DE analysis

Sergey Naumenko

2021-03-29

Contents

Overview

- Principal Investigator: Beth Overmoyer
- Experiment: RNAseq_analysis_of_inflammatory_breast_cancer_hbc04141
- study 6 was excluded because if low read depth in 3373-3
- https://www.bioconductor.org/packages/release/bioc/vignettes/DEGreport/inst/doc/DEGreport.html
- AnnotationHub. We use ensembl version matching bcbio pipeline - v94.
- HBC materials
- HBC materials - functional analysis
- http://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html

```
## Setup
### Bioconductor and CRAN libraries used

library(DESeq2)
```

```
## Loading required package: S4Vectors
## Loading required package: stats4
## Loading required package: BiocGenerics
## Loading required package: parallel
##
## Attaching package: 'BiocGenerics'
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min
##
## Attaching package: 'S4Vectors'
## The following object is masked from 'package:base':
```

```
##
##     expand.grid
```

```
## Loading required package: IRanges
```

```
## Loading required package: GenomicRanges
```

```
## Loading required package: GenomeInfoDb
```

```
## Loading required package: SummarizedExperiment
```

```
## Loading required package: MatrixGenerics
```

```
## Loading required package: matrixStats
```

```
##
## Attaching package: 'MatrixGenerics'
```

```
## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars
```

```
## Loading required package: Biobase
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
##
## Attaching package: 'Biobase'
```

```
## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians
```

```
## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
```

```
## v readr   1.4.0     v forcats 0.5.1
## -- Conflicts ----------------------------------------- tidyverse_conflicts() --
## x dplyr::collapse()   masks IRanges::collapse()
## x dplyr::combine()    masks Biobase::combine(), BiocGenerics::combine()
## x dplyr::count()      masks matrixStats::count()
## x dplyr::desc()       masks IRanges::desc()
## x tidyr::expand()     masks S4Vectors::expand()
## x dplyr::filter()     masks stats::filter()
## x dplyr::first()      masks S4Vectors::first()
## x dplyr::lag()        masks stats::lag()
## x ggplot2::Position() masks BiocGenerics::Position(), base::Position()
## x purrr::reduce()     masks GenomicRanges::reduce(), IRanges::reduce()
## x dplyr::rename()     masks S4Vectors::rename()
## x dplyr::slice()      masks IRanges::slice()
```

```
library(RColorBrewer)
library(pheatmap)
library(DEGreport)
library(tximport)
library(ggplot2)
library(ggrepel)
library(knitr)
library(AnnotationHub)
```

```
## Loading required package: BiocFileCache
## Loading required package: dbplyr
##
## Attaching package: 'dbplyr'
## The following objects are masked from 'package:dplyr':
##
##     ident, sql
##
## Attaching package: 'AnnotationHub'
## The following object is masked from 'package:Biobase':
##
##     cache
```

```
library(ensembldb)
```

```
## Loading required package: GenomicFeatures
## Loading required package: AnnotationDbi
##
## Attaching package: 'AnnotationDbi'
## The following object is masked from 'package:dplyr':
##
##     select
## Loading required package: AnnotationFilter
##
## Attaching package: 'ensembldb'
```

```
## The following object is masked from 'package:dplyr':
##
##     filter
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
ggplot2::theme_set(theme_light(base_size = 14))

opts_chunk[["set"]](
    cache = FALSE,
    dev = c("png", "pdf"),
    error = TRUE,
    highlight = TRUE,
    message = FALSE,
    prompt = FALSE,
    tidy = FALSE,
    warning = FALSE)
```

```
# Have a folder called 'data', and copy your Salmon folders here from the cluster.
## List all directories containing data
### change the pattern to something specific to your Salmon folders
samples <- list.files(path = "./data/final",
                      full.names = T,
                      pattern = "^S")

## Obtain a vector of all filenames including the path
files <- file.path(samples, "salmon", "quant.sf")
files
```

```
##  [1] "./data/final/S3154-2/salmon/quant.sf"
##  [2] "./data/final/S3169-2/salmon/quant.sf"
##  [3] "./data/final/S3188-2/salmon/quant.sf"
##  [4] "./data/final/S3190-4/salmon/quant.sf"
##  [5] "./data/final/S3193-1/salmon/quant.sf"
##  [6] "./data/final/S3194-3/salmon/quant.sf"
##  [7] "./data/final/S3220-1/salmon/quant.sf"
##  [8] "./data/final/S3234-1/salmon/quant.sf"
##  [9] "./data/final/S3291-3/salmon/quant.sf"
## [10] "./data/final/S3292-3/salmon/quant.sf"
## [11] "./data/final/S3372-1/salmon/quant.sf"
## [12] "./data/final/S3374-2/salmon/quant.sf"
## [13] "./data/final/S3404-1/salmon/quant.sf"
## [14] "./data/final/S3424-1/salmon/quant.sf"
## [15] "./data/final/S3474-3/salmon/quant.sf"
## [16] "./data/final/S3477-1/salmon/quant.sf"
## [17] "./data/final/S3563-3/salmon/quant.sf"
## [18] "./data/final/S3582-4/salmon/quant.sf"
## [19] "./data/final/S3644-1/salmon/quant.sf"
## [20] "./data/final/S3652-1/salmon/quant.sf"
## [21] "./data/final/S3688-2/salmon/quant.sf"
## [22] "./data/final/S3697-3/salmon/quant.sf"
## [23] "./data/final/S3713-1/salmon/quant.sf"
## [24] "./data/final/S3715-2/salmon/quant.sf"
## [25] "./data/final/S3723-1/salmon/quant.sf"
```

```
## [26] "./data/final/S3728-3/salmon/quant.sf"
## [27] "./data/final/S3732-1/salmon/quant.sf"
## [28] "./data/final/S3741-3/salmon/quant.sf"
## [29] "./data/final/S3816-1/salmon/quant.sf"
## [30] "./data/final/S3822-1/salmon/quant.sf"
## [31] "./data/final/S3825-1/salmon/quant.sf"
## [32] "./data/final/S3837-2/salmon/quant.sf"
## [33] "./data/final/S4047-1/salmon/quant.sf"
## [34] "./data/final/S4056-1/salmon/quant.sf"
## [35] "./data/final/S4089-1/salmon/quant.sf"
## [36] "./data/final/S4101-3/salmon/quant.sf"
## [37] "./data/final/S4136-1/salmon/quant.sf"
## [38] "./data/final/S4144-2/salmon/quant.sf"
## [39] "./data/final/S4172-1/salmon/quant.sf"
## [40] "./data/final/S4176-3/salmon/quant.sf"
## [41] "./data/final/S4237-1/salmon/quant.sf"
## [42] "./data/final/S4249-1/salmon/quant.sf"
## [43] "./data/final/S4261-1/salmon/quant.sf"
## [44] "./data/final/S4295-5/salmon/quant.sf"
```

```
## Since all quant files have the same name it is useful to have names for each element
### change the string in str_replace so the pattern matches your filenames
names(files) <- str_replace(samples, "./data/final/", "")
```

```
# Load the data and metadata
meta <- read_csv("tables/metadata_corrected.csv") %>%
  column_to_rownames(var = "samplename")
protein_coding_genes <- read_csv("tables/ensembl_w_description.protein_coding.csv")
```

```
# Connect to AnnotationHub
ah <- AnnotationHub()

# Query AnnotationHub
hs_ens <- query(ah, c("Homo sapiens", "EnsDb"))

# Get Ensembl94 - used in bcbio
hs_ens <- hs_ens[["AH64923"]]

# Extract gene-level information
txdb <- transcripts(hs_ens,
                    return.type = "data.frame") %>%
  dplyr::select(tx_id, gene_id)

genedb <- genes(hs_ens,
                return.type = "data.frame") %>%
  dplyr::select(gene_id, gene_name, symbol)

gene_symbol <- genedb %>% dplyr::select(gene_id, symbol)

hsdb <- inner_join(txdb, genedb)
write.table(hsdb,
            file = "data/ensembl94_hg38_annotations.txt",
            sep = "\t",
            row.names = F,
            quote = F)
```

```
# Read in  a tx2gene file with transcript identifiers in the first column and gene identifiers in the s
#wormdb <- read.table("ensembl94_WBcel235_annotations.txt", sep="\t", header=T)
tx2gene <- hsdb[, c("tx_id", "gene_id")]

# Run tximport
txi_file <- "data/txi.RDS"
if (file.exists(txi_file)){
    txi <- readRDS(txi_file)
}else{
    txi <- tximport(files,
                type = "salmon",
                tx2gene = tx2gene,
                countsFromAbundance = "lengthScaledTPM",
                ignoreTxVersion = FALSE)
    saveRDS(txi, txi_file)
}

# Look at the counts
class(txi)
```

```
## [1] "list"
```

```
attributes(txi)
```

```
## $names
## [1] "abundance"           "counts"              "length"
## [4] "countsFromAbundance"
```

```
txi$counts %>% View()
```

Checking to see that the transcript to gene mapping is correct

When you have annotations that are from a different source from your reference you can run into problems (i.e lose genes). Some checks you can do before proceeding:

1. Look at the dimensions of your count matrix. Do you have ~20k genes present? `dim(txi$counts)`
2. When running `tximport()` you will get a message in your console. If you see something like `transcripts missing from tx2gene` start troubleshooting.

```
dim(txi$counts)
```

```
## [1] 58735    44
```

Sanity check that metadata matches your expression

It is always a good idea to check if:

1. Do you have expression data for all samples listed in your metadata?
2. Are the samples in your expression data in the same order as your metadata?

```
### Check that sample names match in both files
all(colnames(txi$counts) %in% rownames(meta))
```

```
## [1] TRUE
```

```
# Not the same? Make them the same
### This will change depending on what names you have listed!
```

```
#paste0(meta$samplename, "_", meta$library)
#rownames(meta) <- paste0(meta$samplename, "_", meta$library)
#meta$genotype <- relevel(meta$genotype, ref="Wildtype")

### Check that sample names match in both files
all(colnames(txi$counts) %in% rownames(meta))
```

```
## [1] TRUE
```

```
### Check that all samples are in the same order
meta <- meta[colnames(txi$counts),]
all(colnames(txi$counts) == rownames(meta))
```

```
## [1] TRUE
```

Run DESeq2

```
estimating size factors
estimating dispersions
gene-wise dispersion estimates
mean-dispersion relationship
final dispersion estimates
fitting model and testing
```

- Estimating size factors and count normalization
- Gene-wise dispersions
- Mean-dispersion(variance) relationship and the Negative Binomial Model
- Model fitting and hypothesis testing

Wald test

Here we subset protein coding genes.

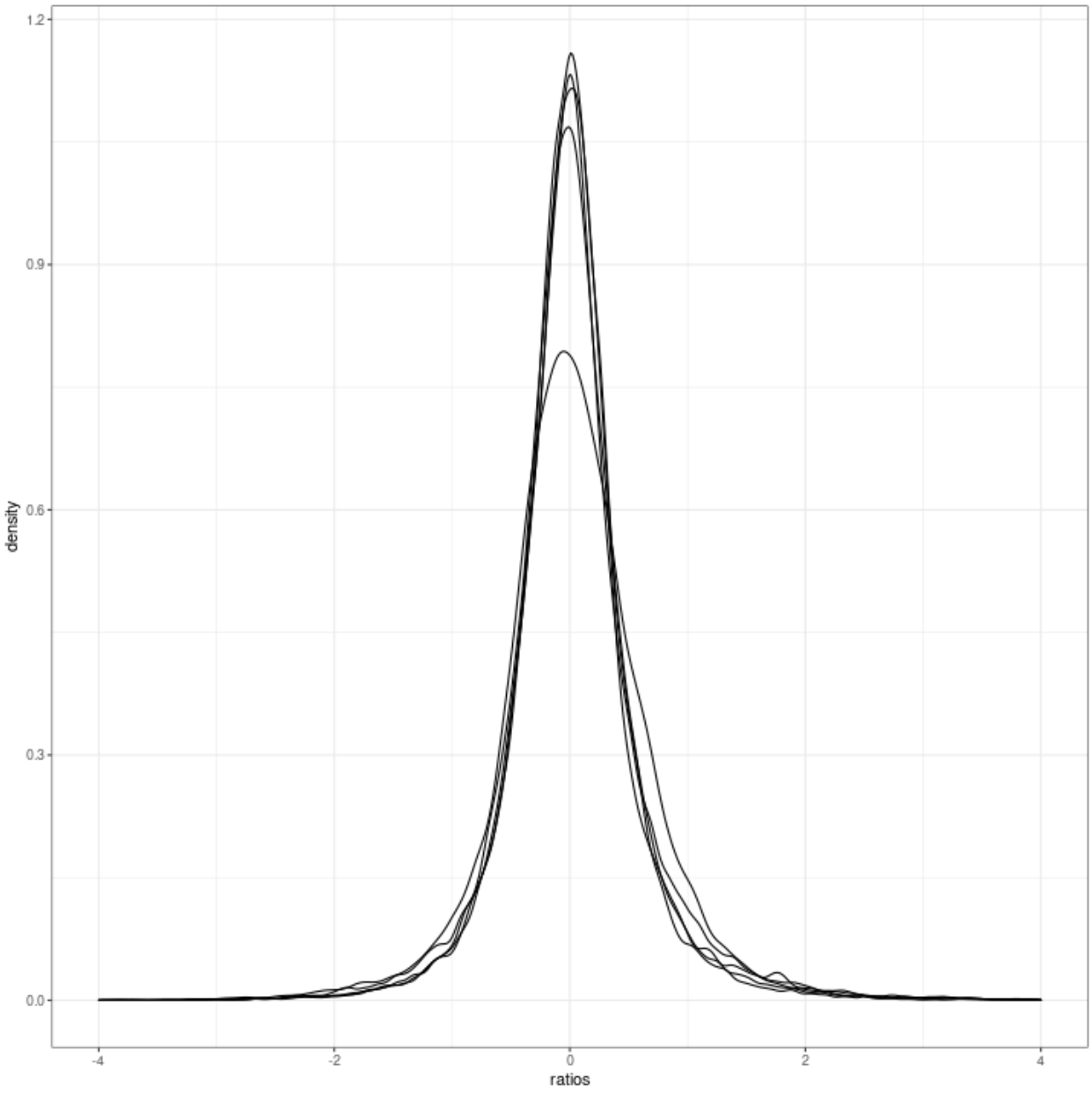
```
## Create DESeq2Dataset object
dds_file <- "data/dds.RDS"
meta$treatment <- as.factor(meta$treatment)
meta$response <- as.factor(meta$response)
meta$er <- as.factor(meta$er)
meta$date_of <- as.factor(meta$date_of)
meta$tumor_percentage <- as.factor(meta$tumor_percentage)
meta$tumor_percentage_high <- as.factor(meta$tumor_percentage_high)

non_responders <- meta %>% dplyr::filter(study_id %in% c(2, 19)) %>% row.names()

if (file.exists(dds_file)){
    dds <- readRDS(dds_file)
}else{
    dds <- DESeqDataSetFromTximport(txi,
                                    colData = meta,
                                    design = ~treatment)

    dds <- dds[,!colnames(dds) %in% non_responders]
    design(dds) <- formula(~treatment + response + er + tumor_percentage_high + date_of)

    # subset protein-coding genes
```

```
    pc_genes <- intersect(protein_coding_genes$ensembl_gene_id, row.names(dds))
    dds <- dds[pc_genes,]
    # 100 reads / 40 samples
    keep <- rowSums(counts(dds)) >= 100
    dds <- dds[keep,]

    # Run DESeq2
    dds <- DESeq(dds)
    saveRDS(dds, dds_file)
}
```

DEGreport QC

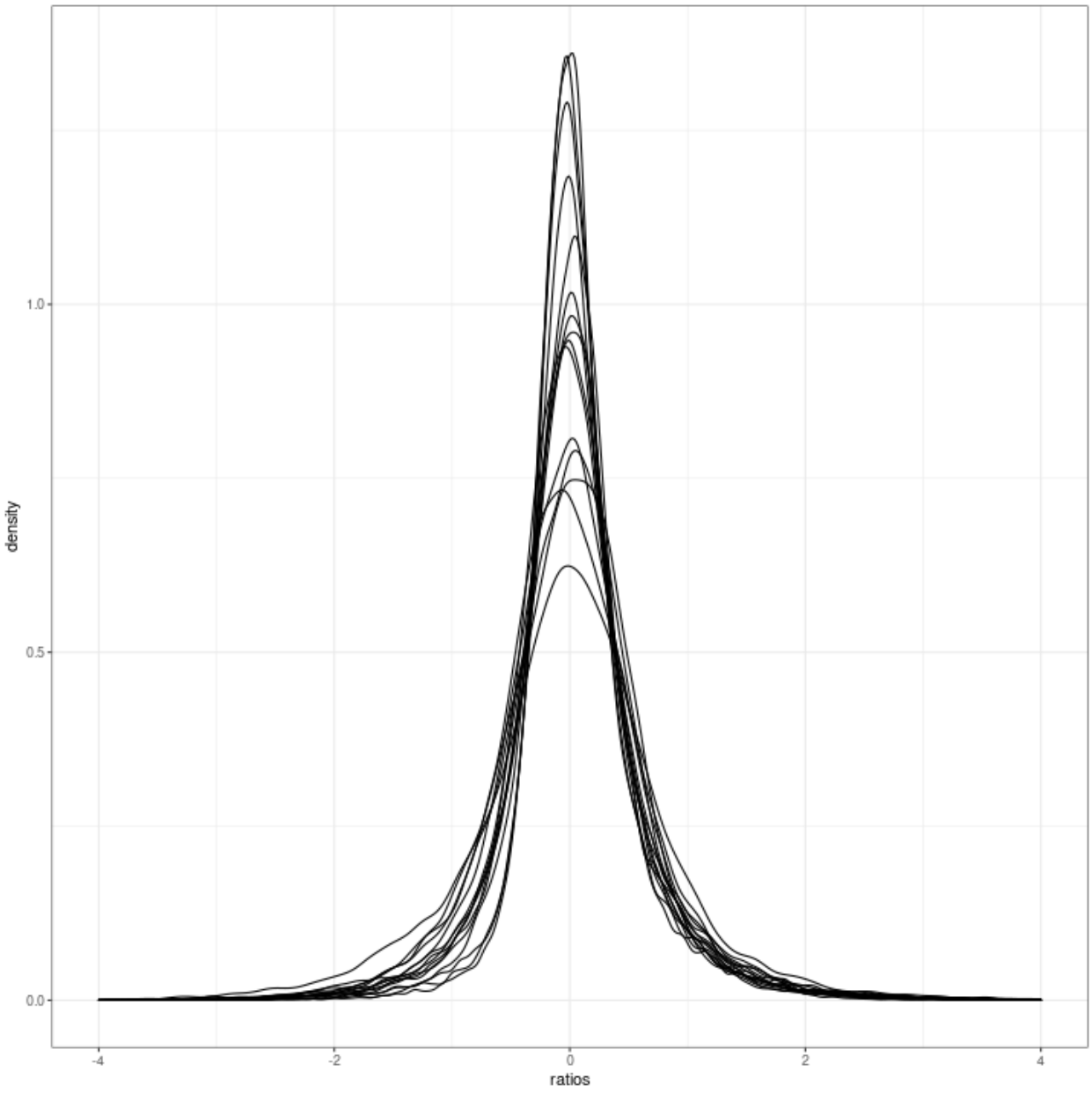
Size factor QC - samples 1-15

```
counts <- counts(dds, normalized = TRUE)
design <- as.data.frame(colData(dds))
degCheckFactors(counts[, 1:5])
```



