final_project

2025-05-07

```
#loading the required libraries
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.0     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts
to become errors
```

```
library(tximport)
library(DESeq2)
```

```
## Loading required package: S4Vectors
## Loading required package: stats4
## Loading required package: BiocGenerics
##
## Attaching package: 'BiocGenerics'
##
## The following objects are masked from 'package:lubridate':
##
##     intersect, setdiff, union
##
## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union
##
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
##
## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, 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 objects are masked from 'package:lubridate':
##
##     second, second<-
##
## The following objects are masked from 'package:dplyr':
##
##     first, rename
##
## The following object is masked from 'package:tidyr':
##
##     expand
##
## The following object is masked from 'package:utils':
##
##     findMatches
##
## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname
##
## Loading required package: IRanges
##
```

```
## Attaching package: 'IRanges'
##
## The following object is masked from 'package:lubridate':
##
##     %within%
##
## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice
##
## The following object is masked from 'package:purrr':
##
##     reduce
##
## Loading required package: GenomicRanges
## Loading required package: GenomeInfoDb
## Loading required package: SummarizedExperiment
## Loading required package: MatrixGenerics
## Loading required package: matrixStats
##
## Attaching package: 'matrixStats'
##
## The following object is masked from 'package:dplyr':
##
##     count
##
##
## 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
```

```
# Set your netid
netid <- 'rs8579'

# Sample names = directory names from salmon output
sample_names <- c('control1', 'control2', 'control3', 'treated1', 'treated2', 'treated
3')

# Sample condition (must match order)
sample_condition <- c(rep('control', 3), rep('treated', 3))

# Build the file paths to quant.sf
files <- file.path("/scratch", netid, "final_project", "project.2025", "res", "salmon",
sample_names, "quant.sf")

# Set names of the vector for tximport
names(files) <- sample_names
```

```
tx2gene <- read.table(file.path("/scratch", "rs8579", "final_project", "project.2025",
"res", "salmon", "tx2gene.tsv"), header = FALSE, sep = "\t")
```

```
txi <- tximport(files, type="salmon", tx2gene=tx2gene) # The txi variable is a simple li
st object containing gene counts and other info
```

```
## reading in files with read_tsv
```

```
## 1 2 3 4 5 6
## summarizing abundance
## summarizing counts
## summarizing length
```

```
# Create metadata dataframe
metadata.df <- data.frame(
  sample = factor(sample_names),
  condition = factor(sample_condition, levels = c('control', 'treated'))  # control = re
ference level
)

# Set row names to sample names
row.names(metadata.df) <- sample_names

# View metadata
metadata.df
```

```
##            sample condition
## control1 control1   control
## control2 control2   control
## control3 control3   control
## treated1 treated1   treated
## treated2 treated2   treated
## treated3 treated3   treated
```

```
dds <- DESeqDataSetFromTximport(txi,
                                colData = metadata.df,
                                design = ~ condition)
```

```
## using counts and average transcript lengths from tximport
```

```
dds <- estimateSizeFactors(dds)
```

```
## using 'avgTxLength' from assays(dds), correcting for library size
```

```
counts(dds) %>% # use the counts function to extract a matrix and "pipe" matrix to the h
ead() function to view first 6 rows.
  head()
```

```
##                 control1 control2 control3 treated1 treated2 treated3
## ENSG00000000003     1668     1656     1536     1487     1467     1269
## ENSG00000000005        0        0        0        0        1        0
## ENSG00000000419     2142     2125     1924     2416     2652     2035
## ENSG00000000457     1107     1124      994      895      931      719
## ENSG00000000460     1800     1905     1517     1468     1670     1293
## ENSG00000000938        3        2        0        1        3        3
```

```
counts(dds) %>% # pipe count matrix to the dim() function which returns the matrix dimen
sions (rows and columns)
  dim()
```

```
## [1] 78487     6
```

```
#filtering the low count genes below:
keep <- rowSums(counts(dds)) >= 10
dds <- dds[keep,]
```

```
dds <- DESeq(dds)
```

```
## using pre-existing normalization factors
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

