Final Project: Differential Gene Expression Analysis with Breast Cancer RNA-seq Data

Jessie Bologna

5/5/2021

All Code/Scripts run provided below -

Step 1: QC of fastq sample files

```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=4:00:00
#SBATCH --mem=4GB
#SBATCH --job-name=final_a
#SBATCH --mail-type=FAIL
#SBATCH --mail-user=jb7303@nyu.edu
#SBATCH --array=1-6

module purge

module load fastp/intel/0.20.1

echo The array index is: ${SLURM_ARRAY_TASK_ID}

table=/scratch/work/courses/BI7653/project.2021/project_fastqs.txt
line="$(head -n ${SLURM_ARRAY_TASK_ID} "${table}" | tail -n 1)"
sample="$(printf "%s" "${line}" | cut -f1)"
fq1="$(printf "%s" "${line}" | cut -f2)"

fqdir=/scratch/work/courses/BI7653/project.2021/fastqs

fastp -i $fqdir/$fq1 --trim_poly_g -o $sample.out.fq

module purge
```

First use fastp to trim the fastq adapters:

```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=4:00:00
#SBATCH --mem=4GB
#SBATCH --job-name=final_a
#SBATCH --mail-type=FAIL
#SBATCH --mail-user=jb7303@nyu.edu
#SBATCH --array=1-6

module purge

module load fastqc/0.11.9

# Path to 3-column (tab-delimited) table with sample name, fastq 1 file names
table=/scratch/work/courses/BI7653/project.2021/project_fastqs.txt

# Define sample, fq1 variables for current array index

line="$(head -n $SLURM_ARRAY_TASK_ID $table | tail -n 1)"
sample="$(printf "%s" "${line}" | cut -f1)"
fq1="$(printf "%s" "${line}" | cut -f2)"

fqdir =/scratch/jb7303/final_project

# Run fastqc
fastqc $sample.out.fq

echo _ESTATUS_ [ fastqc for $sample ]: $?
echo _END_ [ fastp for $sample ]: $(date)

module purge
```

Next, run fastqc and Multiqc to evaluate our reads to confirm downstream analysis accuracy (html files attached for reference):

Step 2: Reference file: Download, Normalize, and Index the human reference file -

```
wget 'http://ftp.ensembl.org/pub/release-104/fasta/homo_sapiens/cdna/Homo_sapiens.GRCh38.cdna.all.fa.gz
```

First download the file:

```
java -jar "${PICARD_JAR}" NormalizeFasta -I Homo_sapiens.GRCh38.cdna.all.fa -O GRCH38.cdna.all_normaliz
```

Next, Gunzip and Normalize using Picard Tools:

```
salmon index -t GRCH38.cdna.all_normalized.fasta -i GRCH38.cdna.all_normalized.fasta_index -k 31
```

Create and index file to be used to run Salmon in the following steps: note: I tried reindexing using a different method because treated3 samples had an error in the next step, the second method of indexing still resulted in the same error. It was discovered that the file was inproperly downloaded and thus needed to be re-downloaded and rerun through QC steps. ***

```
#!/bin/bash
#
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=4
#SBATCH --time=24:00:00
#SBATCH --mem=8GB
#SBATCH --job-name=final_d
#SBATCH --mail-type=FAIL
#SBATCH --mail-user=jb7303@nyu.edu
#SBATCH --array=1-6

module purge

module load salmon/1.4.0

echo The array index is: ${SLURM_ARRAY_TASK_ID}

table=/scratch/work/courses/BI7653/project.2021/project_fastqs.txt
line="$(head -n ${SLURM_ARRAY_TASK_ID} "${table}" | tail -n 1)"
sample="$(printf "%s" "${line}" | cut -f1)"
fq1="$(printf "%s" "${line}" | cut -f2)"

fqdir=/scratch/work/courses/BI7653/project.2021/fastqs

salmon_index_dir=/scratch/jb7303/final_project/GRCH38.cdna.all_normalized.fasta_index

mkdir "${sample}"
cd "${sample}"

salmon quant -i ${salmon_index_dir} -l A -r $fqdir/$fq1 --validateMappings --gcBias --threads ${SLURM_C

echo _ESTATUS_ [ salmon quant $sample ]: $?
echo _END_ [ salmon.slurm ]: $(date)

module purge
```

Step 3: Run Salmon: A psuedo alignment tool to quantify transcripts-

---

```
library(tximport)
library(DESeq2)
sample_names <- c('control1','control2','control3','treated1','treated2','treated3')
sample_condition <- c( rep('Control',3), rep('NRDE2_treated',3))

files <- file.path("~/Jessie School/NYU/NGS/Final_project/supplimentary_docs",sample_names,paste(sample

names(files) <- sample_names
tx2gene <- read.table("tx2gene.csv",header=F,sep=",")
txi <- tximport(files, type="salmon", tx2gene=tx2gene)
samples <- data.frame(sample_names=sample_names,condition=sample_condition)
row.names(samples) <- sample_names

head(txi$counts)
```