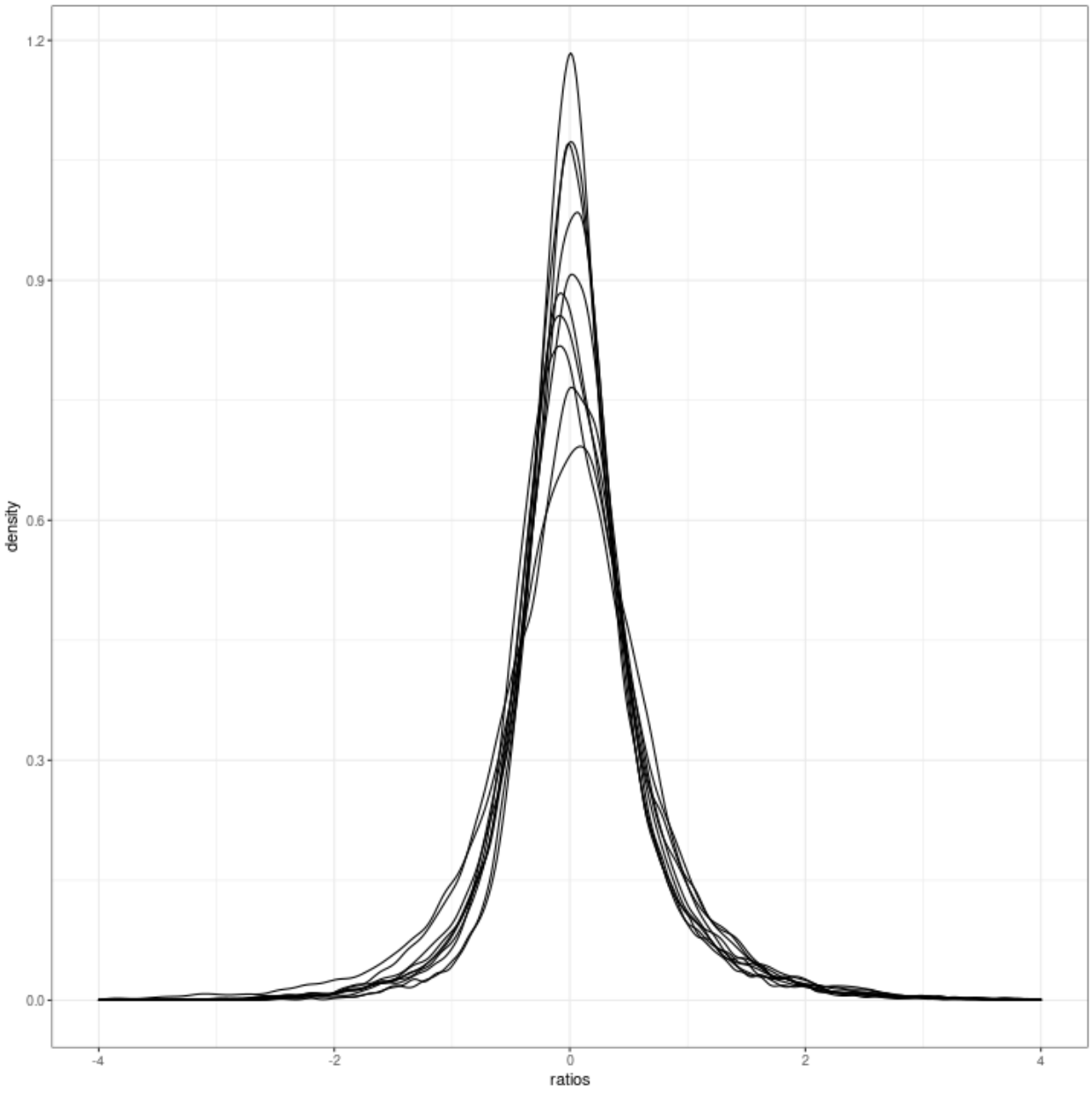
Size factor QC - samples 16-30

```
degCheckFactors(counts[, 16:30])
```



Size factor QC - samples 31-40

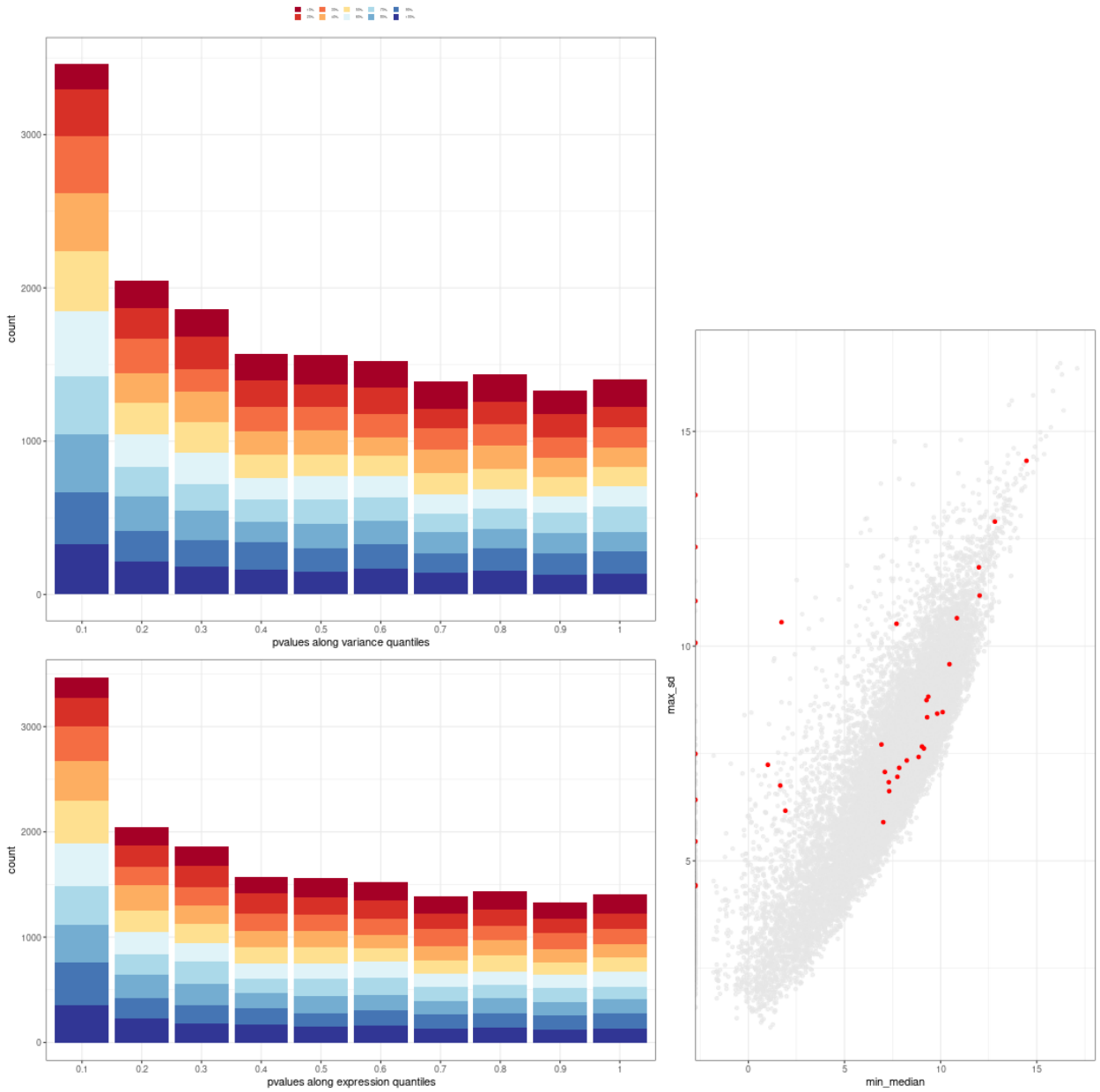
```
degCheckFactors(counts[, 31:40])
```



Mean-Variance QC plots

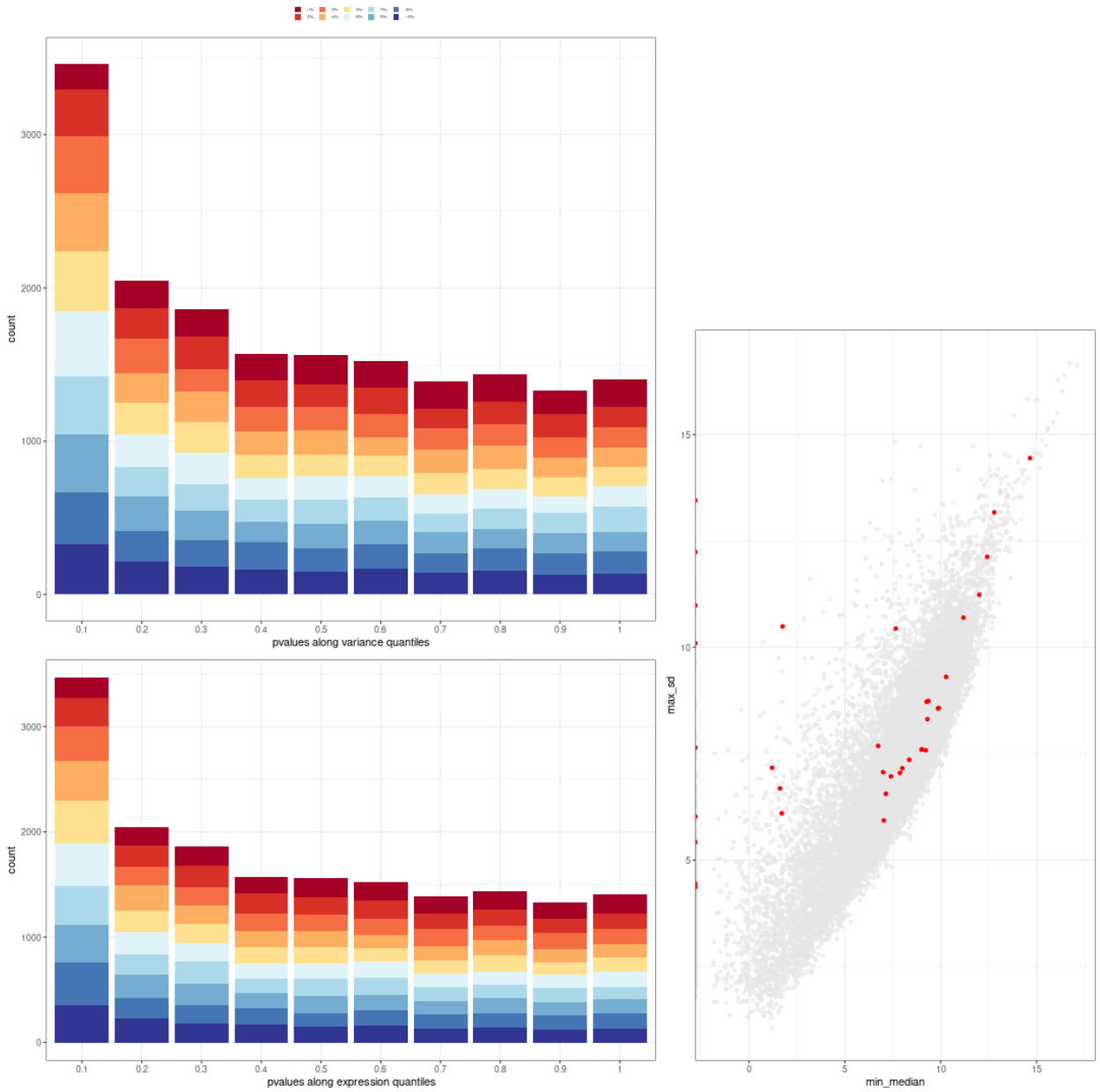
treatment

```
res <- results(dds)
degQC(counts, design[["treatment"]], pvalue = res[["pvalue"]])
```



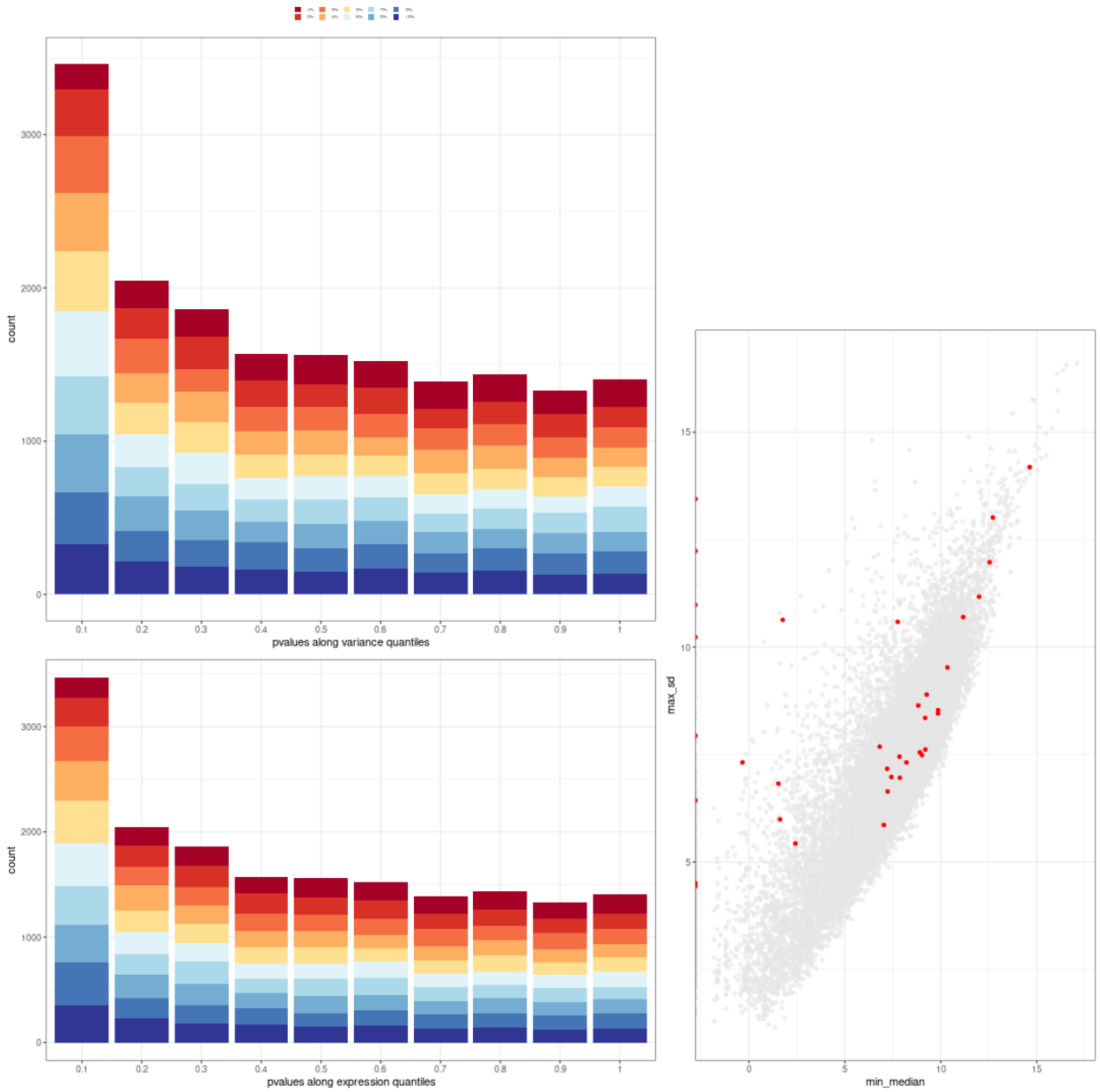
response

```
degQC(counts, design[["response"]], pvalue = res[["pvalue"]])
```



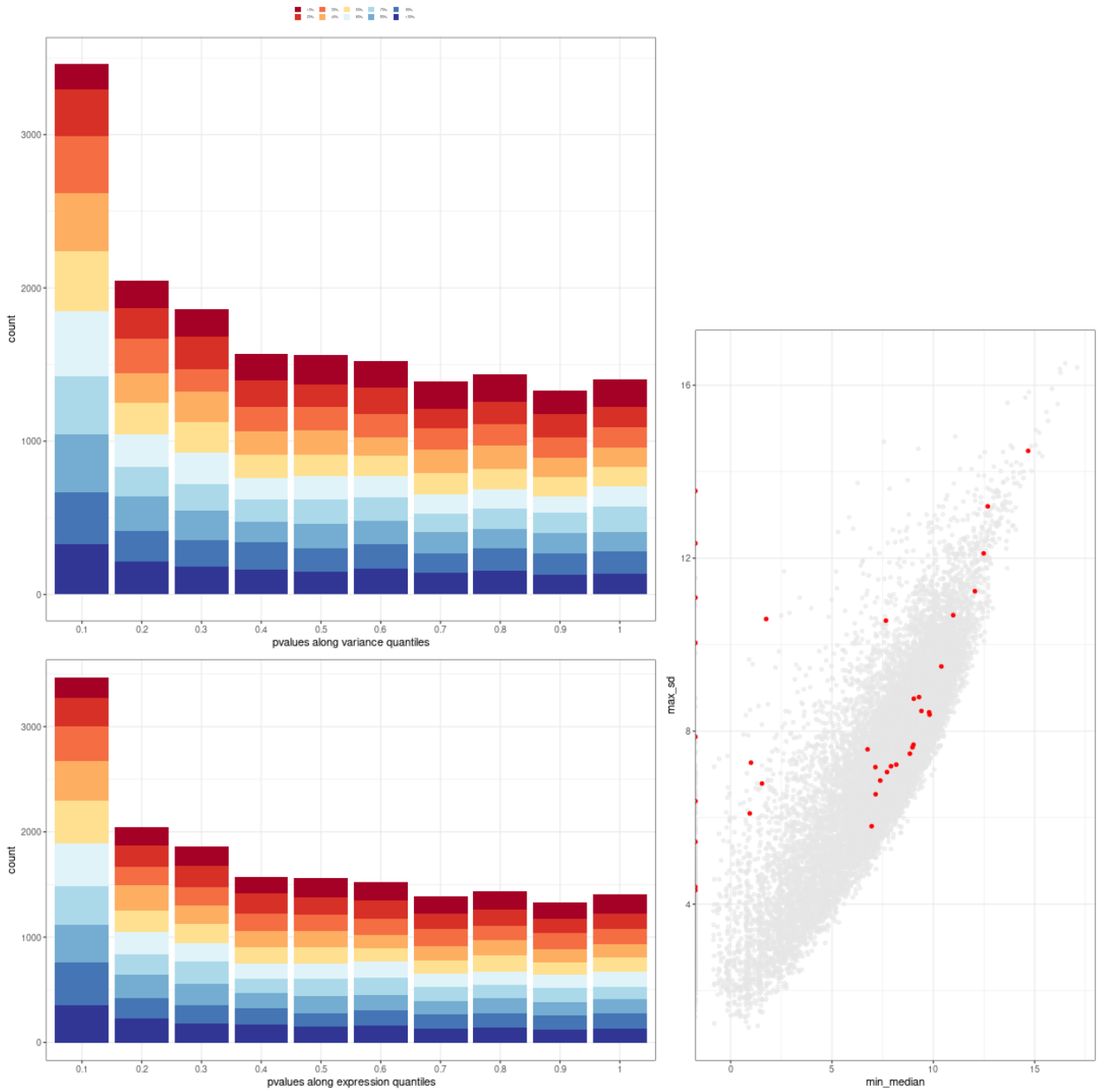
ER

```
degQC(counts, design[["er"]], pvalue = res[["pvalue"]])
```