```
res <- results(dds, contrast = c('condition', 'treated', 'control'), alpha = 0.05)

# View top results
res
```

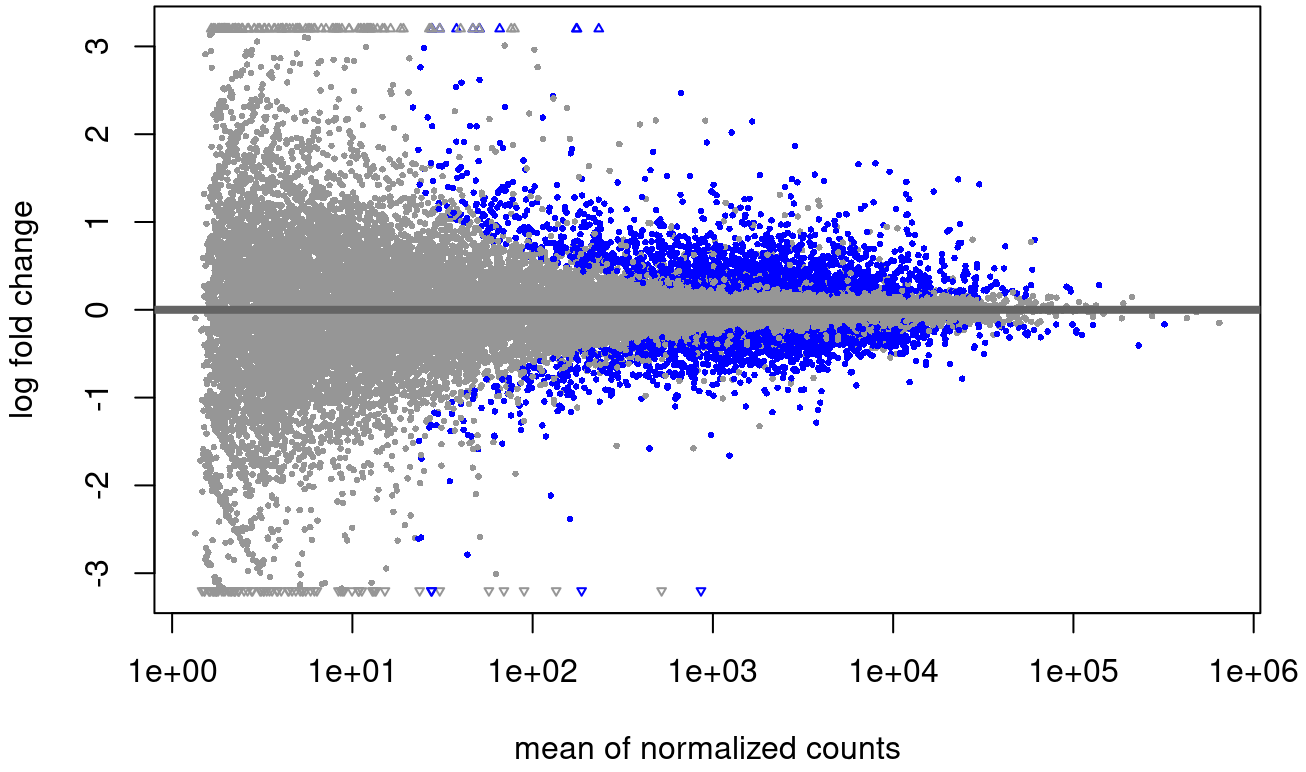
```
## log2 fold change (MLE): condition treated vs control
## Wald test p-value: condition treated vs control
## DataFrame with 24766 rows and 6 columns
##                   baseMean log2FoldChange     lfcSE      stat      pvalue
##                  <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000000003 1510.97053     -0.0266523 0.1024142 -0.260240 7.94679e-01
## ENSG00000000419 2218.67598      0.3367730 0.0741199  4.543622 5.52958e-06
## ENSG00000000457  956.01232     -0.2797829 0.0868522 -3.221370 1.27579e-03
## ENSG00000000460 1597.98583     -0.0799940 0.0906208 -0.882733 3.77381e-01
## ENSG00000000938    1.90726      0.9224572 1.6243824  0.567882 5.70115e-01
## ...                    ...            ...       ...       ...         ...
## ENSG00000310534    43.2405      -0.761399  1.793209 -0.424602    0.671127
## ENSG00000310535    23.3162       0.352185  2.908964  0.121069    0.903636
## ENSG00000310536    12.1791      -0.569181  0.773863 -0.735505    0.462032
## ENSG00000310537    52.7287      -0.338298  0.345417 -0.979389    0.327388
## ENSG00000310539    32.1650       0.129149  0.498282  0.259188    0.795490
##                       padj
##                  <numeric>
## ENSG00000000003 8.97378e-01
## ENSG00000000419 8.22998e-05
## ENSG00000000457 8.97571e-03
## ENSG00000000460 5.97632e-01
## ENSG00000000938          NA
## ...                     ...
## ENSG00000310534    0.822514
## ENSG00000310535    0.955054
## ENSG00000310536          NA
## ENSG00000310537    0.549594
## ENSG00000310539    0.897886
```