Step 4: Convert Salmon TPM's to gene-level counts & conduct DESeq2 -

```
##                    control1 control2 control3 treated1 treated2 treated3
## ENSG00000000003.15 1689.711 1681.267 1539.471 1513.906 1475.388 1277.780
## ENSG00000000005.6     0.000    0.000    0.000    0.000    1.000    0.000
## ENSG00000000419.12  905.671  993.550  942.518 1053.566 1211.513  945.418
## ENSG00000000457.14 1122.401 1146.503 1008.157  903.649  944.409  726.010
## ENSG00000000460.17 1925.017 2012.596 1599.891 1584.311 1775.944 1360.165
## ENSG00000000938.13    3.000    2.000    0.000    1.000    3.000    3.000
```

```
# create DESeq2 object
library("DESeq2")
ddsTxi <- DESeqDataSetFromTximport(txi,
                                   colData = samples,
                                   design = ~ condition)
```

```
# keep only genes with 10 or more reads
keep <- rowSums(counts(ddsTxi)) >= 10
ddsTxi <- ddsTxi[keep,]

# run DESeq on the filtered data
ddsTxi <- DESeq(ddsTxi)


# get the shrunken values - and order by p-values - using contrast will additionally set to 0 the estim

resultsNames(ddsTxi)
```

```
## [1] "Intercept"                          "condition_NRDE2_treated_vs_Control"
```

```
res <- results(ddsTxi, contrast = c('condition','Control','NRDE2_treated'))
res_ordered<- res[order(res$padj),]
head(res_ordered, 20)
```

```
## log2 fold change (MLE): condition Control vs NRDE2_treated
## Wald test p-value: condition Control vs NRDE2_treated
## DataFrame with 20 rows and 6 columns
##                      baseMean log2FoldChange     lfcSE      stat       pvalue
##                     <numeric>      <numeric> <numeric> <numeric>    <numeric>
## ENSG00000196396.10  6634.677       -1.15682 0.0411796  -28.0921 1.22490e-173
## ENSG00000175334.8   6467.739       -1.66316 0.0632389  -26.2997 1.93534e-152
## ENSG00000206286.11  2809.415        1.39443 0.0572691   24.3488 5.96982e-131
## ENSG00000119720.18   997.138        4.59591 0.1934671   23.7555 9.63241e-125
## ENSG00000101384.12 11773.506       -1.30597 0.0596870  -21.8803 4.00061e-106
## ...                      ...            ...       ...       ...          ...
## ENSG00000228116.9    187.906     -10.661293 0.5875507  -18.1453  1.39826e-73
## ENSG00000079739.17  7150.550      -0.891515 0.0517351  -17.2323  1.52006e-66
## ENSG00000155660.11 23038.515      -0.754786 0.0439569  -17.1710  4.37544e-66
## ENSG00000182934.12 13227.984      -0.864454 0.0512224  -16.8765  6.70156e-64
## ENSG00000197702.14  6157.698      -0.658469 0.0395841  -16.6347  3.90870e-62
##                           padj
##                      <numeric>
## ENSG00000196396.10 1.68375e-169
## ENSG00000175334.8  1.33016e-148
## ENSG00000206286.11 2.73537e-127
## ENSG00000119720.18 3.31018e-121
## ENSG00000101384.12 1.09985e-102
## ...                        ...
## ENSG00000228116.9   1.20128e-70
## ENSG00000079739.17  1.22911e-63
## ENSG00000155660.11  3.34138e-63
## ENSG00000182934.12  4.84840e-61
## ENSG00000197702.14  2.68645e-59
```

```
# get the shrunken values
res_shrunk <- lfcShrink(ddsTxi, contrast = c('condition','Control','NRDE2_treated'), type= 'normal' )
res_shrunkOrdered <- res_shrunk[order(res_shrunk$pvalue),]
head(res_shrunkOrdered,10)
```

```
## log2 fold change (MAP): condition Control vs NRDE2_treated
## Wald test p-value: condition Control vs NRDE2_treated
## DataFrame with 10 rows and 6 columns
##                      baseMean log2FoldChange     lfcSE      stat       pvalue
##                     <numeric>      <numeric> <numeric> <numeric>    <numeric>
## ENSG00000196396.10  6634.677       -1.14597 0.0407896  -28.0921 1.22490e-173
## ENSG00000175334.8   6467.739       -1.62684 0.0618447  -26.2997 1.93534e-152
## ENSG00000206286.11  2809.415        1.36937 0.0562119   24.3488 5.96982e-131
## ENSG00000119720.18   997.138        3.79060 0.1576105   23.7555 9.63241e-125
## ENSG00000101384.12 11773.506       -1.28050 0.0585187  -21.8803 4.00061e-106
## ENSG00000128595.17 22876.387       -1.43526 0.0666856  -21.5219 9.71282e-103
## ENSG00000124333.16  2742.898       -1.45559 0.0683968  -21.2721 2.06049e-100
## ENSG00000143384.13 21358.666       -1.04060 0.0493345  -21.0924  9.34863e-99
## ENSG00000117632.23 16816.515       -1.31972 0.0655333  -20.1373  3.47550e-90
```

```
## ENSG00000164066.13  1269.180        1.47100 0.0741811   19.8110  2.39456e-87
##                            padj
##                       <numeric>
## ENSG00000196396.10 1.68375e-169
## ENSG00000175334.8  1.33016e-148
## ENSG00000206286.11 2.73537e-127
## ENSG00000119720.18 3.31018e-121
## ENSG00000101384.12 1.09985e-102
## ENSG00000128595.17  2.22521e-99
## ENSG00000124333.16  4.04621e-97
## ENSG00000143384.13  1.60633e-95
## ENSG00000117632.23  5.30824e-87
## ENSG00000164066.13  3.29157e-84
```

```
summary(res)
```

```
##
## out of 17961 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 1789, 10%
## LFC < 0 (down)     : 2033, 11%
## outliers [1]       : 56, 0.31%
## low counts [2]     : 4159, 23%
## (mean count < 29)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
mcols(res)$description
```

```
## [1] "mean of normalized counts for all samples"
## [2] "log2 fold change (MLE): condition Control vs NRDE2_treated"
## [3] "standard error: condition Control vs NRDE2_treated"
## [4] "Wald statistic: condition Control vs NRDE2_treated"
## [5] "Wald test p-value: condition Control vs NRDE2_treated"
## [6] "BH adjusted p-values"
```

```
# how many genes have a pvalue less than 0.05
sum(res$padj < 0.05, na.rm=TRUE)
```