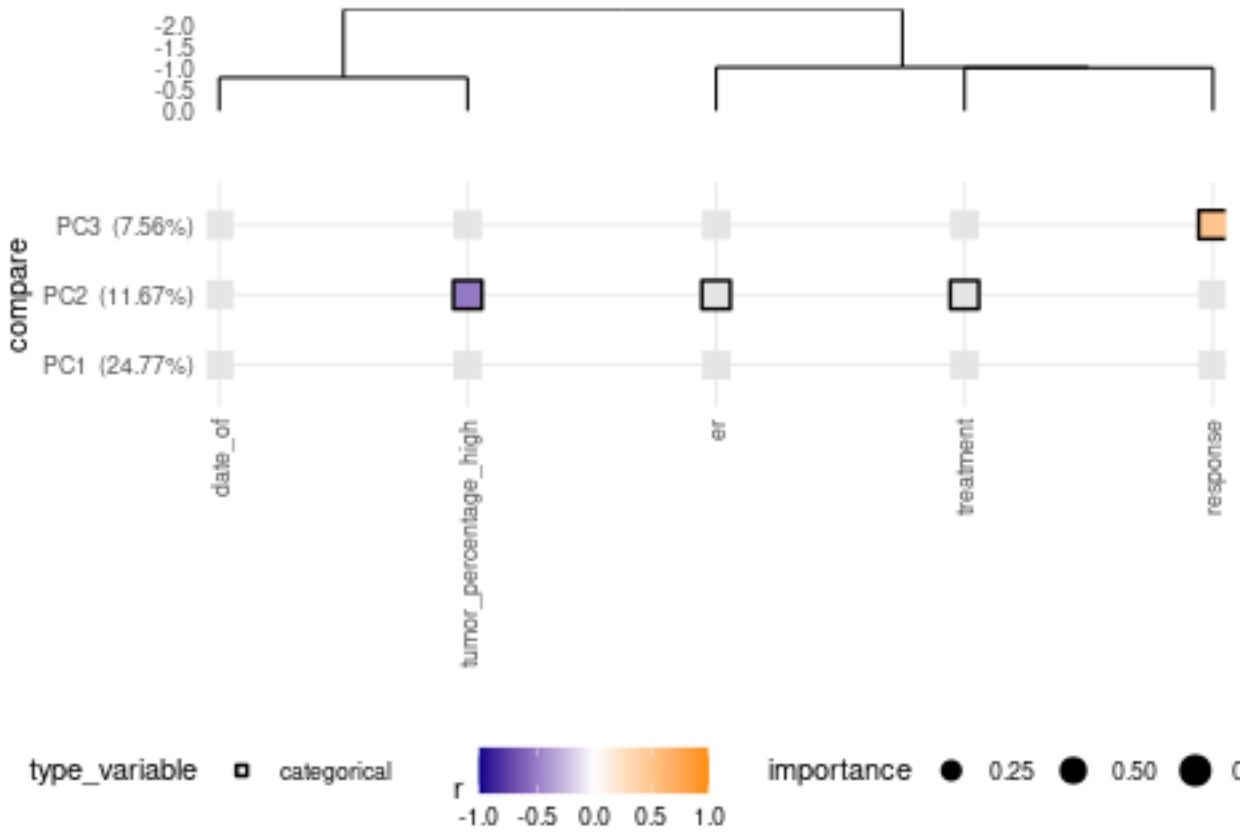
tumor_percentage_high

```
degQC(counts, design[["tumor_percentage_high"]], pvalue = res[["pvalue"]])
```

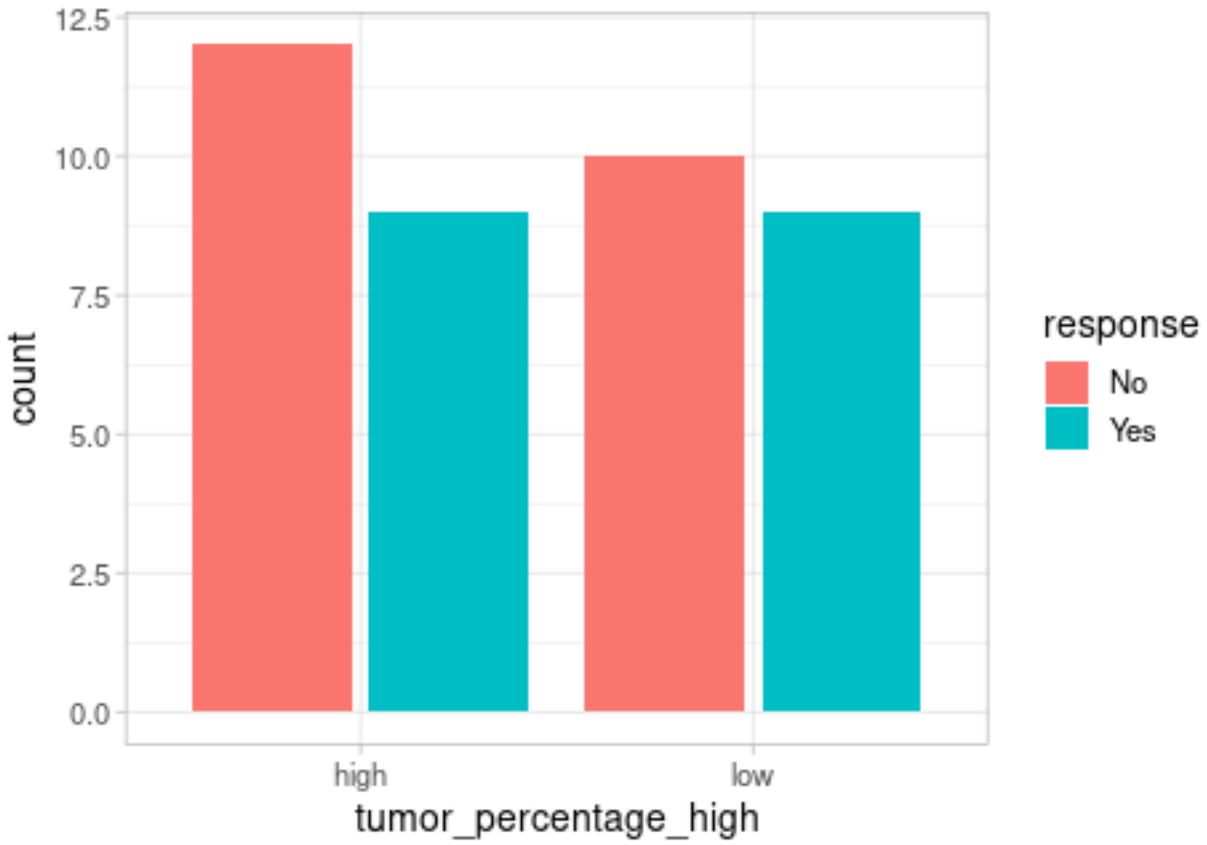


Covariates effect on count data

```
mdata <- colData(dds) %>%  as.data.frame()  %>%
  dplyr::select(treatment, response, er, date_of, tumor_percentage_high)
resCov <- degCovariates(log2(counts(dds)+0.5), mdata)
```

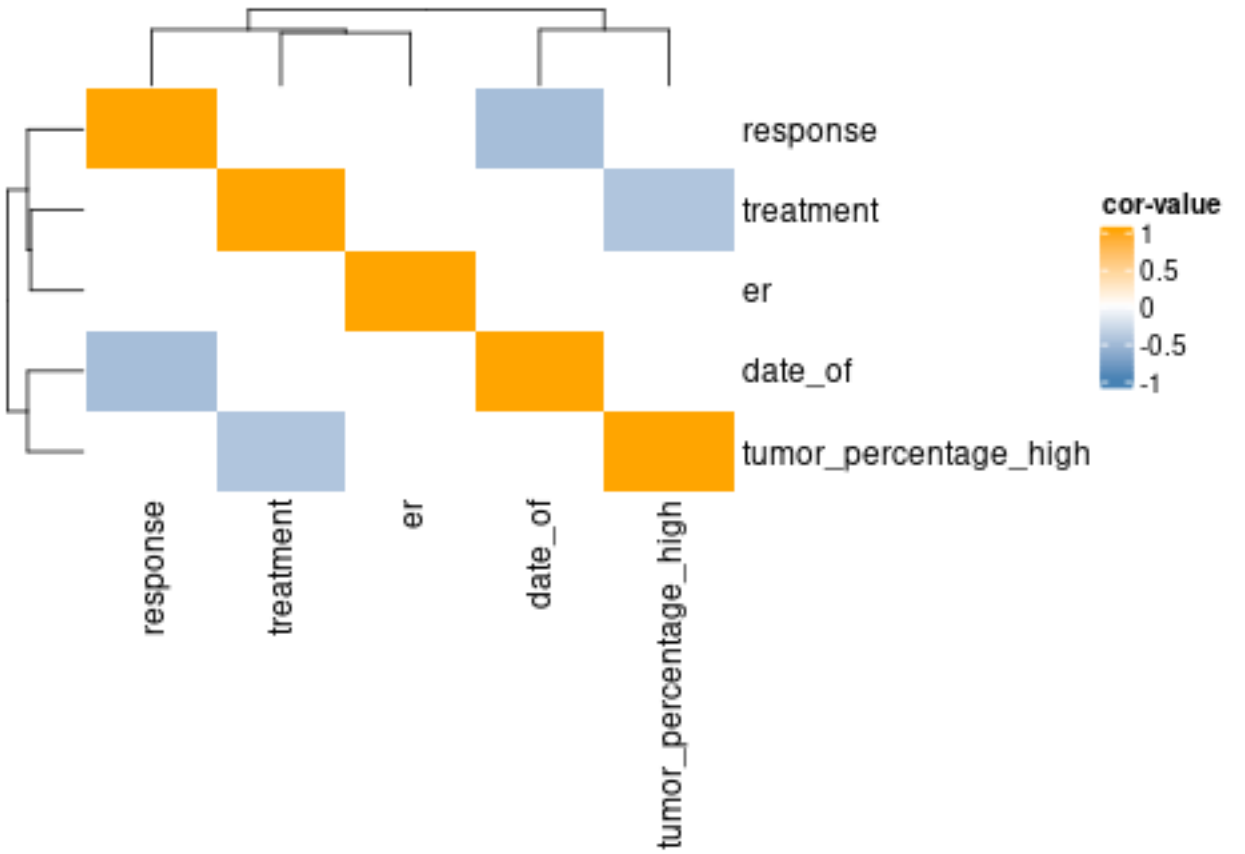



```
mdata %>% ggplot(aes(tumor_percentage_high, fill = response)) + geom_bar(position = "dodge2")
```

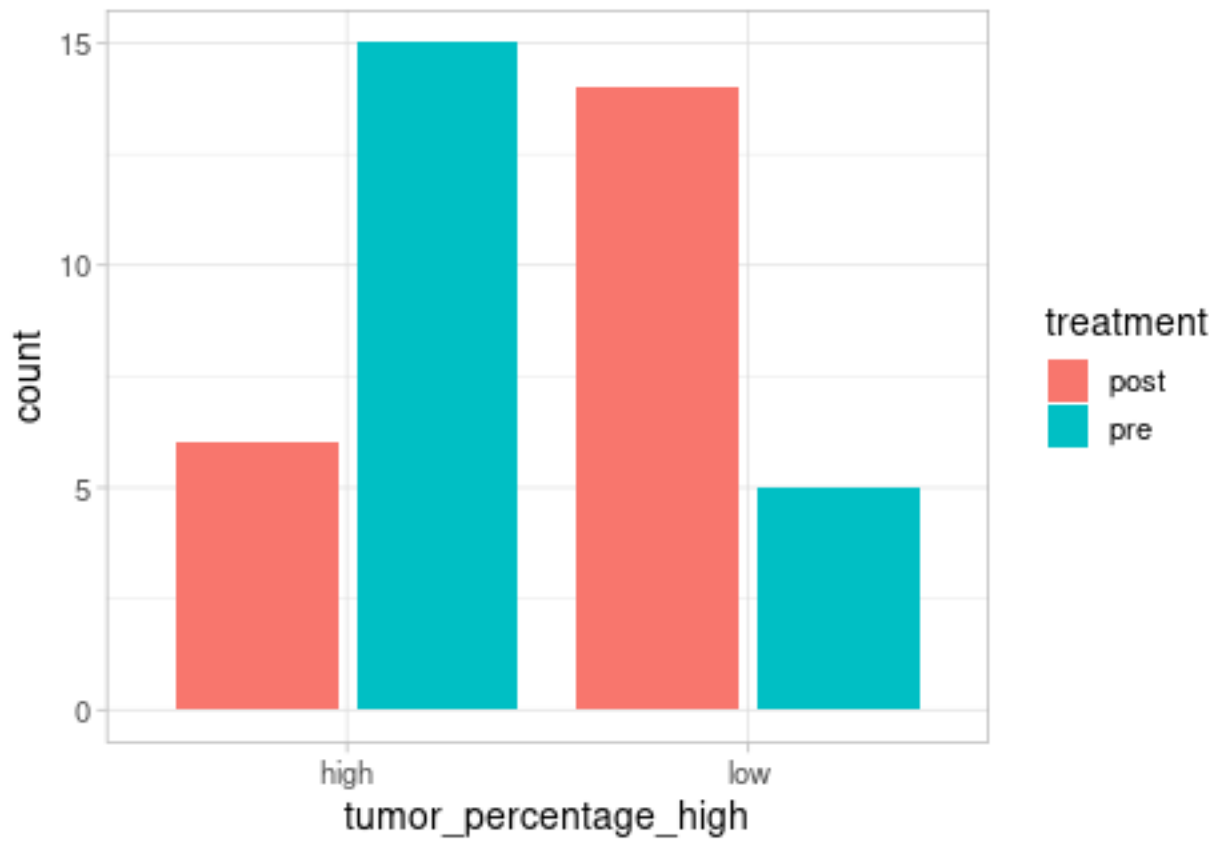


Covariates correlation with metrics

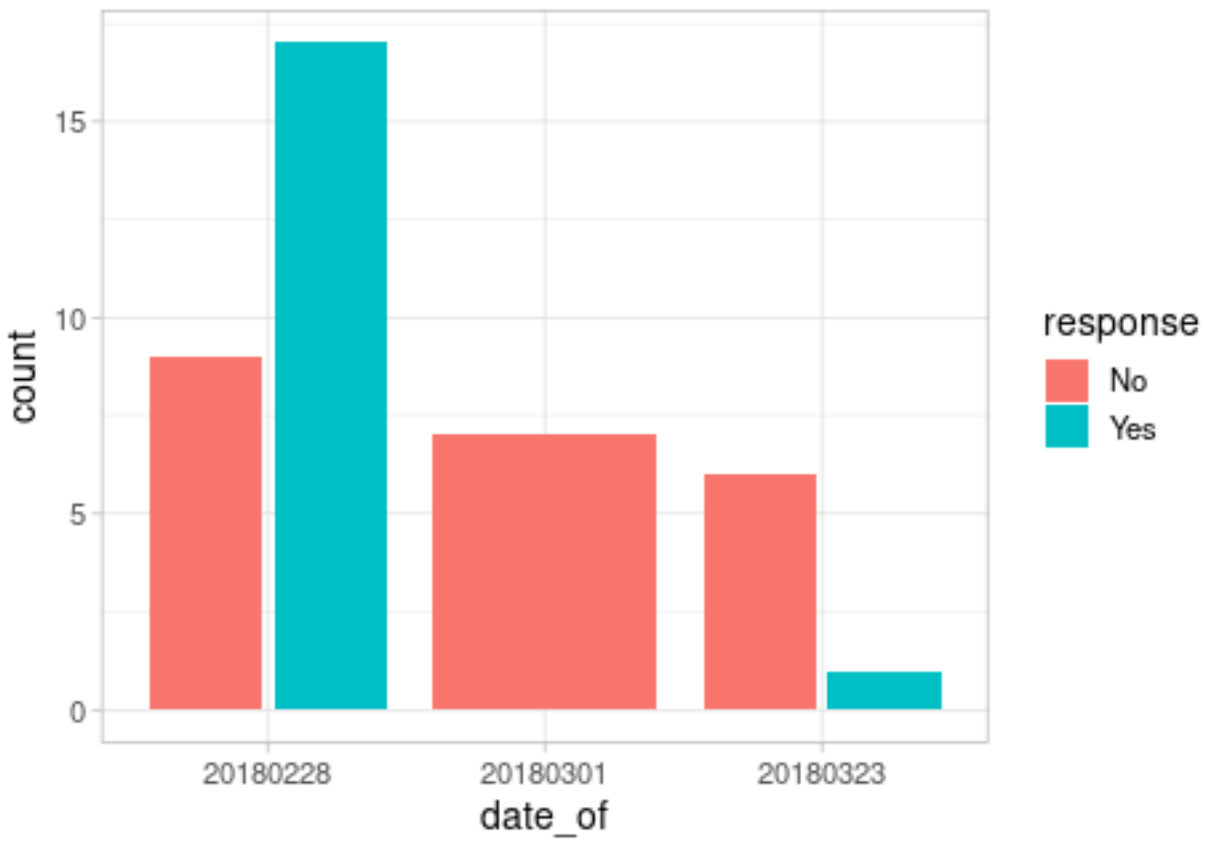
```
cor <- degCorCov(mdata)
```




```
mdata %>% ggplot(aes(tumor_percentage_high, fill = treatment)) + geom_bar(position = "dodge2")
```




```
mdata %>% ggplot(aes(date_of, fill = response)) + geom_bar(position = "dodge2")
```