```
#confirming that the lof2FC changes that we see are RNAi/control (in this case, treated/control)
levels(dds$condition)
```

```
## [1] "control" "treated"
```

Since control is first, it will be the reference level (denominator)

```
plotMA(res)
```




```
#plotting the log2FC values before shrinking the estimates
```

now, I will lfc shrinkage to shrink the log2FC and plot it again to see it it helps the dispersion:

```
resultsNames(dds)
```

```
## [1] "Intercept"                   "condition_treated_vs_control"
```

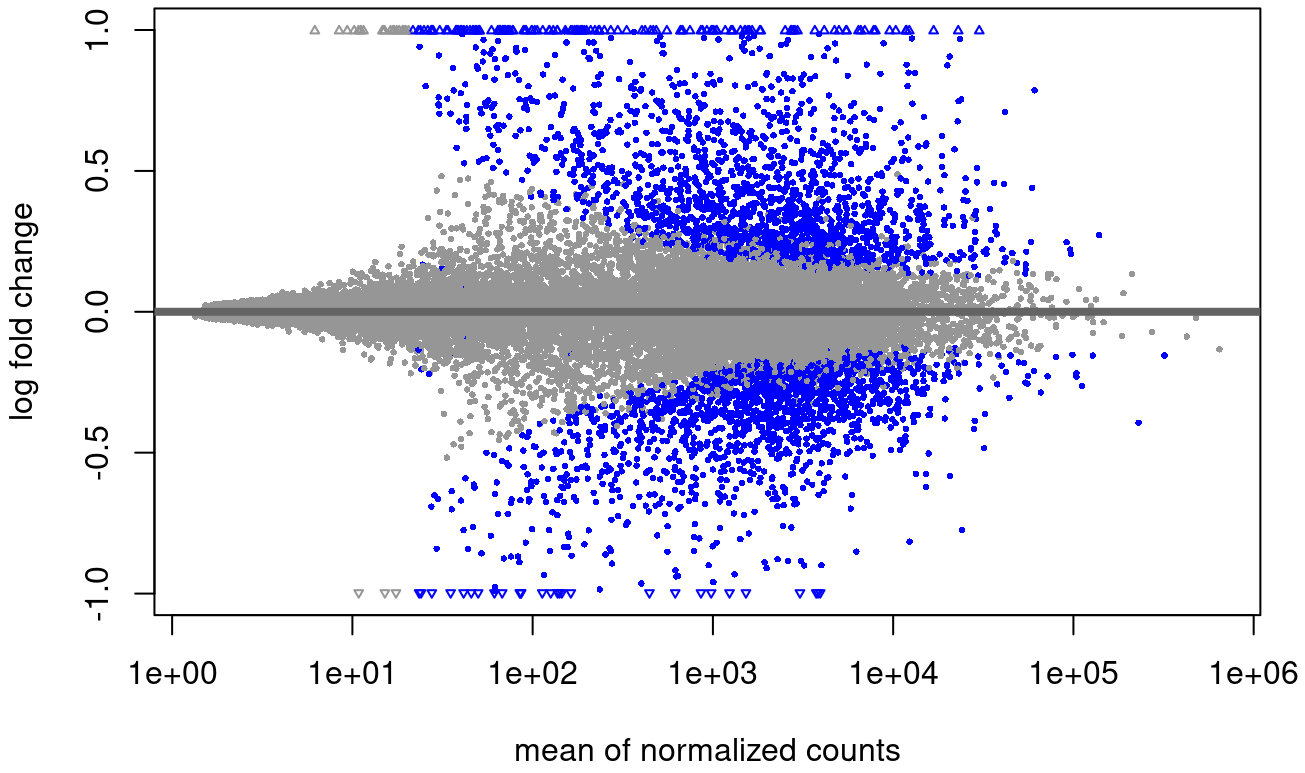
```
#shrinking the log2FC estimates
res.lfcShrink <- lfcShrink(dds,
                           res = res,
                           coef = 'condition_treated_vs_control',
                           type = 'apeglm')
```