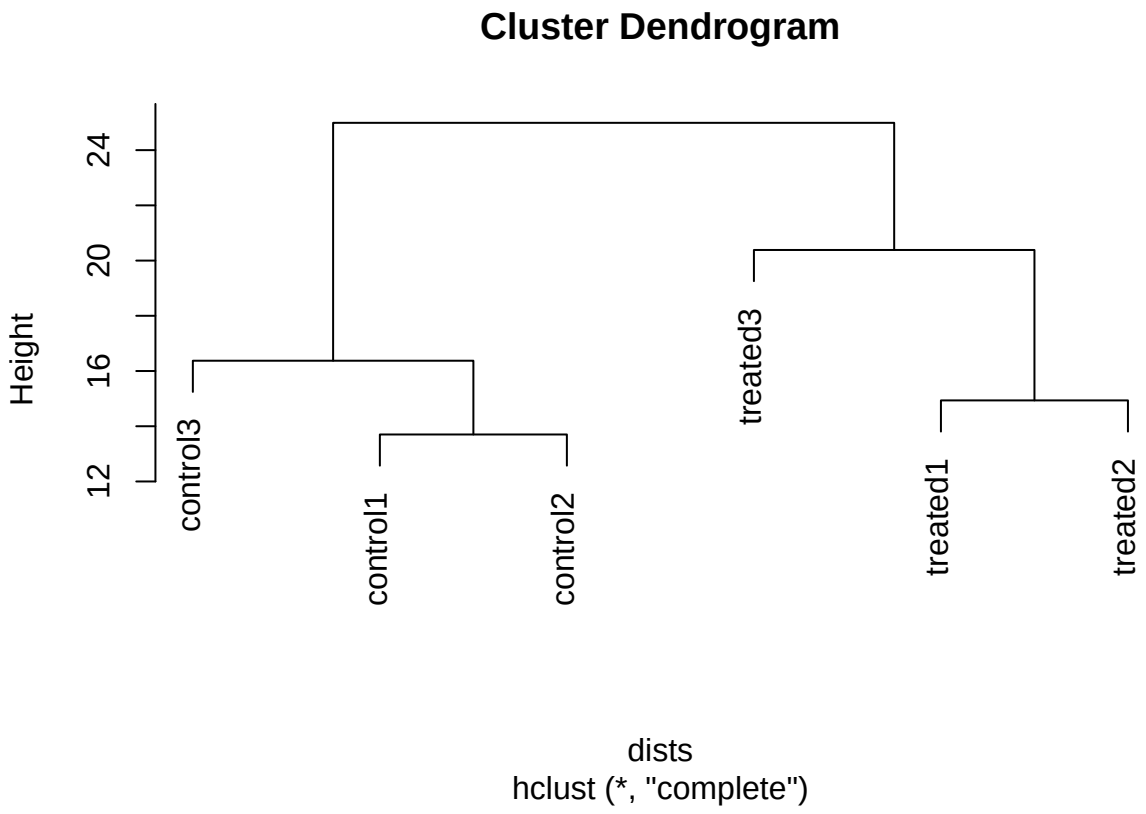
```
## [1] 3043
```

```
# 3043

res_05 <- results(ddsTxi, alpha = 0.05)
summary(res_05)
```