Sample-level QC analysis

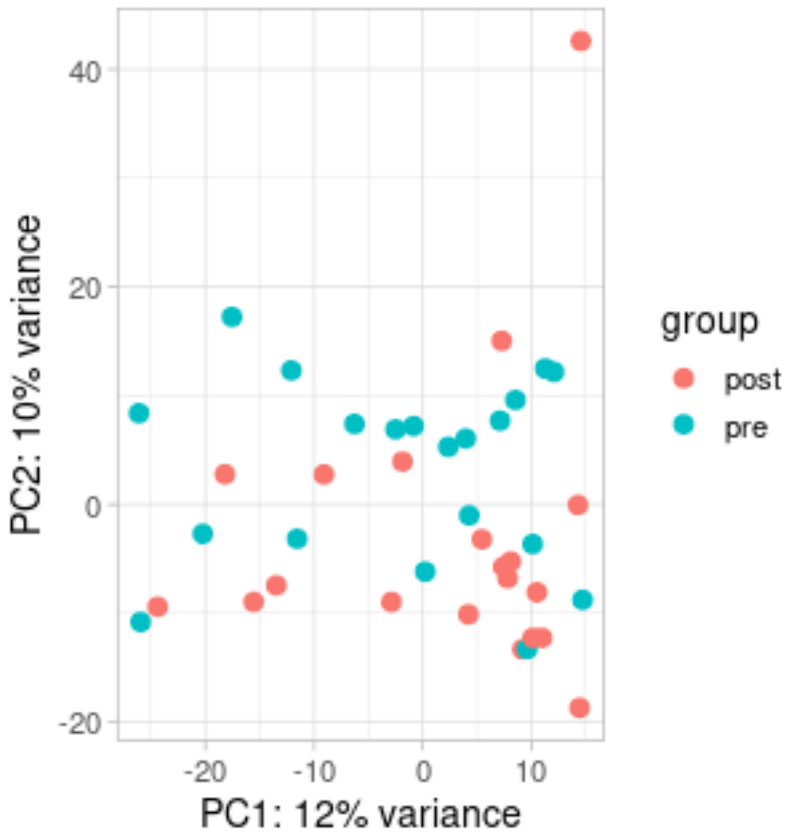
```
### Transform counts for data visualization (unsupervised analysis)
rld_file <- "data/rld.RDS"
if (file.exists(rld_file)){
    rld <- readRDS(rld_file)
}else{
    rld <- rlog(dds, blind = TRUE)
    saveRDS(rld, rld_file)
}
class(rld) # what type of object is this
```

```
## [1] "DESeqTransform"
## attr(,"package")
## [1] "DESeq2"
```

```
# we also need just a matrix of transformed counts
rld_mat <- assay(rld)
```

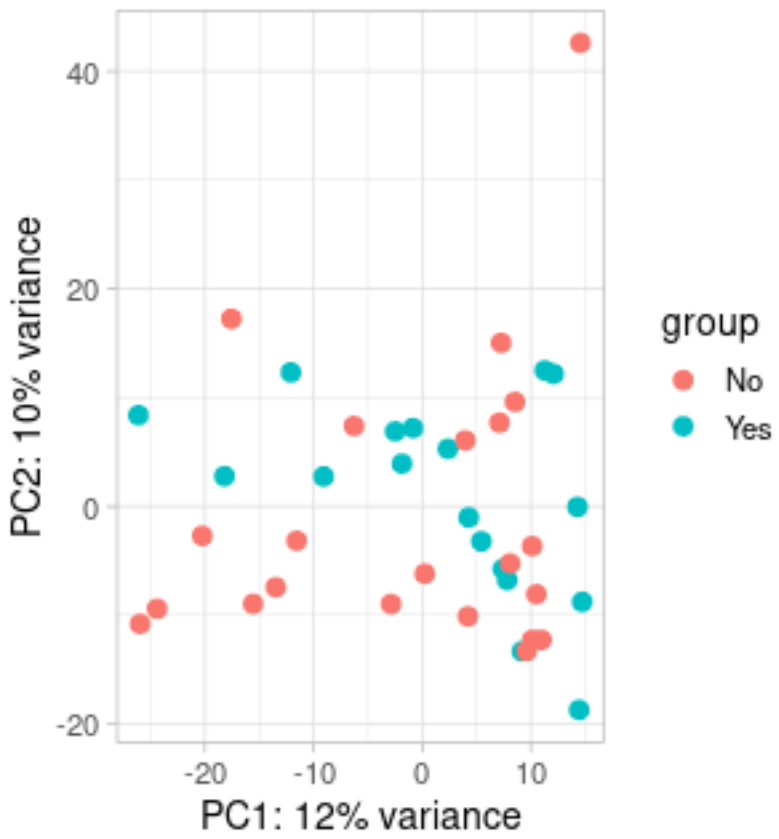
PCA - treatment

```
# Use the DESeq2 function
plotPCA(rld, intgroup = c("treatment"))
```



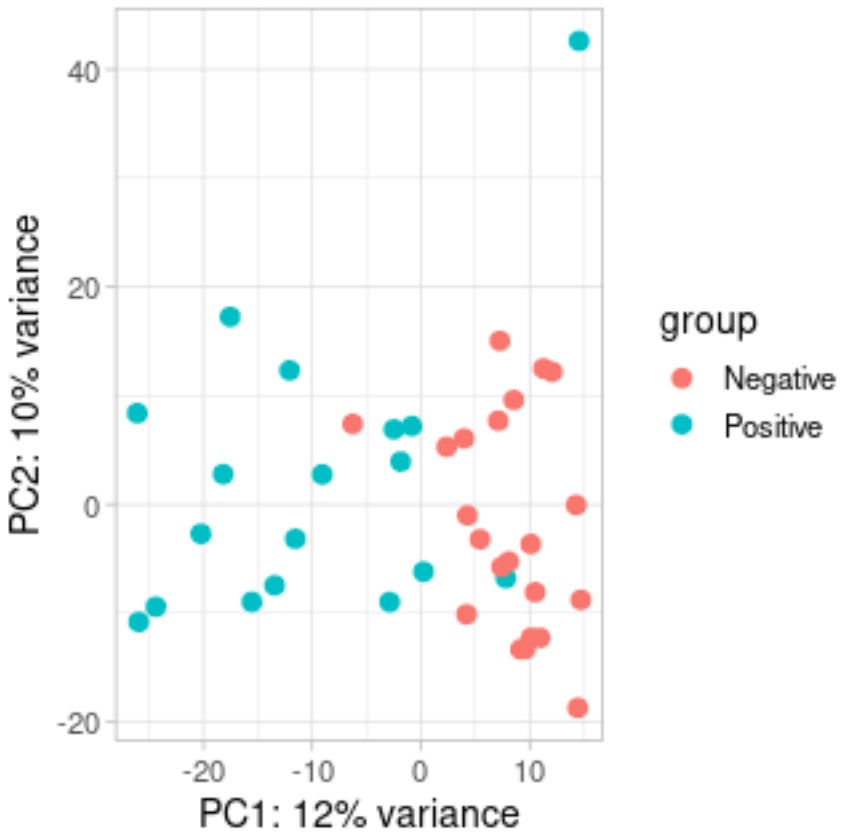
PCA - response

```
# Use the DESeq2 function
plotPCA(rld, intgroup = c("response"))
```



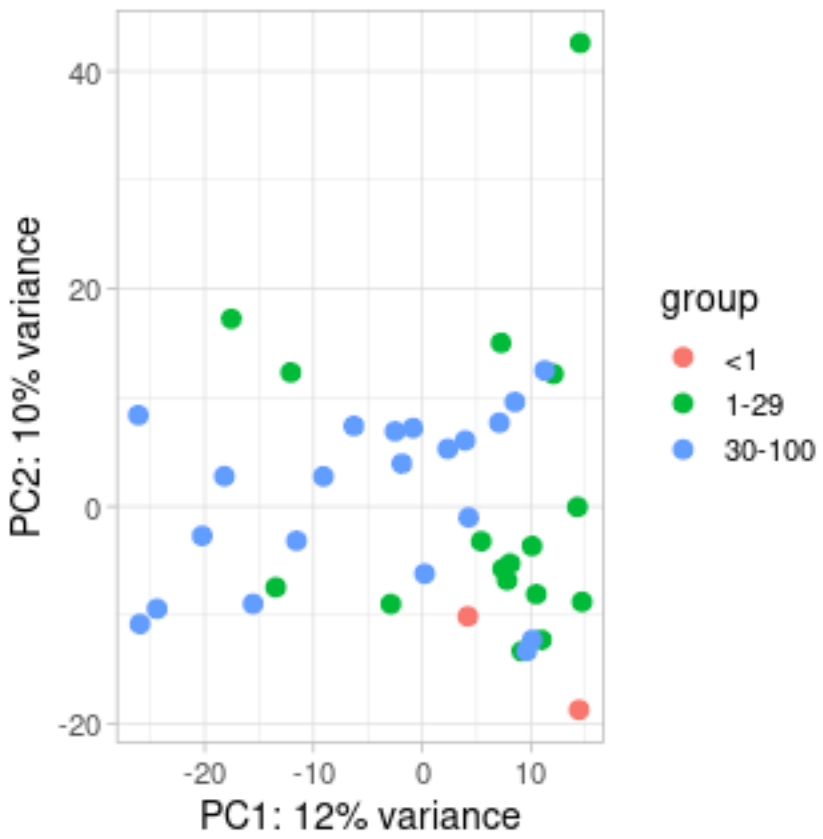
PCA - ER

```
# Use the DESeq2 function
plotPCA(rld, intgroup = c("er"))
```



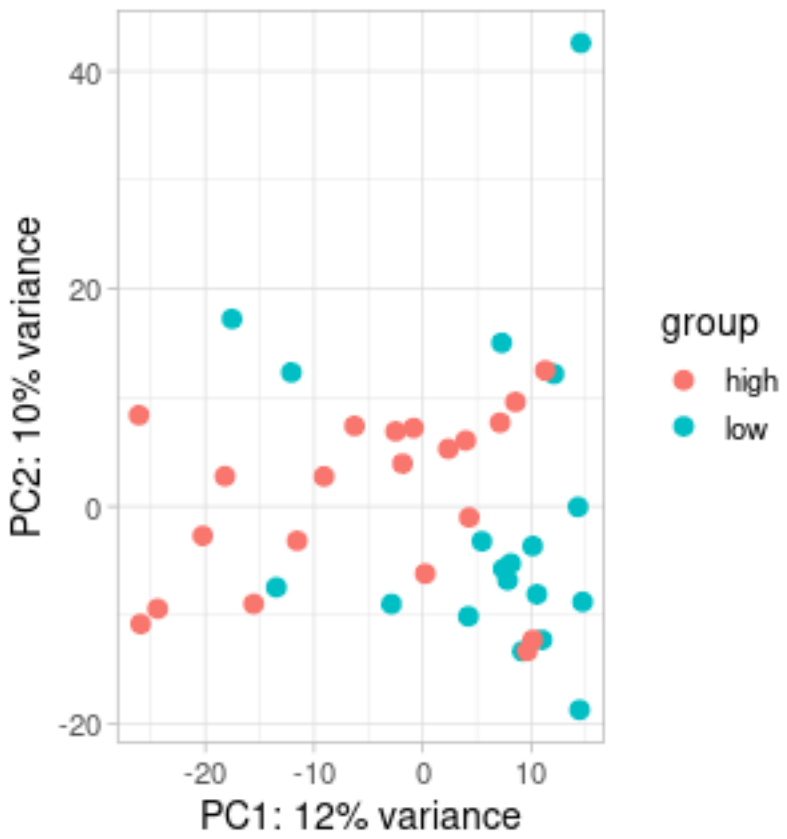
PCA - tumor_percentage

```
# Use the DESeq2 function
plotPCA(rld, intgroup = c("tumor_percentage"))
```



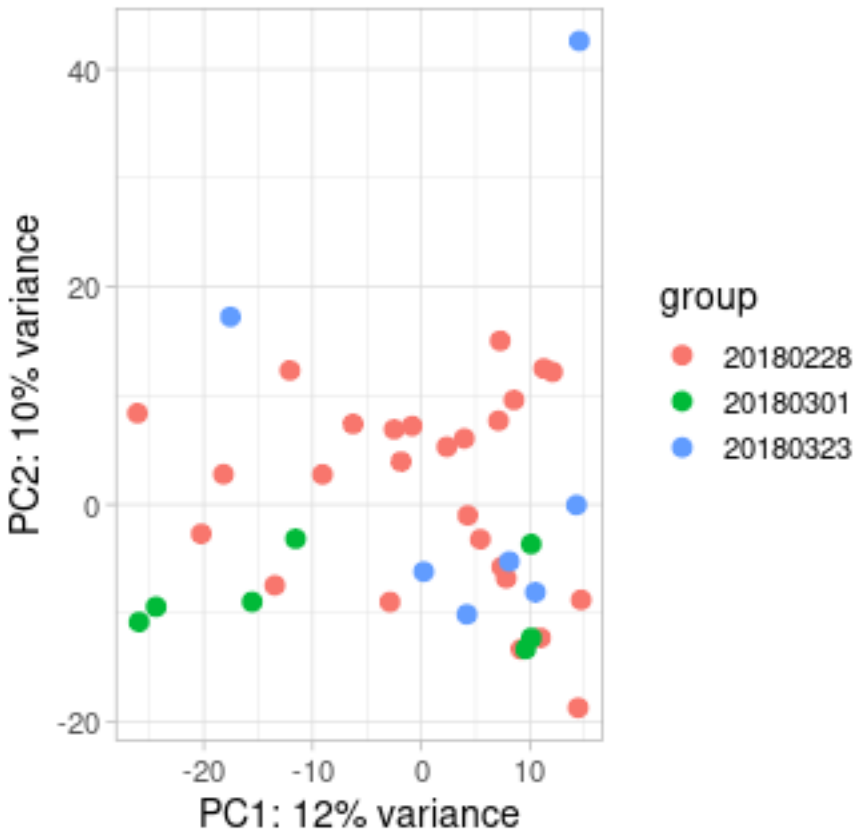
PCA - tumor_percentage_high

```
# Use the DESeq2 function
plotPCA(rld, intgroup = c("tumor_percentage_high"))
```



PCA - date_of

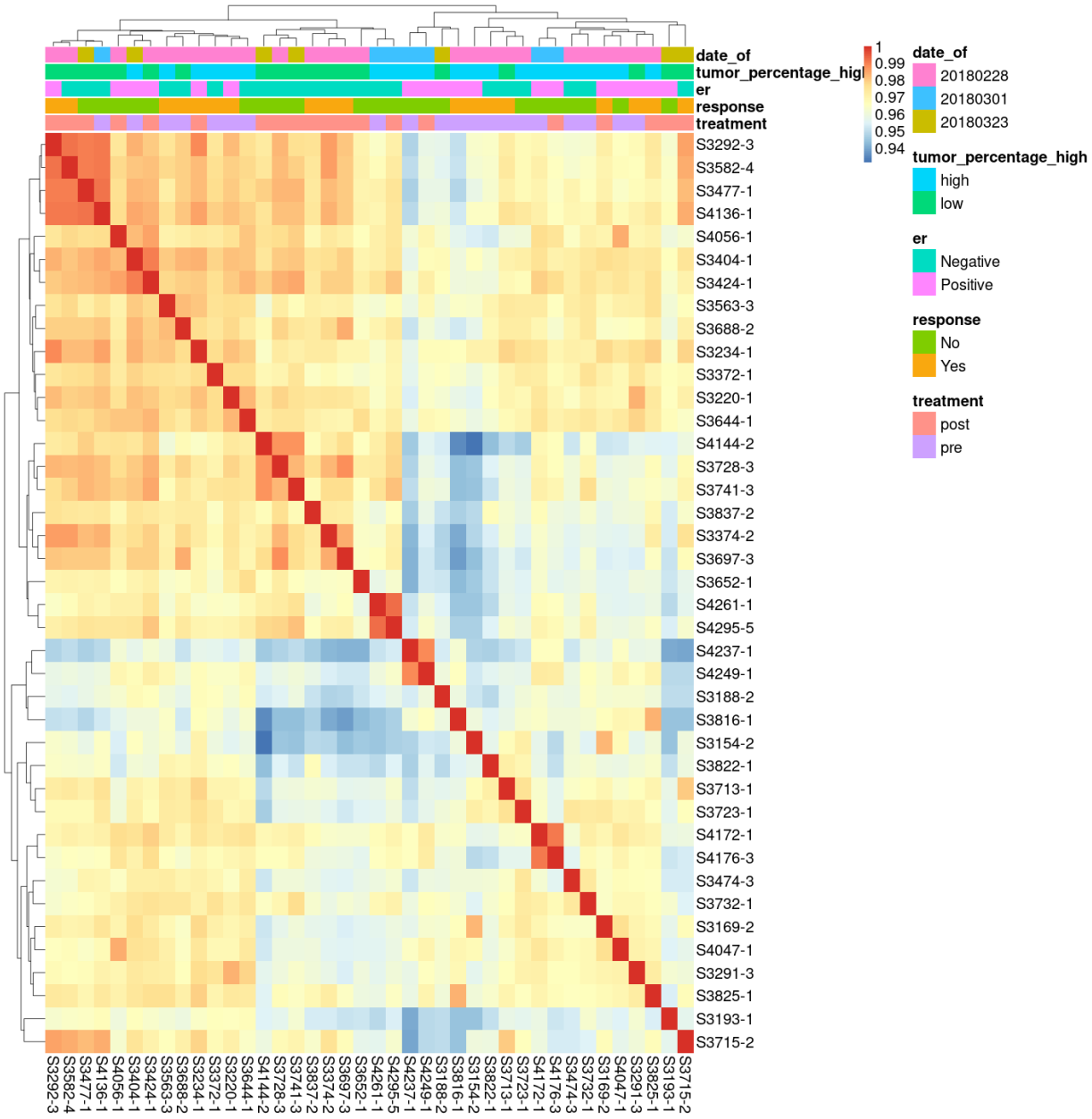
```
# Use the DESeq2 function
plotPCA(rld, intgroup = c("date_of"))
```