```
## using 'apeglm' for LFC shrinkage. If used in published research, please cite:
##     Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for
##     sequence count data: removing the noise and preserving large differences.
##     Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895
```

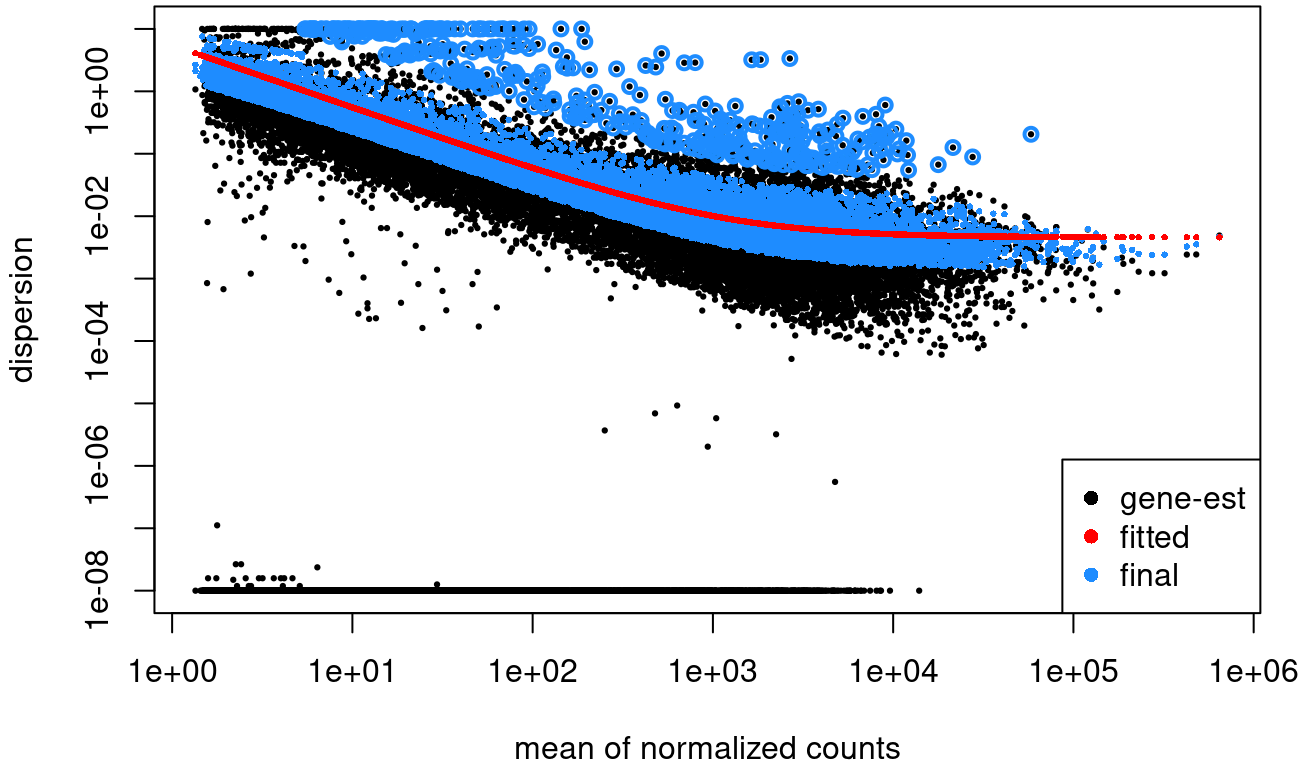
```
res.lfcShrink %>%
  as_tibble() %>%
  summarise(padj_NA = sum(is.na(padj)), # summarise collapses output to a single row wit
h new columns with summaries of the data
            padj_notNA = sum(!is.na(padj)))
```