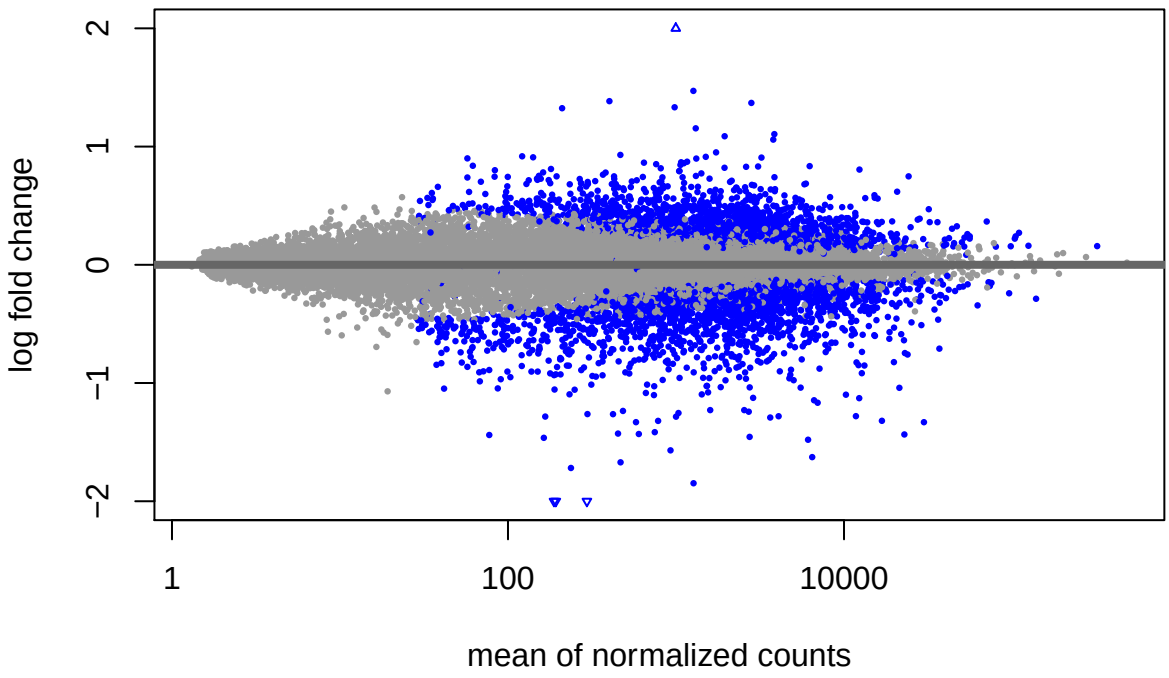
```
##
## out of 17961 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)       : 1662, 9.3%
## LFC < 0 (down)     : 1381, 7.7%
```

```
## outliers [1]       : 56, 0.31%
## low counts [2]     : 4159, 23%
## (mean count < 29)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

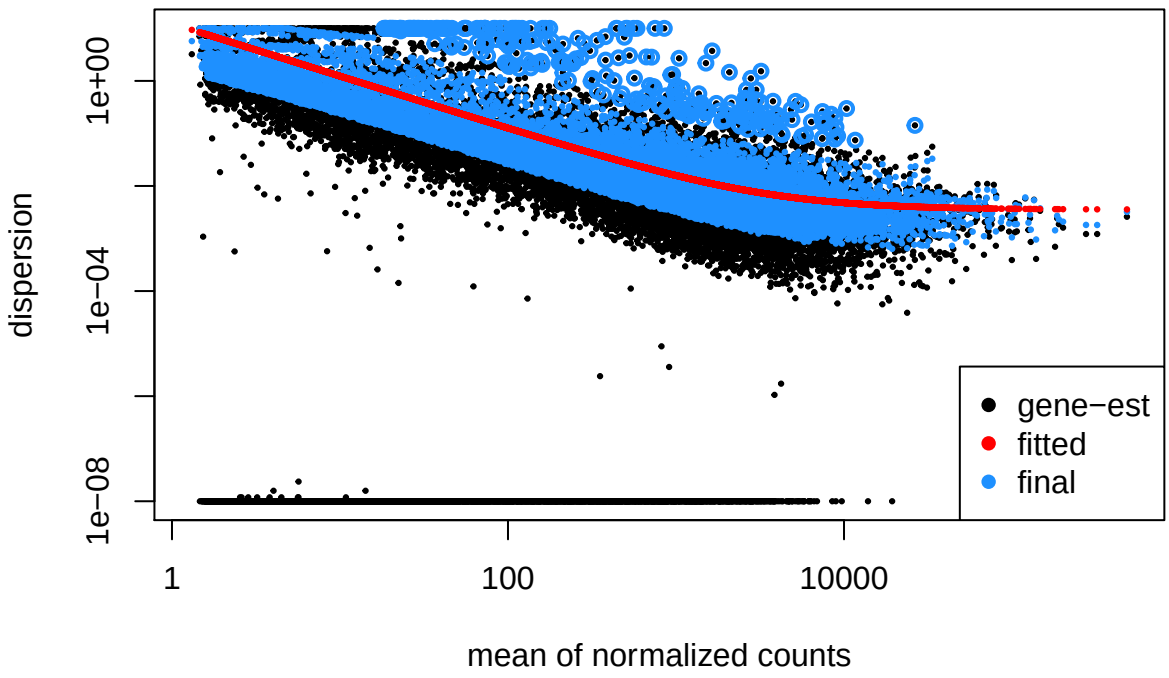
```
# save results as .csv file
write.csv(res_shrunkOrdered, 'dds_shrunken_results_ordered.csv')