Inter-correlation analysis

Without study_id

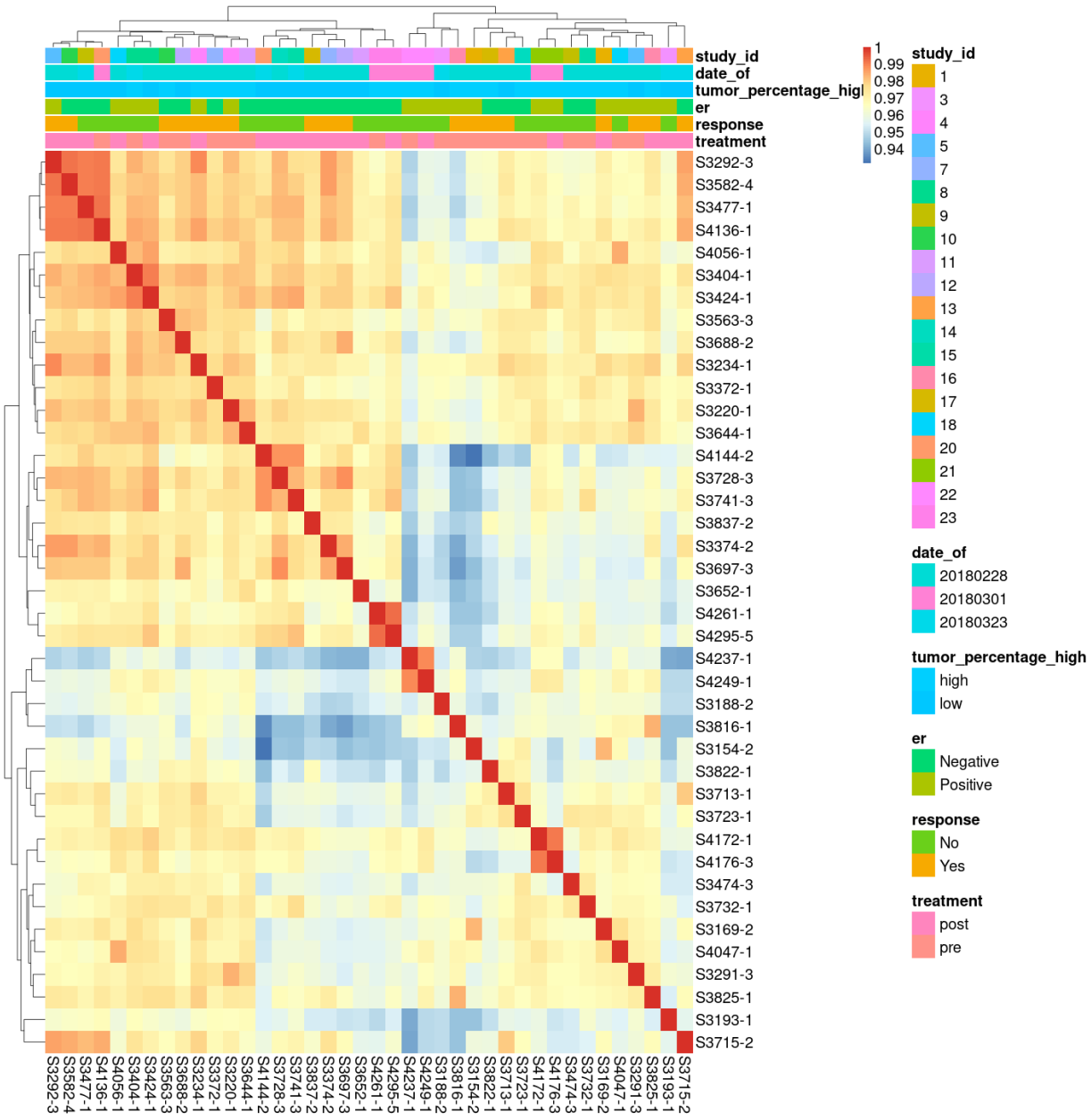
```
# Correlation matrix
rld_cor <- cor(rld_mat)

meta$study_id <- as.factor(meta$study_id)
# Create annotation file for samples
annotation <- meta[, c("treatment", "response", "er", "tumor_percentage_high", "date_of")]

# Change colors
heat.colors <- brewer.pal(6, "Blues")

# Plot heatmap
pheatmap(rld_cor,
         annotation = annotation,
         border = NA,
         fontsize = 20)
```



With study_id

```
# Correlation matrix
rld_cor <- cor(rld_mat)

meta$study_id <- as.factor(meta$study_id)
# Create annotation file for samples
annotation <- meta[, c("treatment", "response", "er", "tumor_percentage_high", "date_of", "study_id")]

# Change colors
heat.colors <- brewer.pal(6, "Blues")

# Plot heatmap
```

```
pheatmap(rld_cor,
         annotation = annotation,
         border = NA,
         fontsize = 20)
```



Treatment Post vs Pre - see Table3

```
# Get results for treatment
contrast <- c("treatment", "post", "pre")
resTreatment <- results(dds, contrast = contrast, alpha = 0.05)
length(which(resTreatment$padj < 0.05))
```

```
## [1] 171
```

```r
# Add annotations
resTreatment_tb <- resTreatment %>%
  data.frame() %>%
  rownames_to_column(var = "gene") %>%
  as_tibble() %>%
  left_join(gene_symbol, by = c("gene" = "gene_id"))

resTreatment_tb_significant <- dplyr::filter(resTreatment_tb, padj < 0.05)

samples_pre <- meta %>% dplyr::filter(treatment == "pre") %>% row.names()

counts_pre <- txi$abundance %>%
  as.data.frame() %>%
  dplyr::select(one_of(samples_pre)) %>%
  rowMeans() %>%
  as.data.frame() %>%
  rownames_to_column(var = "ensembl_gene_id")

colnames(counts_pre) <- c("ensembl_gene_id", "pre_expression_mean_tpm")

samples_post <- meta %>% dplyr::filter(treatment == "post") %>% row.names()

counts_post <- txi$abundance %>%
        as.data.frame() %>%
        dplyr::select(one_of(samples_post)) %>%
        rowMeans() %>%
        as.data.frame() %>%
        rownames_to_column(var = "ensembl_gene_id")
colnames(counts_post) <- c("ensembl_gene_id", "post_expression_mean_tpm")

counts_post <- counts_post %>% left_join(counts_pre, by = c("ensembl_gene_id" = "ensembl_gene_id"))

resTreatment_tb_significant <- resTreatment_tb_significant %>%
         left_join(counts_post, by = c("gene" = "ensembl_gene_id"))

write_csv(resTreatment_tb_significant,
          "tables/T3.DE_treatment.csv")

# Separate into up and down-regulated gene sets
sigTreatment_up <- rownames(resTreatment)[which(resTreatment$padj < 0.01 & resTreatment$log2FoldChange >
sigTreatment_down <- rownames(resTreatment)[which(resTreatment$padj < 0.01 & resTreatment$log2FoldChang
```

Response Yes vs No - see Table4

```r
# Get results for rescue vs wt
contrast <- c("response", "Yes", "No")
resResponse <- results(dds, contrast = contrast, alpha = 0.05)
length(which(resResponse$padj < 0.05))
```

```
## [1] 1659
```

```r
# Add annotations
resResponse_tb <- resResponse %>%
```

```
  data.frame() %>%
  rownames_to_column(var = "gene") %>%
  as_tibble() %>%
  left_join(gene_symbol, by = c("gene" = "gene_id"))

resResponse_tb_significant <- dplyr::filter(resResponse_tb, padj < 0.05)

samples_no <- meta %>% dplyr::filter(response == "No") %>% row.names()

counts_no <- txi$abundance %>%
                as.data.frame() %>%
                dplyr::select(any_of(samples_no)) %>%
                rowSums() %>%
                as.data.frame() %>%
                rownames_to_column(var = "ensembl_gene_id")
colnames(counts_no) <- c("ensembl_gene_id", "no_expression_mean_tpm")

samples_yes <- meta %>% dplyr::filter(response == "Yes") %>% row.names()

counts_yes <- txi$abundance %>%
                as.data.frame() %>%
                dplyr::select(any_of(samples_yes)) %>%
                rowSums() %>%
                as.data.frame() %>%
                rownames_to_column(var = "ensembl_gene_id")

colnames(counts_yes) <- c("ensembl_gene_id", "yes_expression_mean_tpm")

counts_yes <-counts_yes %>%
            left_join(counts_no,
                      by = c("ensembl_gene_id" = "ensembl_gene_id"))

resResponse_tb_significant <- resResponse_tb_significant %>%
          left_join(counts_yes, by = c("gene" = "ensembl_gene_id"))


write_csv(resResponse_tb_significant,
          "tables/T4.DE_response.csv")

# Separate into up and down-regulated gene sets
sigResponse_up <- rownames(resResponse)[which(resResponse$padj < 0.01 & resResponse$log2FoldChange > 0)]
sigResponse_down <- rownames(resResponse)[which(resResponse$padj < 0.01 & resResponse$log2FoldChange < 0
```

ER : Positive vs Negative - Table 5

```
contrast <- c("er", "Positive", "Negative")
resER <- results(dds, contrast = contrast, alpha = 0.05)
length(which(resER$padj < 0.05))
```

```
## [1] 1806
```

```
# Add annotations
resER_tb <- resER %>%
```

```
  data.frame() %>%
  rownames_to_column(var = "gene") %>%
  as_tibble() %>%
  left_join(gene_symbol, by = c("gene" = "gene_id"))

resER_tb_significant <- dplyr::filter(resER_tb, padj < 0.05)

samples_pos <- meta %>% dplyr::filter(er == "Positive") %>% row.names()

counts_pos <- txi$abundance %>%
                as.data.frame() %>%
                dplyr::select(any_of(samples_pos)) %>%
                rowMeans() %>%
                as.data.frame() %>%
                rownames_to_column(var = "ensembl_gene_id")
colnames(counts_pos) <- c("ensembl_gene_id", "Positive_expression_mean_tpm")

samples_neg <- meta %>% dplyr::filter(er == "Negative") %>% row.names()

counts_neg <- txi$abundance %>%
                as.data.frame() %>%
                dplyr::select(any_of(samples_neg)) %>%
                rowMeans() %>%
                as.data.frame() %>%
                rownames_to_column(var = "ensembl_gene_id")

colnames(counts_neg) <- c("ensembl_gene_id", "Negative_expression_mean_tpm")

counts_pos <-counts_pos %>%
            left_join(counts_neg,
                      by = c("ensembl_gene_id" = "ensembl_gene_id"))

resER_tb_significant <- resER_tb_significant %>%
          left_join(counts_pos, by = c("gene" = "ensembl_gene_id"))


write_csv(resER_tb_significant,
          "tables/T5.DE_ER.csv")

# Separate into up and down-regulated gene sets
sigER_up <- rownames(resER)[which(resER$padj < 0.01 & resER$log2FoldChange > 0)]
sigER_down <- rownames(resER)[which(resER$padj < 0.01 & resER$log2FoldChange < 0)]
```

tumor_percentage_high : High vs Low - Table 6

```
contrast <- c("tumor_percentage_high", "high", "low")
resTP <- results(dds, contrast = contrast, alpha = 0.05)
length(which(resTP$padj < 0.05))
```

```
## [1] 7
```

```
# Add annotations
resTP_tb <- resTP %>%
```

```
  data.frame() %>%
  rownames_to_column(var = "gene") %>%
  as_tibble() %>%
  left_join(gene_symbol, by = c("gene" = "gene_id"))

resTP_tb_significant <- dplyr::filter(resTP_tb, padj < 0.05)

samples_high <- meta %>% dplyr::filter(tumor_percentage_high == "high") %>% row.names()

counts_high <- txi$abundance %>%
                as.data.frame() %>%
                dplyr::select(any_of(samples_high)) %>%
                rowMeans() %>%
                as.data.frame() %>%
                rownames_to_column(var = "ensembl_gene_id")
colnames(counts_high) <- c("ensembl_gene_id", "High_expression_mean_tpm")

samples_low <- meta %>% dplyr::filter(tumor_percentage_high == "low") %>% row.names()

counts_low <- txi$abundance %>%
                as.data.frame() %>%
                dplyr::select(any_of(samples_low)) %>%
                rowMeans() %>%
                as.data.frame() %>%
                rownames_to_column(var = "ensembl_gene_id")

colnames(counts_low) <- c("ensembl_gene_id", "Low_expression_mean_tpm")

counts_high <-counts_high %>%
            left_join(counts_low,
                      by = c("ensembl_gene_id" = "ensembl_gene_id"))

resTP_tb_significant <- resTP_tb_significant %>%
          left_join(counts_high, by = c("gene" = "ensembl_gene_id"))


write_csv(resTP_tb_significant,
          "tables/T6.DE_tumor_percentage_high.csv")

# Separate into up and doum-regulated gene sets
sigTP_up <- rownames(resTP)[which(resTP$padj < 0.01 & resTP$log2FoldChange > 0)]
sigTP_down <- rownames(resTP)[which(resTP$padj < 0.01 & resTP$log2FoldChange < 0)]
```

date_of: 20180323 vs 20180228 - Table 7

```
contrast <- c("date_of", "20180323", "20180228")
resDO <- results(dds, contrast = contrast, alpha = 0.05)
length(which(resDO$padj < 0.05))
```

```
## [1] 157
```

```
# Add annotations
resDO_tb <- resDO %>%
```

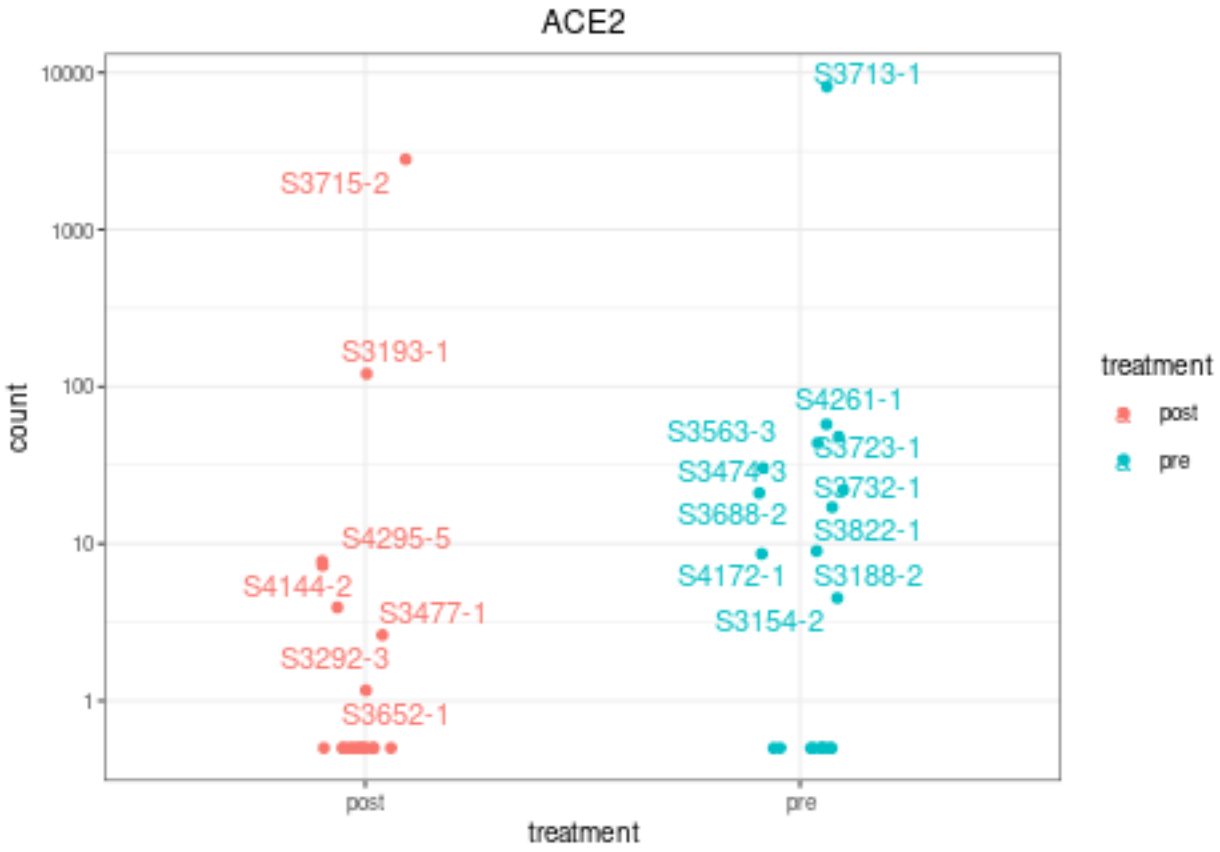
```
  data.frame() %>%
  rownames_to_column(var = "gene") %>%
  as_tibble() %>%
  left_join(gene_symbol, by = c("gene" = "gene_id"))

resDO_tb_significant <- dplyr::filter(resDO_tb, padj < 0.05)

samples_23 <- meta %>% dplyr::filter(date_of == "20180323") %>% row.names()

counts_23 <- txi$abundance %>%
                as.data.frame() %>%
                dplyr::select(any_of(samples_23)) %>%
                rowMeans() %>%
                as.data.frame() %>%
                rownames_to_column(var = "ensembl_gene_id")
colnames(counts_23) <- c("ensembl_gene_id", "20180323_expression_mean_tpm")

samples_28 <- meta %>% dplyr::filter(date_of == "20180228") %>% row.names()

counts_28 <- txi$abundance %>%
                as.data.frame() %>%
                dplyr::select(any_of(samples_28)) %>%
                rowMeans() %>%
                as.data.frame() %>%
                rownames_to_column(var = "ensembl_gene_id")

colnames(counts_28) <- c("ensembl_gene_id", "20180228_expression_mean_tpm")

counts_23 <-counts_23 %>%
            left_join(counts_28,
                      by = c("ensembl_gene_id" = "ensembl_gene_id"))

resDO_tb_significant <- resDO_tb_significant %>%
          left_join(counts_23, by = c("gene" = "ensembl_gene_id"))


write_csv(resDO_tb_significant,
          "tables/T7.DE_dete_of.csv")

# Separate into up and down-regulated gene sets
sigDO_up <- rownames(resDO)[which(resDO$padj < 0.01 & resDO$log2FoldChange > 0)]
sigDO_down <- rownames(resDO)[which(resDO$padj < 0.01 & resDO$log2FoldChange < 0)]
```