```
## # A tibble: 1 × 2
##   padj_NA padj_notNA
##     <int>      <int>
## 1    8662      16104
```

```
#plotting the estimates after shrinking
plotMA(res.lfcShrink)
```




```
plotDispEsts(dds)
```




```
#ordering the res using the padj value to get the top 10 most significant genes
res_ordered <- res[order(res$padj), ]
top10 <- head(res_ordered, 10) #printing out the top 10
top10[,c(2,5,6)] #selecting only specific columns to view (for the results section of th
e paper)
```

```
## log2 fold change (MLE): condition treated vs control
## Wald test p-value: condition treated vs control
## DataFrame with 10 rows and 3 columns
##                 log2FoldChange       pvalue         padj
##                      <numeric>    <numeric>    <numeric>
## ENSG00000175334        1.65818 1.57213e-137 2.53176e-133
## ENSG00000163041        1.66799 3.19987e-118 2.57654e-114
## ENSG00000196396        1.15095 8.30924e-101  4.46040e-97
## ENSG00000105976        1.57137  2.24035e-95  9.01965e-92
## ENSG00000128595        1.48572  9.10202e-94  2.93158e-90
## ENSG00000101384        1.31508  3.01267e-86  8.08602e-83
## ENSG00000124333        1.48742  4.65482e-86  1.07087e-82
## ENSG00000117632        1.34686  2.54574e-79  5.12457e-76
## ENSG00000145919        1.29112  2.72808e-67  4.68204e-64
## ENSG00000213281        1.18569  2.90738e-67  4.68204e-64
```

```
#this will tell me the total # of genes that are significant based on the padj value (< 0.05):