# cluster the results
rld <- rlog(ddsTxi)
dists <- dist(t(assay(rld)))
plot(hclust(dists))
```



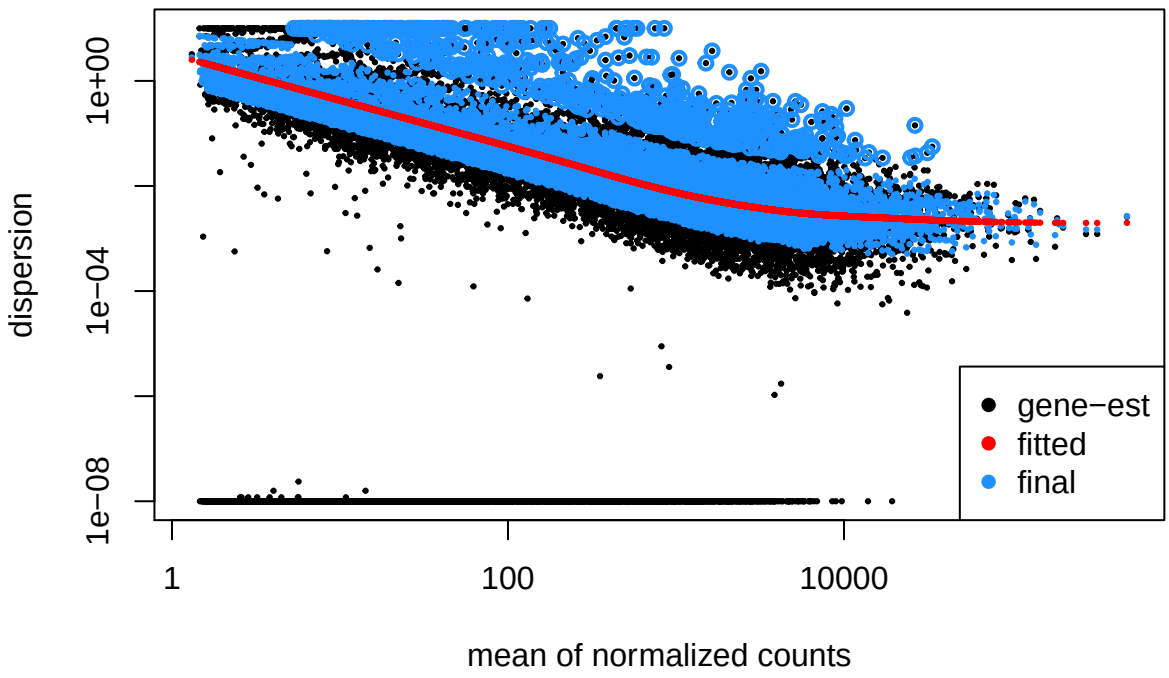