Visualization

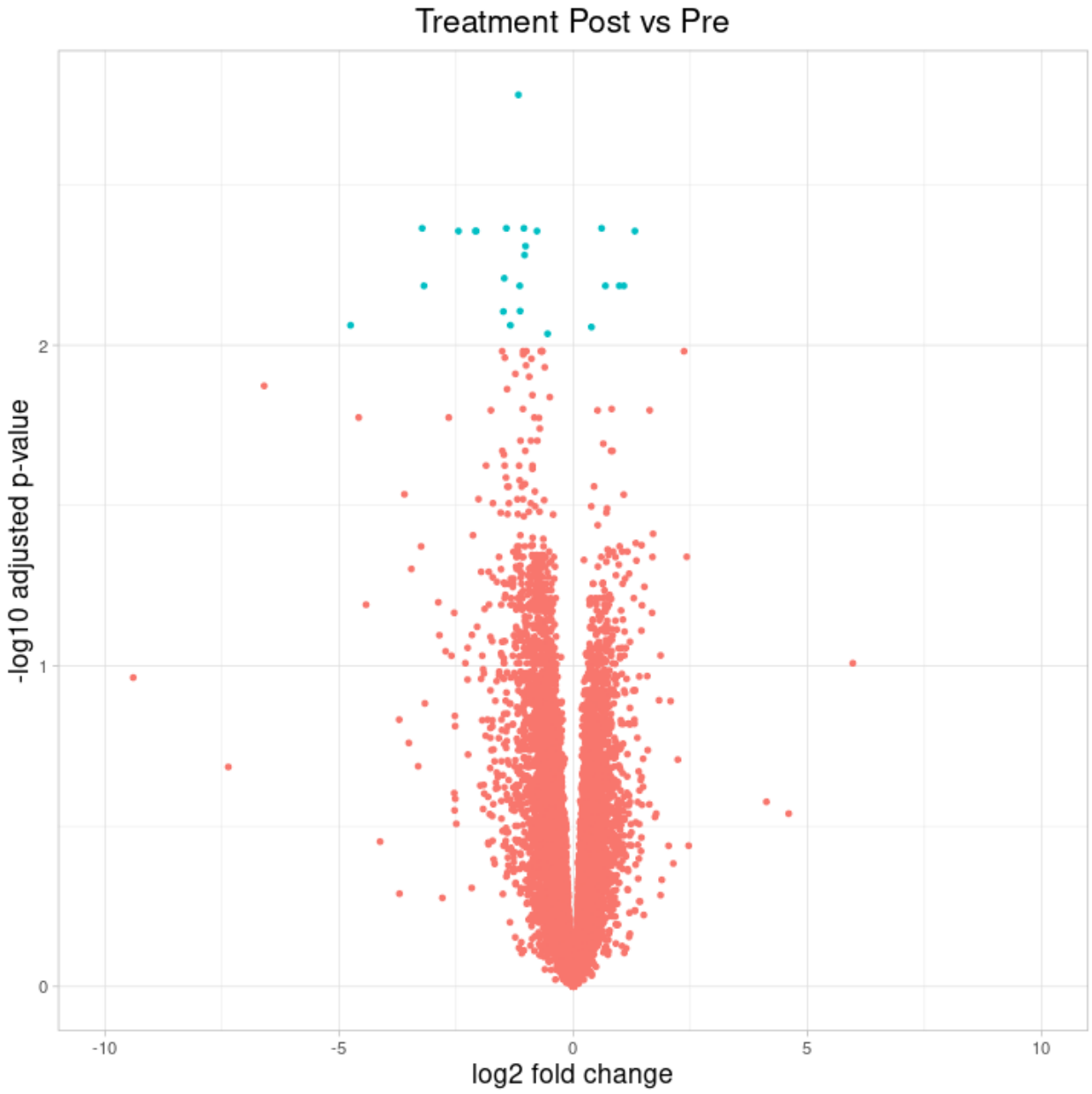
Gene example

```
d <- plotCounts(dds,
                gene = "ENSG00000130234",
                intgroup = "treatment",
                returnData = TRUE)

ggplot(d, aes(x = treatment, y = count, color = treatment)) +
```

```
    geom_point(position = position_jitter(w = 0.1, h = 0)) +
    geom_text_repel(aes(label = rownames(d))) +
    theme_bw(base_size = 10) +
    ggtitle("ACE2") +
    theme(plot.title = element_text(hjust = 0.5)) +
    scale_y_log10()
```

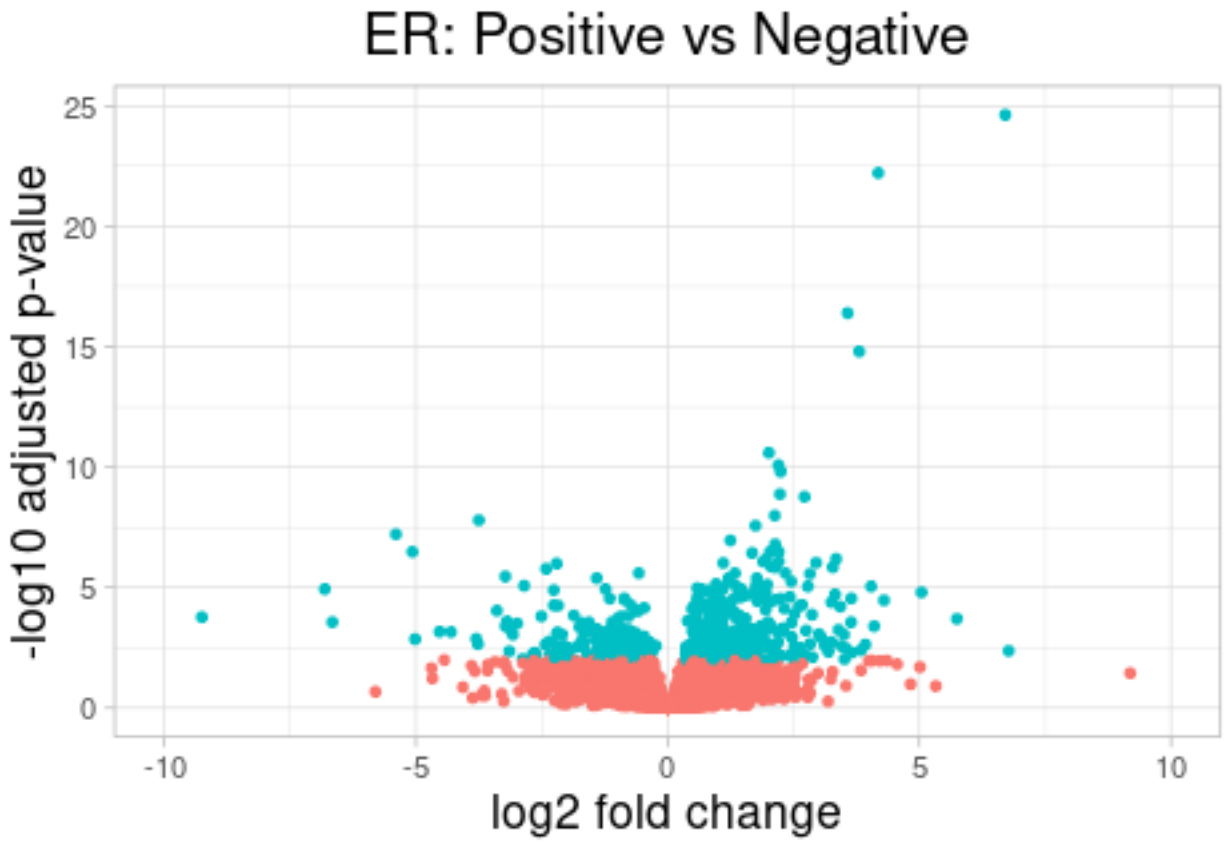



```
# Add a column for significant genes
resTreatment_tb <- resTreatment_tb %>% mutate(threshold = padj < 0.01)

## Volcano plot
ggplot(resTreatment_tb) +
  geom_point(aes(x = log2FoldChange, y = -log10(padj), colour = threshold)) +
  ggtitle("Treatment Post vs Pre") +
  xlab("log2 fold change") +
  ylab("-log10 adjusted p-value") +
  scale_x_continuous(limits = c(-10,10)) +
  theme(legend.position = "none",
        plot.title = element_text(size = rel(1.5), hjust = 0.5),
        axis.title = element_text(size = rel(1.25)))
```

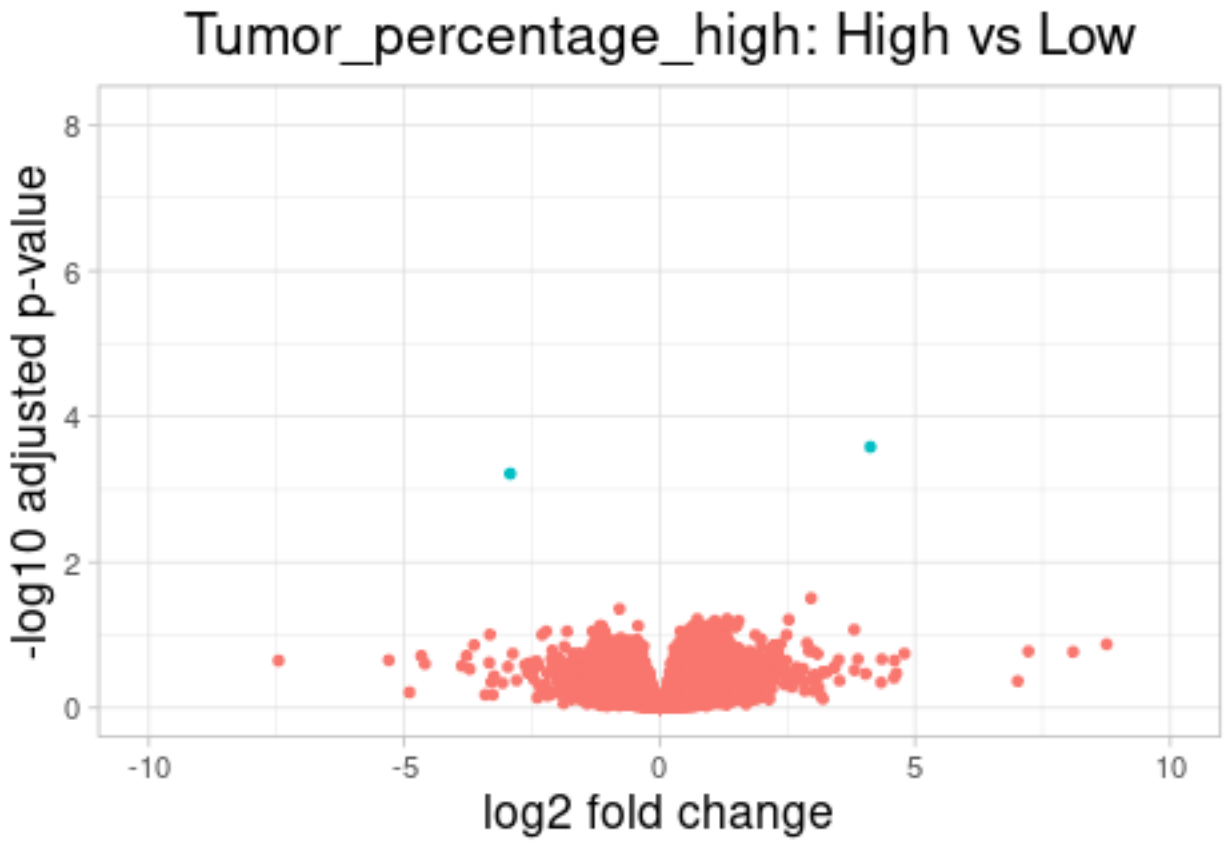



```
# Add a column for significant genes
resResponse_tb <- resResponse_tb %>% mutate(threshold = padj < 0.01)

ggplot(resResponse_tb) +
  geom_point(aes(x = log2FoldChange, y = -log10(padj), colour = threshold)) +
  ggtitle("Response Yes vs No") +
  xlab("log2 fold change") +
  ylab("-log10 adjusted p-value") +
  scale_x_continuous(limits = c(-10,10)) +
  scale_y_continuous(limits = c(0, 7.5))+
  theme(legend.position = "none",
        plot.title = element_text(size = rel(1.5), hjust = 0.5),
        axis.title = element_text(size = rel(1.25)))
```

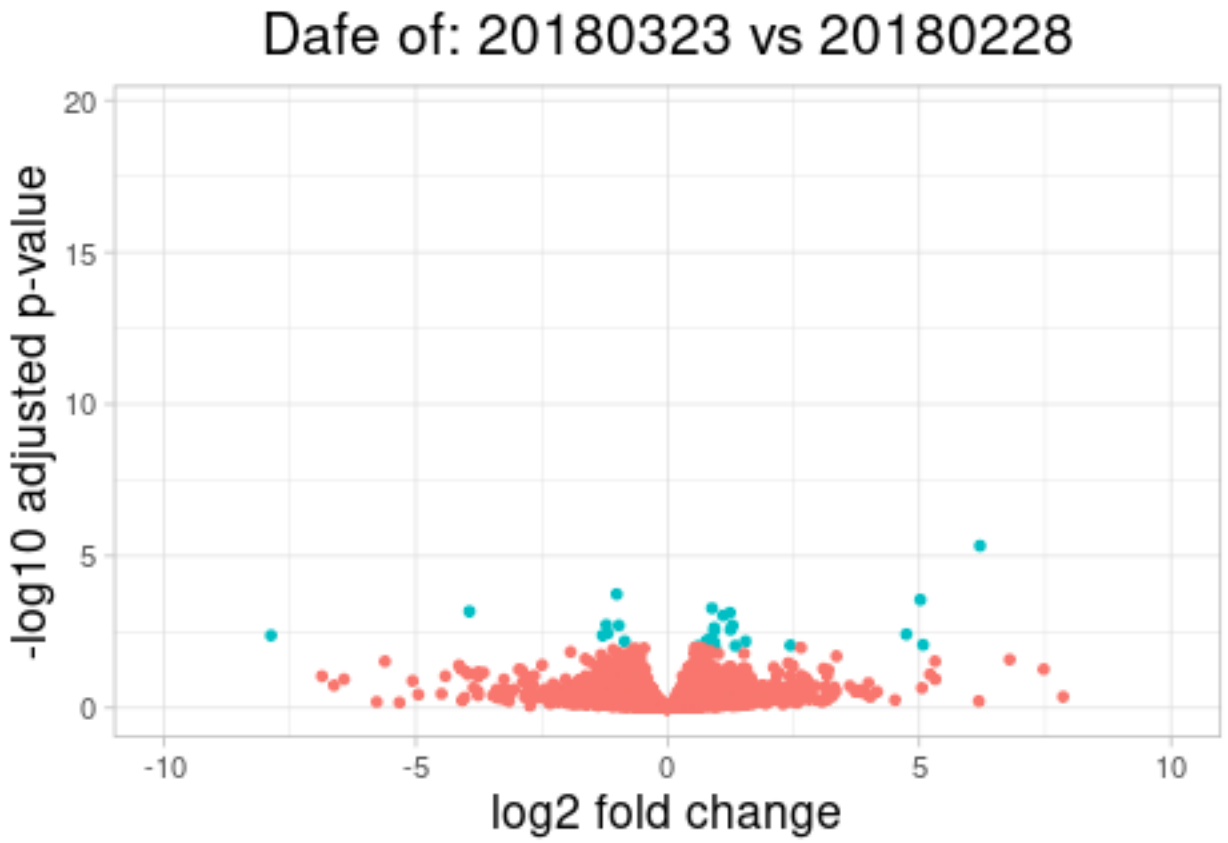

```
# Add a column for significant genes
resER_tb <- resER_tb %>% mutate(threshold = padj < 0.01)

ggplot(resER_tb) +
  geom_point(aes(x = log2FoldChange, y = -log10(padj), colour = threshold)) +
  ggtitle("ER: Positive vs Negative") +
  xlab("log2 fold change") +
  ylab("-log10 adjusted p-value") +
  scale_x_continuous(limits = c(-10,10)) +
  theme(legend.position = "none",
        plot.title = element_text(size = rel(1.5), hjust = 0.5),
        axis.title = element_text(size = rel(1.25)))
```




```
# Add a column for significant genes
resTP_tb <- resTP_tb %>% mutate(threshold = padj < 0.01)

ggplot(resTP_tb) +
  geom_point(aes(x = log2FoldChange, y = -log10(padj), colour = threshold)) +
  ggtitle("Tumor_percentage_high: High vs Low") +
  xlab("log2 fold change") +
  ylab("-log10 adjusted p-value") +
  scale_x_continuous(limits = c(-10,10)) +
  theme(legend.position = "none",
        plot.title = element_text(size = rel(1.5), hjust = 0.5),
        axis.title = element_text(size = rel(1.25)))
```