sum(res$padj < 0.05, na.rm = TRUE)
```

```
## [1] 3609
```

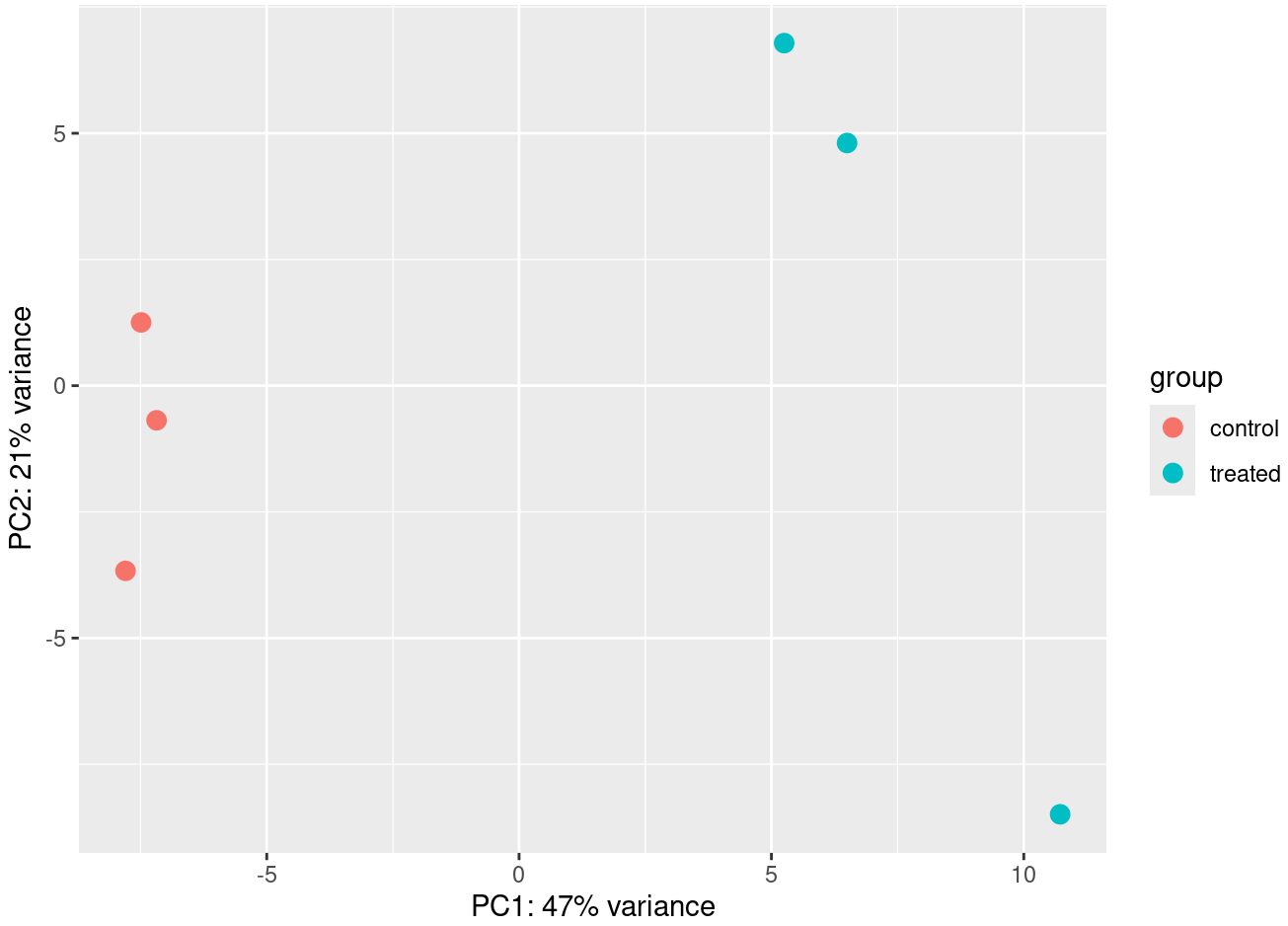
```
#sig_up will identify the genes that are significant as well as upregulated in treated vs. sig_down will identify the genes that are significant+downregulated
sig_up <- sum(res$padj < 0.05 & res$log2FoldChange > 0, na.rm = TRUE)
sig_down <- sum(res$padj < 0.05 & res$log2FoldChange < 0, na.rm = TRUE)
c(up = sig_up, down = sig_down) #combining them both
```

