                 as.character(combs[row,"knockout"]),
                                 combs[row,"temperature"],
                                 combs[row, "time"])
}

for (ind in seq_along(deseq_results)) {
  plots[[ind]] <- plot_res(deseq_results[[ind]],
                           combs[ind, "knockout"],
                           combs[ind, "temperature"],
                           combs[ind, "time"])
}

ggarrange(plotlist = plots, ncol = 2, nrow = 1, common.legend = T)
```


Significance
Significant IDPs
Others
Wildtype 42 C, t = 10
Wildtype 42 C, t = 30
Significant IDP ratio:
Significant IDP ratio:
−log10(pvalue)
log2FoldChange