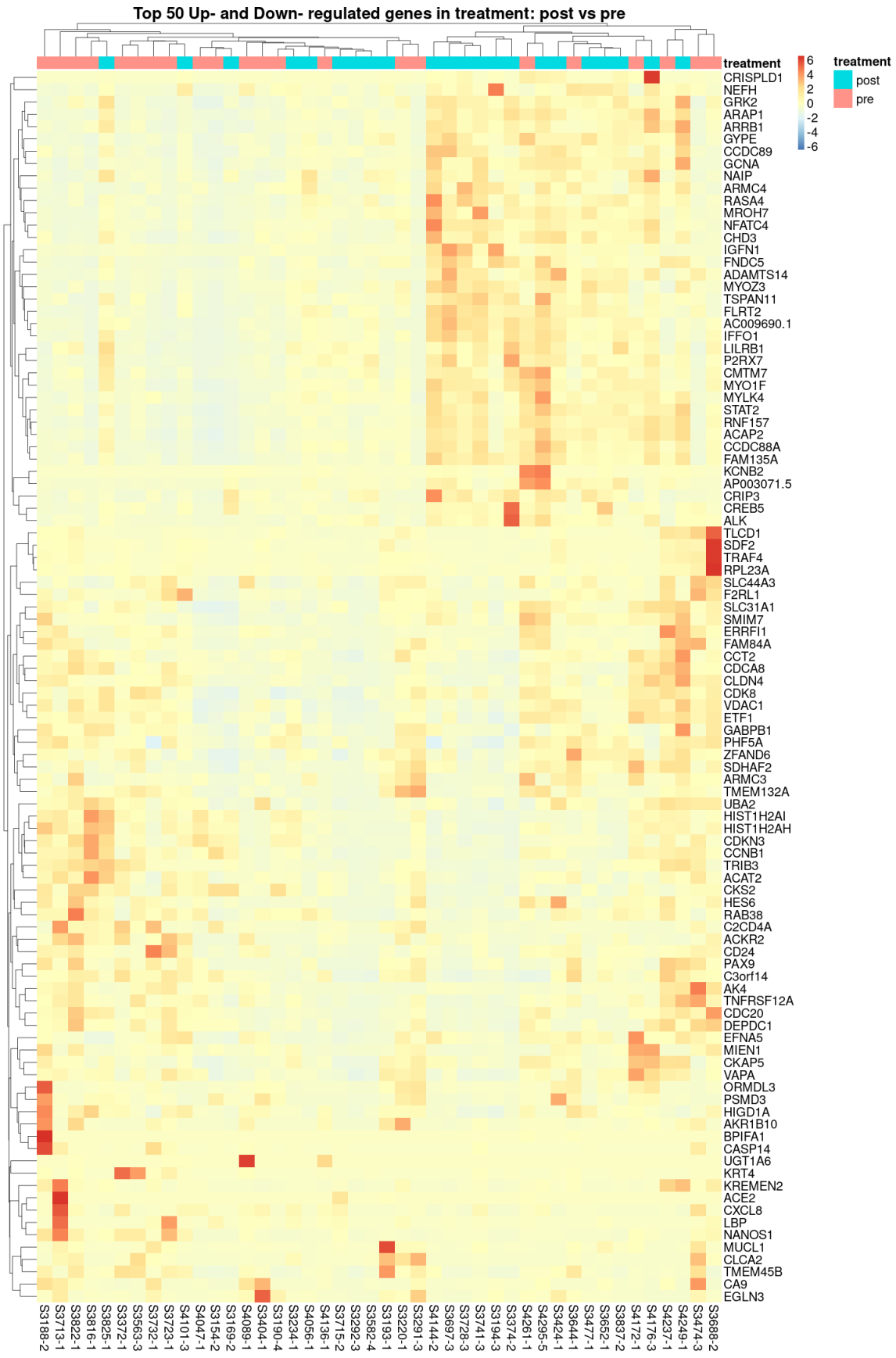

```
# Add a column for significant genes
resDO_tb <- resDO_tb %>% mutate(threshold = padj < 0.01)

ggplot(resDO_tb) +
  geom_point(aes(x = log2FoldChange, y = -log10(padj), colour = threshold)) +
  ggtitle("Dafe of: 20180323 vs 20180228") +
  xlab("log2 fold change") +
  ylab("-log10 adjusted p-value") +
  scale_x_continuous(limits = c(-10,10)) +
  theme(legend.position = "none",
        plot.title = element_text(size = rel(1.5), hjust = 0.5),
        axis.title = element_text(size = rel(1.25)))
```



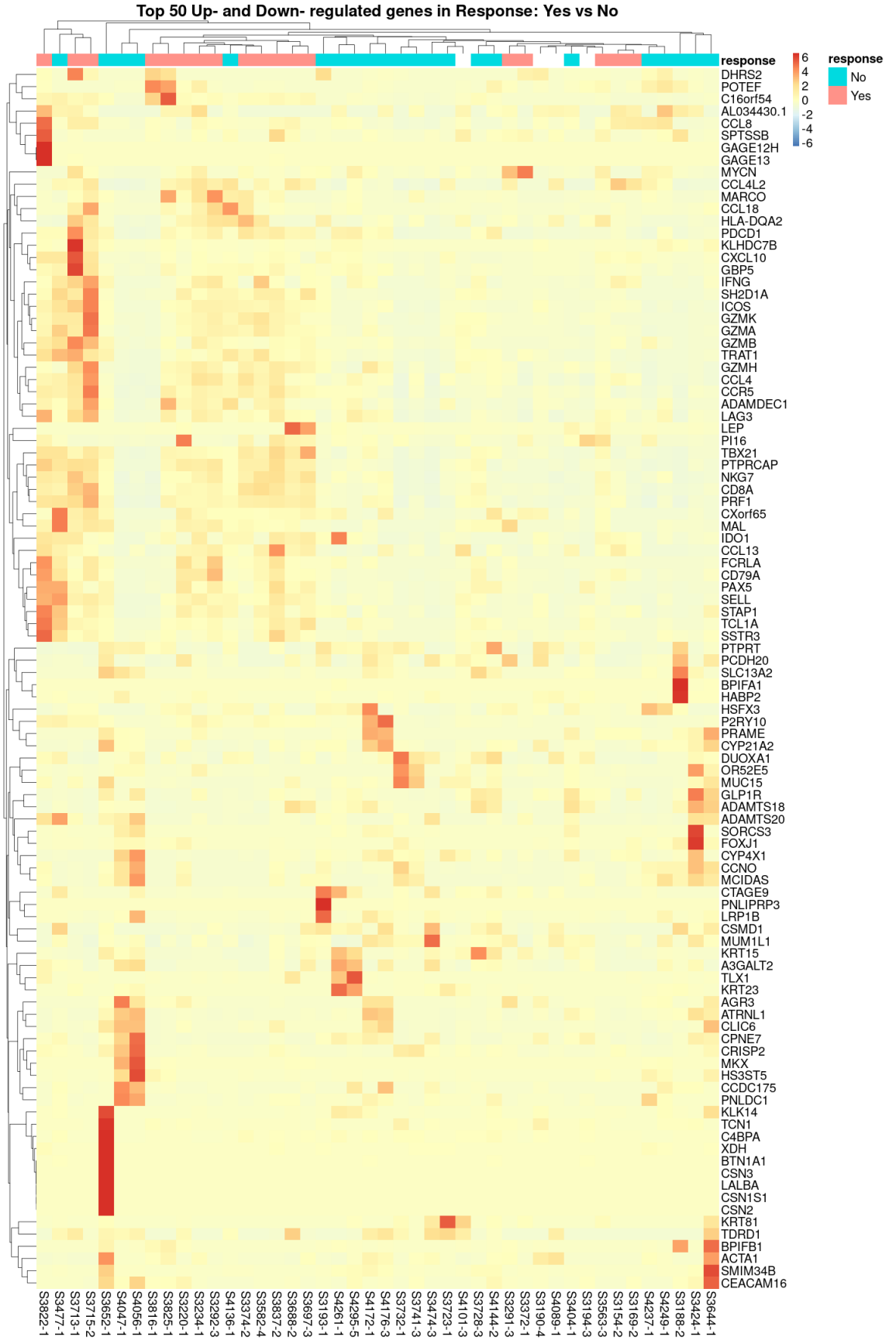
Heatmaps

```
# Create a matrix of normalized expression
sig_up <- resTreatment_tb_significant %>% arrange(-log2FoldChange) %>% head(50) %>% pull(gene)
sig_down <- resTreatment_tb_significant %>% arrange(log2FoldChange) %>% head(50) %>% pull(gene)
sig <- c(sig_up, sig_down)

row_annotation <- gene_symbol %>%
                    as_tibble() %>%
                    dplyr::filter(gene_id %in% sig)

plotmat <- txi$abundance[c(sig_up, sig_down),] %>% as.data.frame() %>%
          rownames_to_column(var = "ensembl_gene_id") %>%
          left_join(gene_symbol, by = c("ensembl_gene_id" = "gene_id")) %>%
          drop_na(symbol)

plotmat$ensembl_gene_id <- NULL

plotmat <- plotmat %>% column_to_rownames(var = "symbol") %>% as.matrix()

# Color palette
heat.colors <- brewer.pal(6, "YlOrRd")

# Plot heatmap
# color = heat.colors,
pheatmap(plotmat, scale = "row",
         show_rownames = TRUE,
```

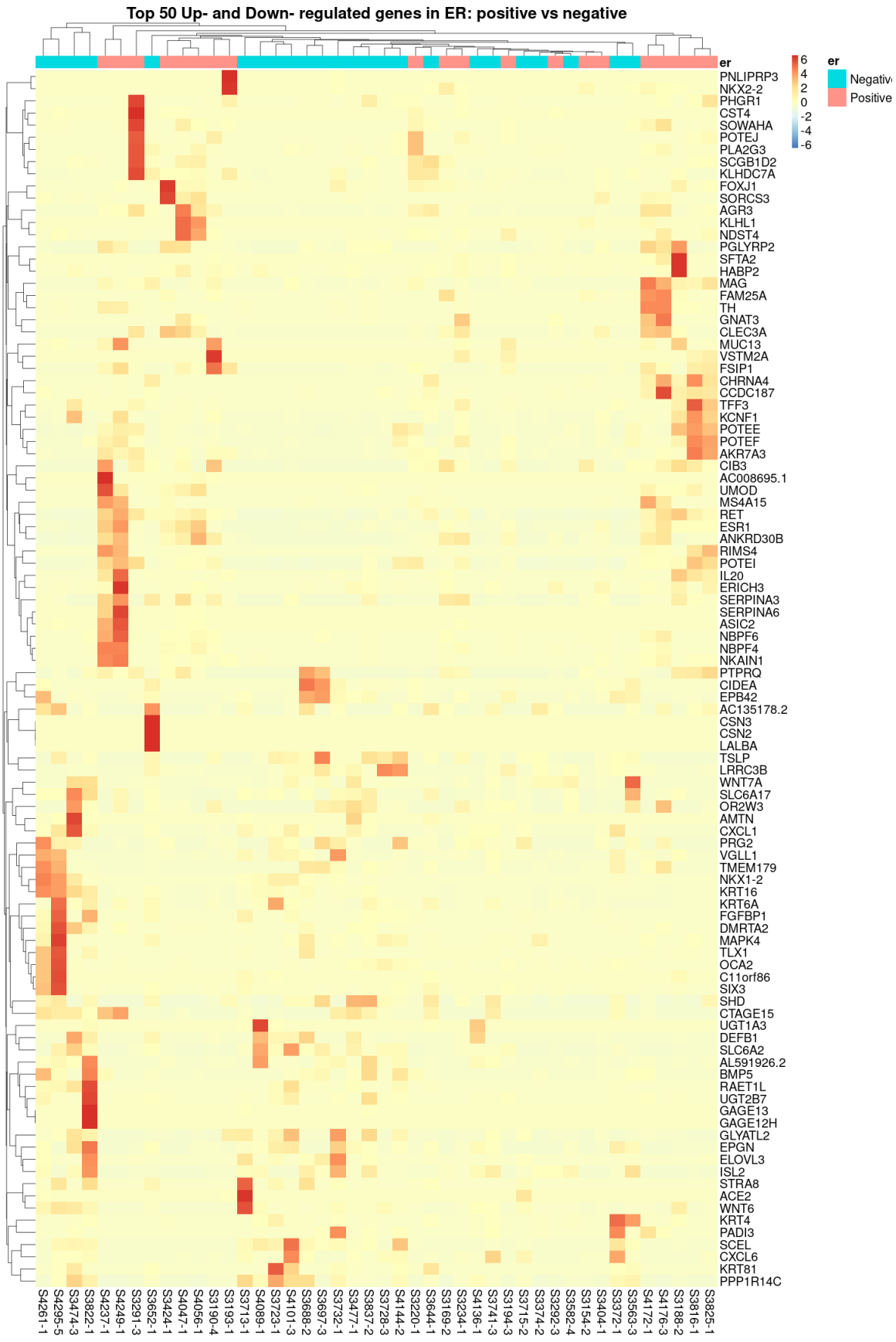
```
        border = FALSE,
        annotation = meta[, c("treatment"), drop = FALSE],
        main = "Top 50 Up- and Down- regulated genes in treatment: post vs pre",
        fontsize = 20)
```


Top 50 Up- and Down- regulated genes in treatment: post vs pre
treatment
post
pre
6
4
2
0
-2
-4
-6
CRISPLD1
NEFH
GRK2
ARAP1
ARRB1
GYPE
CCDC89
GCNA
NAIP
ARMC4
RASA4
MROH7
NFATC4
CHD3
IGFN1
FNDC5
ADAMTS14
MYOZ3
TSPAN11
FLRT2
AC009690.1
IFFO1
LILRB1
P2RX7
CMTM7
MYO1F
MYLK4
STAT2
RNF157
ACAP2
CCDC88A
FAM135A
KCNB2
AP003071.5
CRIP3
CREB5
ALK
TLCD1
SDF2
TRAF4
RPL23A
SLC44A3
F2RL1
SLC31A1
SMIM7
ERRFI1
FAM84A
CCT2
CDCA8
CLDN4
CDK8
VDAC1
ETF1
GABPB1
PHF5A
ZFAND6
SDHAF2
ARMC3
TMEM132A
UBA2
HIST1H2AI
HIST1H2AH
CDKN3
CCNB1
TRIB3
ACAT2
CKS2
HES6
RAB38
C2CD4A
ACKR2
CD24
PAX9
C3orf14
AK4
TNFRSF12A
CDC20
DEPDC1
EFNA5
MIEN1
CKAP5
VAPA
ORMDL3
PSMD3
HIGD1A
AKR1B10
BPIFA1
CASP14
UGT1A6
KRT4
KREMEN2
ACE2
CXCL8
LBP
NANOS1
MUCL1
CLCA2
TMEM45B
CA9
EGLN3
S3188-2
S3713-1
S3822-1
S3816-1
S3825-1
S3372-1
S3563-3
S3732-1
S3723-1
S4101-3
S4047-1
S3154-2
S3169-2
S4089-1
S3404-1
S3190-4
S3234-1
S4056-1
S4136-1
S3715-2
S3292-3
S3582-4
S3193-1
S3220-1
S3291-3
S4144-2
S3697-3
S3728-3
S3741-3
S3194-3
S3374-2
S4261-1
S4295-5
S3424-1
S3644-1
S3477-1
S3652-1
S3837-2
S4172-1
S4176-3
S4237-1
S4249-1
S3474-3
S3688-2

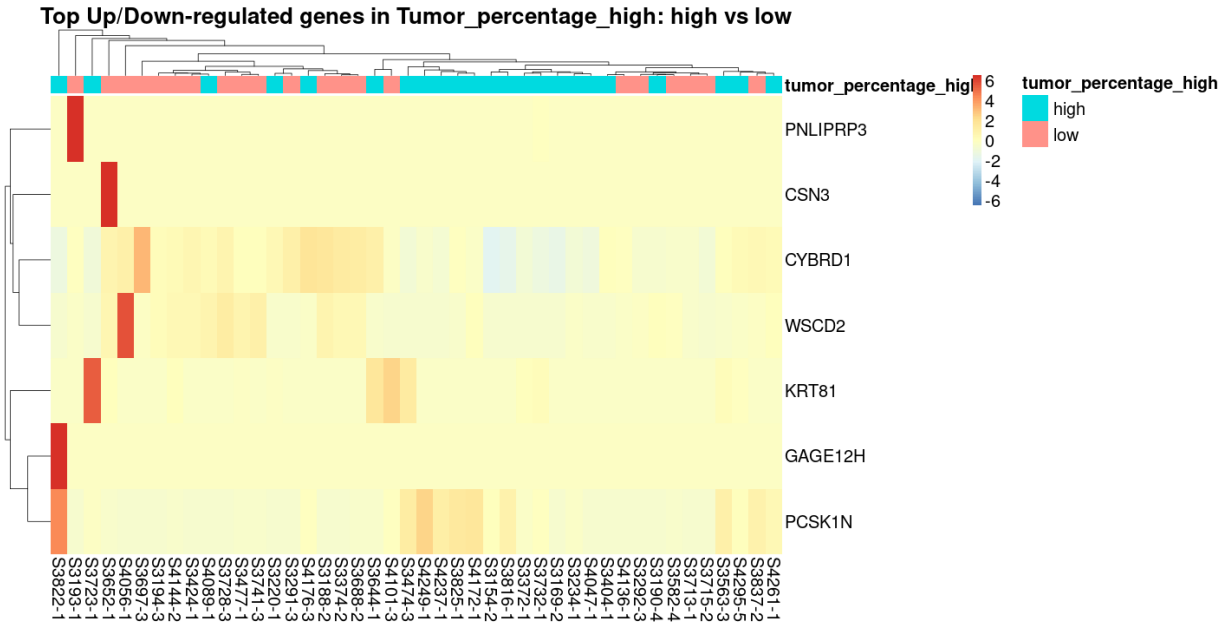
```r
# Create a matrix of normalized expression
sig_up <- resResponse_tb_significant %>% arrange(-log2FoldChange) %>% head(50) %>% pull(gene)
sig_down <- resResponse_tb_significant %>% arrange(log2FoldChange) %>% head(50) %>% pull(gene)
sig <- c(sig_up, sig_down)

row_annotation <- gene_symbol %>%
                    as_tibble() %>%
                    dplyr::filter(gene_id %in% sig)

plotmat <- txi$abundance[c(sig_up, sig_down),] %>% as.data.frame() %>%
          rownames_to_column(var = "ensembl_gene_id") %>%
          left_join(gene_symbol, by = c("ensembl_gene_id" = "gene_id")) %>%
          drop_na(symbol)

plotmat$ensembl_gene_id <- NULL

plotmat <- plotmat %>% column_to_rownames(var = "symbol") %>% as.matrix()

# Color palette
heat.colors <- brewer.pal(6, "YlOrRd")

# Plot heatmap
pheatmap(plotmat,
         scale = "row",
         show_rownames = TRUE,
         border = FALSE,
         annotation = meta[, c("response"), drop = FALSE],
         main = "Top 50 Up- and Down- regulated genes in Response: Yes vs No",
         fontsize = 20)
```