```
##   up down
## 1900 1709
```

PCA plot below:

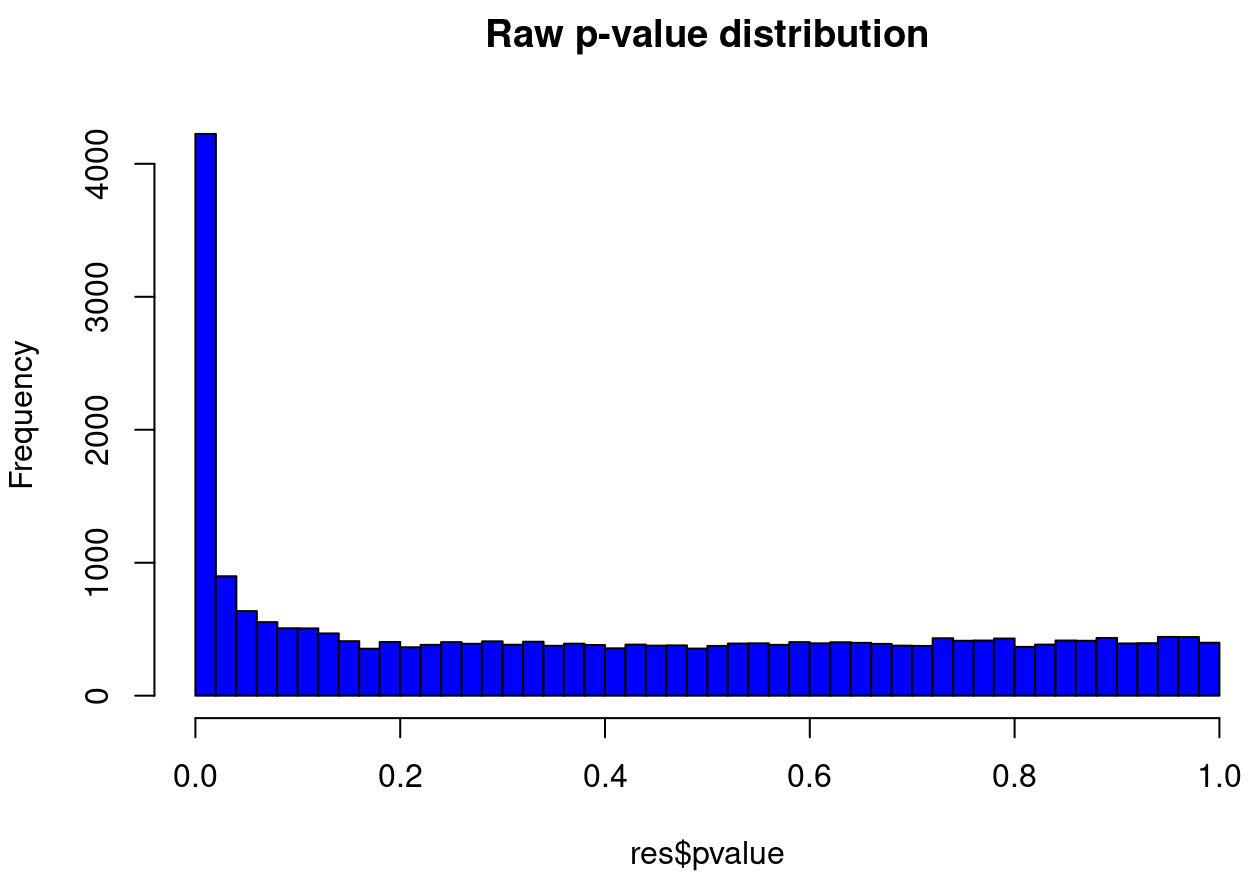
```
#variance stabilizing transformation is an inbuilt function of DESeq2. The other way to do it is extract the counts from dds, normalize it, and then do log transformation before running a PCA analysis. this transformation also normalizes in respect to library size
vsd <- vst(dds, blind = FALSE)
plotPCA(vsd, intgroup = "condition")
```

```
## using ntop=500 top features by variance
```



In the code block below, the raw p-values are plotted for the genes:

```
hist(res$pvalue, breaks = 50, col = "blue", main = "Raw p-value distribution")
```



There is a strong left skew, meaning that there are a high number of genes that are significantly different between treated vs control