```
# MA - Plot - shows the log2 fold changes attributable to a given variable over the mean of normalized
# note that points colored red = padj values lower than 0.1
# points that fall out of the window are plotted as open triangles
# note that it is more useful to show teh shrunken log2 fold changes - which remove the noise associate
plotMA(res_shrunk, ylim =c(-2,2))
```



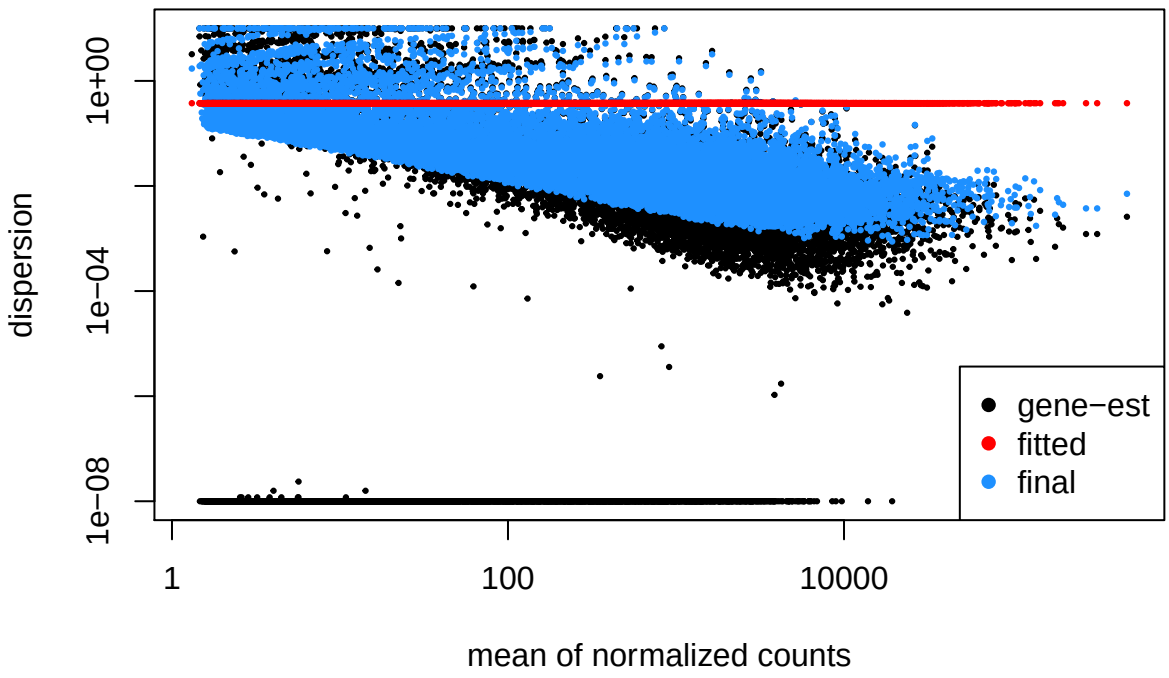

```
dds1<- estimateDispersions(ddsTxi, fitType = 'parametric')
plotDispEsts(dds1)
```



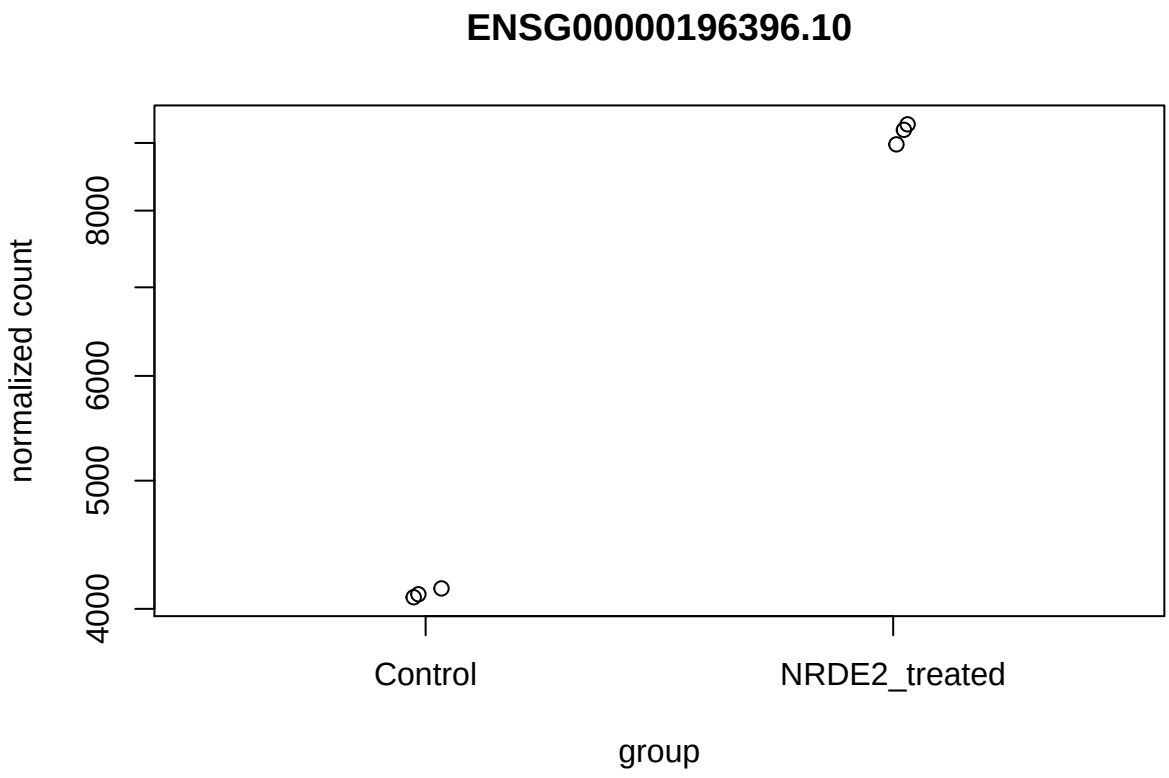