Top 50 Up- and Down- regulated genes in Response: Yes vs No
response
No
Yes
6
4
2
0
-2
-4
-6
DHRS2
POTEF
C16orf54
AL034430.1
CCL8
SPTSSB
GAGE12H
GAGE13
MYCN
CCL4L2
MARCO
CCL18
HLA-DQA2
PDCD1
KLHDC7B
CXCL10
GBP5
IFNG
SH2D1A
ICOS
GZMK
GZMA
GZMB
TRAT1
GZMH
CCL4
CCR5
ADAMDEC1
LAG3
LEP
PI16
TBX21
PTPRCAP
NKG7
CD8A
PRF1
CXorf65
MAL
IDO1
CCL13
FCRLA
CD79A
PAX5
SELL
STAP1
TCL1A
SSTR3
PTPRT
PCDH20
SLC13A2
BPIFA1
HABP2
HSFX3
P2RY10
PRAME
CYP21A2
DUOXA1
OR52E5
MUC15
GLP1R
ADAMTS18
ADAMTS20
SORCS3
FOXJ1
CYP4X1
CCNO
MCIDAS
CTAGE9
PNLIPRP3
LRP1B
CSMD1
MUM1L1
KRT15
A3GALT2
TLX1
KRT23
AGR3
ATRNL1
CLIC6
CPNE7
CRISP2
MKX
HS3ST5
CCDC175
PNLDC1
KLK14
TCN1
C4BPA
XDH
BTN1A1
CSN3
LALBA
CSN1S1
CSN2
KRT81
TDRD1
BPIFB1
ACTA1
SMIM34B
CEACAM16
S3822-1
S3477-1
S3713-1
S3715-2
S3652-1
S4047-1
S4056-1
S3816-1
S3825-1
S3220-1
S3234-1
S3292-3
S4136-1
S3374-2
S3582-4
S3837-2
S3688-2
S3697-3
S3193-1
S4261-1
S4295-5
S4172-1
S4176-3
S3732-1
S3741-3
S3474-3
S3723-1
S4101-3
S3728-3
S4144-2
S3291-3
S3372-1
S3190-4
S4089-1
S3404-1
S3194-3
S3563-3
S3154-2
S3169-2
S4237-1
S4249-1
S3188-2
S3424-1
S3644-1

```r
# Create a matrix of normalized expression
sig_up <- resER_tb_significant %>% arrange(-log2FoldChange) %>% head(50) %>% pull(gene)
sig_down <- resER_tb_significant %>% arrange(log2FoldChange) %>% head(50) %>% pull(gene)
sig <- c(sig_up, sig_down)

row_annotation <- gene_symbol %>%
                    as_tibble() %>%
                    dplyr::filter(gene_id %in% sig)

plotmat <- txi$abundance[c(sig_up, sig_down),] %>% as.data.frame() %>%
          rownames_to_column(var = "ensembl_gene_id") %>%
          left_join(gene_symbol, by = c("ensembl_gene_id" = "gene_id")) %>%
          drop_na(symbol)

plotmat$ensembl_gene_id <- NULL

plotmat <- plotmat %>% column_to_rownames(var = "symbol") %>% as.matrix()

# Color palette
heat.colors <- brewer.pal(6, "YlOrRd")

# Plot heatmap
pheatmap(plotmat,
         scale = "row",
         show_rownames = TRUE,
         border = FALSE,
         annotation = meta[, c("er"), drop = FALSE],
         main = "Top 50 Up- and Down- regulated genes in ER: positive vs negative",
         fontsize = 20)
```

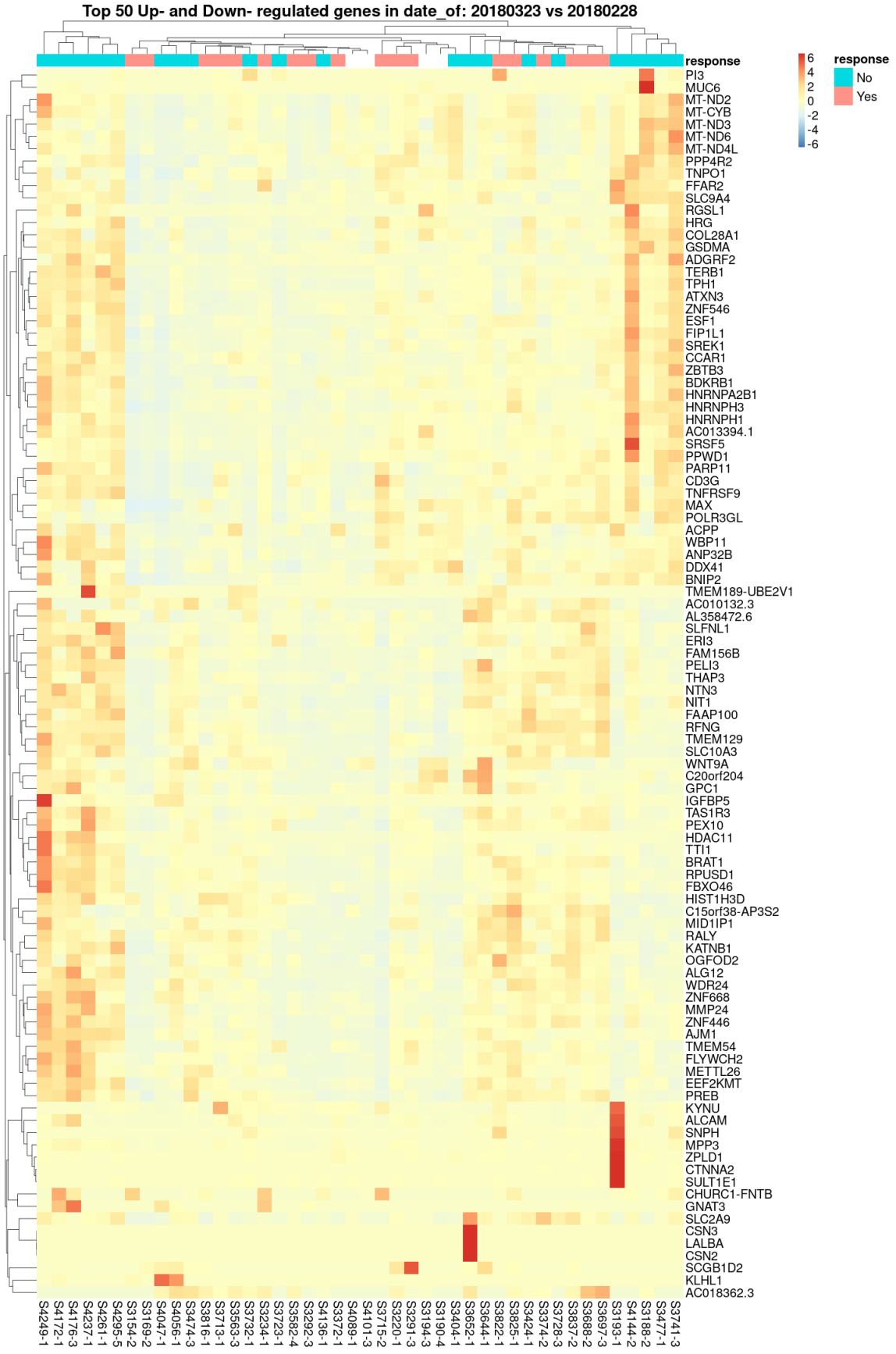

Top 50 Up- and Down- regulated genes in ER: positive vs negative
er
Negativ
Positive
6
4
2
0
-2
-4
-6
PNLIPRP3
NKX2-2
PHGR1
CST4
SOWAHA
POTEJ
PLA2G3
SCGB1D2
KLHDC7A
FOXJ1
SORCS3
AGR3
KLHL1
NDST4
PGLYRP2
SFTA2
HABP2
MAG
FAM25A
TH
GNAT3
CLEC3A
MUC13
VSTM2A
FSIP1
CHRNA4
CCDC187
TFF3
KCNF1
POTEE
POTEF
AKR7A3
CIB3
AC008695.1
UMOD
MS4A15
RET
ESR1
ANKRD30B
RIMS4
POTEI
IL20
ERICH3
SERPINA3
SERPINA6
ASIC2
NBPF6
NBPF4
NKAIN1
PTPRQ
CIDEA
EPB42
AC135178.2
CSN3
CSN2
LALBA
TSLP
LRRC3B
WNT7A
SLC6A17
OR2W3
AMTN
CXCL1
PRG2
VGLL1
TMEM179
NKX1-2
KRT16
KRT6A
FGFBP1
DMRTA2
MAPK4
TLX1
OCA2
C11orf86
SIX3
SHD
CTAGE15
UGT1A3
DEFB1
SLC6A2
AL591926.2
BMP5
RAET1L
UGT2B7
GAGE13
GAGE12H
GLYATL2
EPGN
ELOVL3
ISL2
STRA8
ACE2
WNT6
KRT4
PADI3
SCEL
CXCL6
KRT81
PPP1R14C
S4261-1
S4295-5
S3474-3
S3822-1
S4237-1
S4249-1
S3291-3
S3652-1
S3424-1
S4047-1
S4056-1
S3190-4
S3193-1
S3713-1
S4089-1
S3723-1
S4101-3
S3688-2
S3697-3
S3732-1
S3477-1
S3837-2
S3728-3
S4144-2
S3220-1
S3644-1
S3169-2
S3234-1
S4136-1
S3741-3
S3194-3
S3715-2
S3374-2
S3292-3
S3582-4
S3154-2
S3404-1
S3372-1
S3563-3
S4172-1
S4176-3
S3188-2
S3816-1
S3825-1

```r
# Create a matrix of normalized expression
sig_up <- resTP_tb_significant %>% arrange(-log2FoldChange) %>% head(50) %>% pull(gene)
sig_down <- resTP_tb_significant %>% arrange(log2FoldChange) %>% head(50) %>% pull(gene)
sig <- c(sig_up, sig_down)

row_annotation <- gene_symbol %>%
                    as_tibble() %>%
                    dplyr::filter(gene_id %in% sig)

plotmat <- txi$abundance[c(sig_up, sig_down),] %>% as.data.frame() %>%
          rownames_to_column(var = "ensembl_gene_id") %>%
          left_join(gene_symbol, by = c("ensembl_gene_id" = "gene_id")) %>%
          drop_na(symbol)

plotmat$ensembl_gene_id <- NULL

plotmat <- plotmat %>% column_to_rownames(var = "symbol") %>% as.matrix()

# Color palette
heat.colors <- brewer.pal(6, "YlOrRd")

# Plot heatmap
pheatmap(plotmat,
         scale = "row",
         show_rownames = TRUE,
         border = FALSE,
         annotation = meta[, c("tumor_percentage_high"), drop = FALSE],
         main = "Top Up/Down-regulated genes in Tumor_percentage_high: high vs low",
         fontsize = 20)
```




```r
# Create a matrix of normalized expression
sig_up <- resDO_tb_significant %>% arrange(-log2FoldChange) %>% head(50) %>% pull(gene)
sig_down <- resDO_tb_significant %>% arrange(log2FoldChange) %>% head(50) %>% pull(gene)
sig <- c(sig_up, sig_down)
```

```
row_annotation <- gene_symbol %>%
                    as_tibble() %>%
                    dplyr::filter(gene_id %in% sig)

plotmat <- txi$abundance[c(sig_up, sig_down),] %>% as.data.frame() %>%
          rownames_to_column(var = "ensembl_gene_id") %>%
          left_join(gene_symbol, by = c("ensembl_gene_id" = "gene_id")) %>%
          drop_na(symbol)

plotmat$ensembl_gene_id <- NULL

plotmat <- plotmat %>% column_to_rownames(var = "symbol") %>% as.matrix()

# Color palette
heat.colors <- brewer.pal(6, "YlOrRd")

# Plot heatmap
pheatmap(plotmat,
         scale = "row",
         show_rownames = TRUE,
         border = FALSE,
         annotation = meta[, c("response"), drop = FALSE],
         main = "Top 50 Up- and Down- regulated genes in date_of: 20180323 vs 20180228",
         fontsize = 20)
```


Top 50 Up- and Down- regulated genes in date_of: 20180323 vs 20180228
response
No
Yes
6
4
2
0
-2
-4
-6
PI3
MUC6
MT-ND2
MT-CYB
MT-ND3
MT-ND6
MT-ND4L
PPP4R2
TNPO1
FFAR2
SLC9A4
RGSL1
HRG
COL28A1
GSDMA
ADGRF2
TERB1
TPH1
ATXN3
ZNF546
ESF1
FIP1L1
SREK1
CCAR1
ZBTB3
BDKRB1
HNRNPA2B1
HNRNPH3
HNRNPH1
AC013394.1
SRSF5
PPWD1
PARP11
CD3G
TNFRSF9
MAX
POLR3GL
ACPP
WBP11
ANP32B
DDX41
BNIP2
TMEM189-UBE2V1
AC010132.3
AL358472.6
SLFNL1
ERI3
FAM156B
PELI3
THAP3
NTN3
NIT1
FAAP100
RFNG
TMEM129
SLC10A3
WNT9A
C20orf204
GPC1
IGFBP5
TAS1R3
PEX10
HDAC11
TTI1
BRAT1
RPUSD1
FBXO46
HIST1H3D
C15orf38-AP3S2
MID1IP1
RALY
KATNB1
OGFOD2
ALG12
WDR24
ZNF668
MMP24
ZNF446
AJM1
TMEM54
FLYWCH2
METTL26
EEF2KMT
PREB
KYNU
ALCAM
SNPH
MPP3
ZPLD1
CTNNA2
SULT1E1
CHURC1-FNTB
GNAT3
SLC2A9
CSN3
LALBA
CSN2
SCGB1D2
KLHL1
AC018362.3
S4249-1
S4172-1
S4176-3
S4237-1
S4261-1
S4295-5
S3154-2
S3169-2
S4047-1
S4056-1
S3474-3
S3816-1
S3713-1
S3563-3
S3732-1
S3234-1
S3723-1
S3582-4
S3292-3
S4136-1
S3372-1
S4089-1
S4101-3
S3715-2
S3220-1
S3291-3
S3194-3
S3190-4
S3404-1
S3652-1
S3644-1
S3822-1
S3825-1
S3424-1
S3374-2
S3728-3
S3837-2
S3688-2
S3697-3
S3193-1
S4144-2
S3188-2
S3477-1
S3741-3

R session

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-redhat-linux-gnu (64-bit)
## Running under: Fedora 32 (Workstation Edition)
##
## Matrix products: default
## BLAS/LAPACK: /usr/lib64/libopenblas-r0.3.12.so
##
## locale:
##  [1] LC_CTYPE=en_CA.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_CA.UTF-8        LC_COLLATE=en_CA.UTF-8
##  [5] LC_MONETARY=en_CA.UTF-8    LC_MESSAGES=en_CA.UTF-8
##  [7] LC_PAPER=en_CA.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_CA.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] ensembldb_2.14.0            AnnotationFilter_1.14.0
##  [3] GenomicFeatures_1.42.1      AnnotationDbi_1.52.0
##  [5] AnnotationHub_2.22.0        BiocFileCache_1.14.0
##  [7] dbplyr_2.1.0                knitr_1.30
##  [9] ggrepel_0.9.1               tximport_1.18.0
## [11] DEGreport_1.26.0            pheatmap_1.0.12
## [13] RColorBrewer_1.1-2          forcats_0.5.1
## [15] stringr_1.4.0               dplyr_1.0.5
## [17] purrr_0.3.4                 readr_1.4.0
## [19] tidyr_1.1.3                 tibble_3.1.0
## [21] ggplot2_3.3.3               tidyverse_1.3.0
## [23] DESeq2_1.30.1               SummarizedExperiment_1.20.0
## [25] Biobase_2.50.0              MatrixGenerics_1.2.1
## [27] matrixStats_0.58.0          GenomicRanges_1.42.0
## [29] GenomeInfoDb_1.26.2         IRanges_2.24.1
## [31] S4Vectors_0.28.1            BiocGenerics_0.36.0
##
## loaded via a namespace (and not attached):
##   [1] readxl_1.3.1                 backports_1.2.1
##   [3] circlize_0.4.12              plyr_1.8.6
##   [5] lazyeval_0.2.2               ConsensusClusterPlus_1.54.0
##   [7] splines_4.0.3                BiocParallel_1.24.1
##   [9] digest_0.6.27                htmltools_0.5.1.1
##  [11] fansi_0.4.2                  magrittr_2.0.1
##  [13] memoise_2.0.0                cluster_2.1.0
##  [15] limma_3.46.0                 ComplexHeatmap_2.6.2
##  [17] Biostrings_2.58.0            annotate_1.68.0
##  [19] Nozzle.R1_1.1-1              modelr_0.1.8
##  [21] askpass_1.1                  prettyunits_1.1.1
##  [23] colorspace_2.0-0             blob_1.2.1
```

```
##  [25] rvest_1.0.0                 rappdirs_0.3.3
##  [27] haven_2.3.1                 xfun_0.19
##  [29] crayon_1.4.1                RCurl_1.98-1.2
##  [31] jsonlite_1.7.1              genefilter_1.72.1
##  [33] survival_3.2-7              glue_1.4.2
##  [35] gtable_0.3.0                zlibbioc_1.36.0
##  [37] XVector_0.30.0              GetoptLong_1.0.5
##  [39] DelayedArray_0.16.2         shape_1.4.5
##  [41] scales_1.1.1                DBI_1.1.1
##  [43] edgeR_3.32.1                Rcpp_1.0.6
##  [45] progress_1.2.2              xtable_1.8-4
##  [47] lasso2_1.2-21.1             tmvnsim_1.0-2
##  [49] clue_0.3-58                 bit_4.0.4
##  [51] httr_1.4.2                  ellipsis_0.3.1
##  [53] farver_2.1.0                pkgconfig_2.0.3
##  [55] reshape_0.8.8               XML_3.99-0.5
##  [57] locfit_1.5-9.4              utf8_1.1.4
##  [59] labeling_0.4.2              tidyselect_1.1.0
##  [61] rlang_0.4.10                later_1.1.0.1
##  [63] munsell_0.5.0               BiocVersion_3.12.0
##  [65] cellranger_1.1.0            tools_4.0.3
##  [67] cachem_1.0.4                cli_2.3.1
##  [69] generics_0.1.0              RSQLite_2.2.3
##  [71] broom_0.7.5                 evaluate_0.14
##  [73] fastmap_1.1.0               ggdendro_0.1.22
##  [75] yaml_2.2.1                  bit64_4.0.5
##  [77] fs_1.5.0                    nlme_3.1-149
##  [79] mime_0.9                    xml2_1.3.2
##  [81] biomaRt_2.46.3              compiler_4.0.3
##  [83] rstudioapi_0.13             curl_4.3
##  [85] png_0.1-7                   interactiveDisplayBase_1.28.0
##  [87] reprex_1.0.0                geneplotter_1.68.0
##  [89] stringi_1.5.3               lattice_0.20-41
##  [91] ProtGenerics_1.22.0         Matrix_1.2-18
##  [93] psych_2.0.12                vctrs_0.3.6
##  [95] pillar_1.5.1                lifecycle_1.0.0
##  [97] BiocManager_1.30.10         GlobalOptions_0.1.2
##  [99] cowplot_1.1.1               bitops_1.0-6
## [101] rtracklayer_1.50.0          httpuv_1.5.5
## [103] R6_2.5.0                    promises_1.2.0.1
## [105] MASS_7.3-53                 assertthat_0.2.1
## [107] openssl_1.4.3               rjson_0.2.20
## [109] withr_2.4.1                 GenomicAlignments_1.26.0
## [111] Rsamtools_2.6.0             mnormt_2.0.2
## [113] GenomeInfoDbData_1.2.4      hms_1.0.0
## [115] grid_4.0.3                  rmarkdown_2.5
## [117] Cairo_1.5-12.2              logging_0.10-108
## [119] shiny_1.6.0                 lubridate_1.7.10
```