```
dds2<- estimateDispersions(ddsTxi, fitType = 'local')
plotDispEsts(dds2)
```



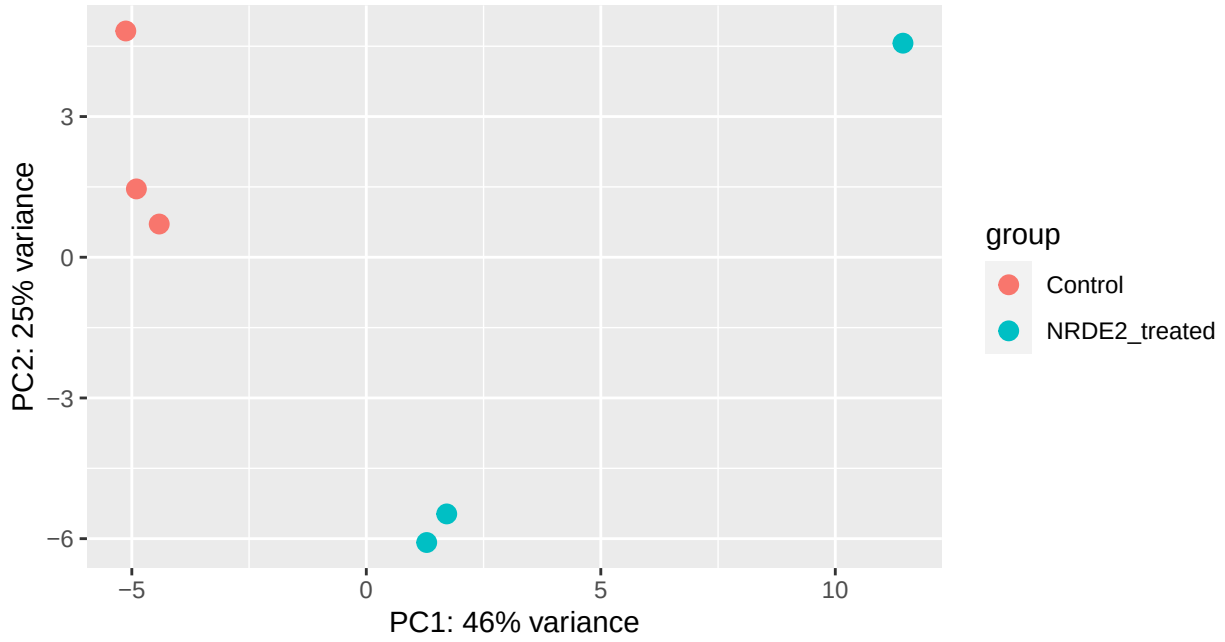

```
dds3<- estimateDispersions(ddsTxi, fitType = 'mean')
plotDispEsts(dds3)
```



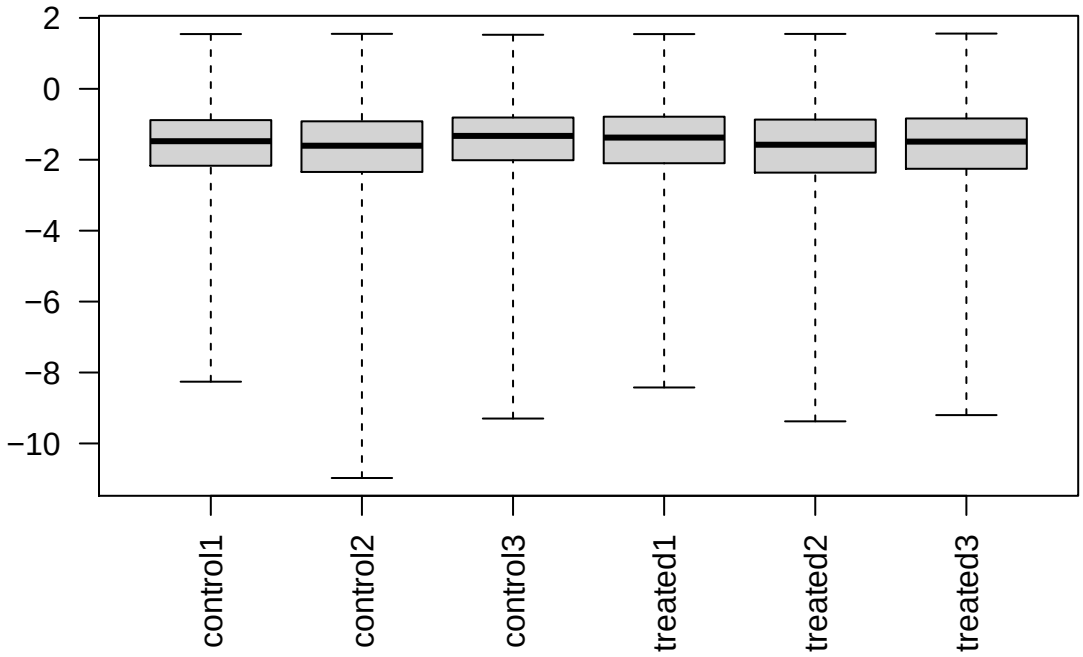

```
# lets see a plot of counts for the different conditions
plotCounts(ddsTxi, gene=which.min(res$padj), intgroup="condition")
```



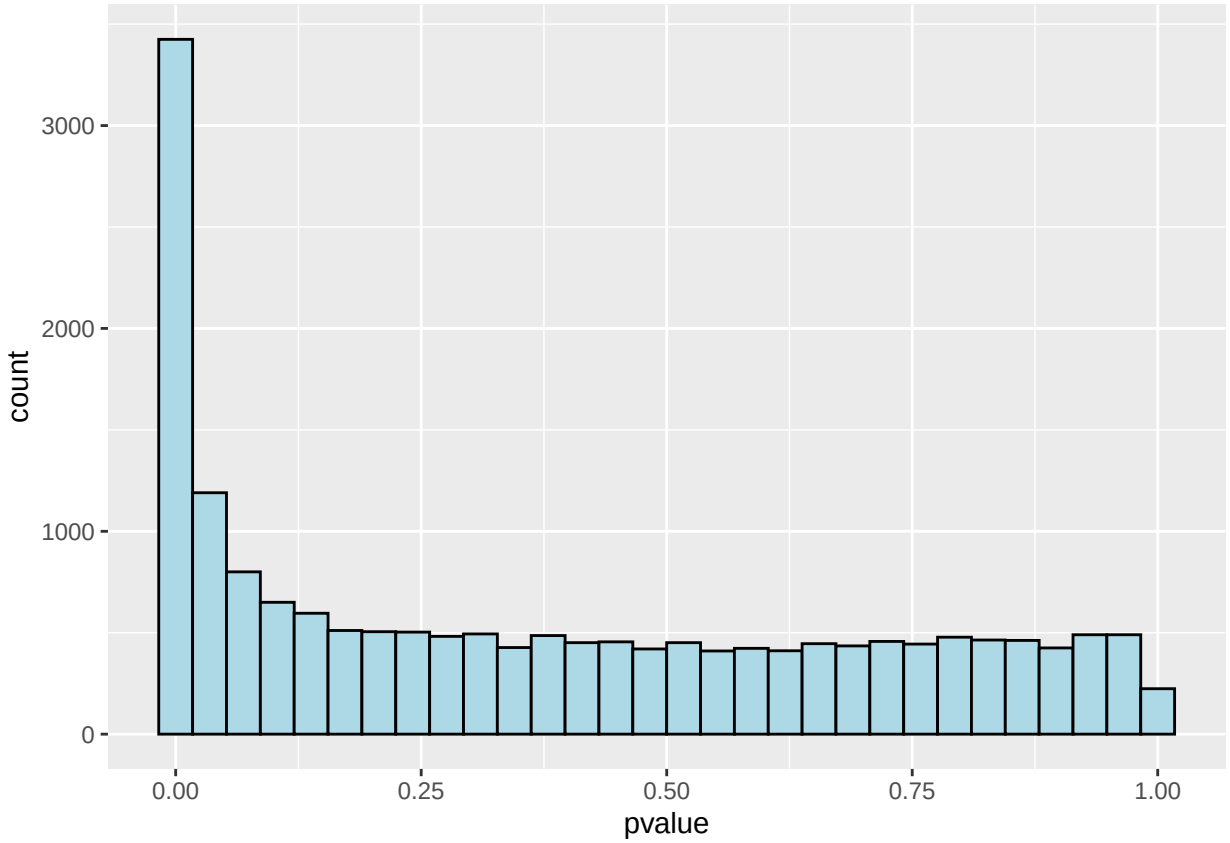

```
# plot PCA of the two groups
plotPCA(rld, intgroup = 'condition')
```




```
par(mar=c(8,5,2,2))
boxplot(log10(assays(ddsTxi)[["cooks"]]), range=0, las=2)
```




```
# histogram of raw p-value
library(ggplot2)
ggplot(as.data.frame(res_shrunk),aes(pvalue)) + geom_histogram(fill="light blue",color='black')
```




```
vsd <- vst(ddsTxi, blind=FALSE)
```

‘‘‘