Next Generation Sequencing
Final Project Option A
Kaitlyn Terrell

Introduction

RNA sequencing (RNA-seq) analysis allows us to gain a more comprehensive understanding of biological pathways, molecular mechanisms, and differential gene expression (DGE). DGE aids in identifying significant genes for key biological processes and drug targets. RNA-seq analysis of NRDE2-depleted (nuclear RNAi defective-2) breast cancer cells highlights the significance of these cells in their role as an RNA splicing factor, the depletion of which causes defects in mitotic progression and genome stability. Although the gene is not widely understood, Jiao, et al. have found that CEP131 (centrosomal protein 131) is a direct target of NRDE2 regulation, which plays a crucial role in centrosome function[1]. The cell line used for RNA-seq in this experiment are MDA-MB-231 cells (derived from metastatic human breast adenocarcinoma), which are widely used in biomedical research due to their aggressive proliferation. They were used here to study NRDE2, it's role in genomic stability, and its influence on mitotic progression. siRNA-mediated knockdown was used to induce NRDE2 depletion versus the untreated control, revealing that the cells expressed severe genomic instability and centrosome development defects post-knockdown[1].

Materials and Methods

The default trimmer TrimGalore was used by implementing the command "–skip_trimming false" in the nf-core/rnaseq script. We can see in the MultiQC report that under TrimGalore, two software's were used cutadapt and trimgalore (see Appendix for versions).

In the MultiQC report, we can see that the percent aligned for all 6 samples is above 90%, suggesting majority of the reads were successfully mapped to the reference. The only unusual aspects that stood out to me were in the FastQC: Status Checks plot, where Per Base Sequence count and Sequence Duplication were both red (no pass) for all 6 samples. Additionally, we can see in better detail that the individual plots for Per Base Sequence Content and the Sequence Duplication Levels show fail for all 6 samples as well. Otherwise, the rest of the MultiQC report appears nominal.

The workflow began with downloading the sample fastq files to be used in this analysis, which are single-end reads and were obtained from NextSeq Illumina sequencing targeting mRNA. Prior to running the nf-core/rnaseq workflow, a json file and a samplesheet csv file were produced. The json file includes parameters for resource allocation to run the nextflow workflow. Because the sequences are single-stranded, the samplesheet csv file only contained one fastq file per sample, where the samples were labeled as 'control' or 'treated' with their filepaths and auto strandedness set. A reference transcriptome was also retrieved utilizing code provided via the nf-core/rnaseq documentation page (see appendix)[2]. The nf-core/rnaseq pipeline (version 3.14.0) was run with the nextflow module (version 23.04.1). Within the script, file paths were provided as variables for any files used therein; Input contained the samplesheet csv file, fasta contained the transcriptome reference fasta file, gtf contained the genome annotation gtf file, output directory was set to res, default trimming was allowed (discussed above), extra_salmon_quant_args was used with GC Bias to ensure proper alignment, and the parameters were set with the json file previously discussed. After running, the output was located in the res folder, and contained several directories of information, including one containing the multiqc html report, one containing the execution_report html file, and the salmon directory which contained mapped read counts, sample information, and their respective quant.sf files to be used in R studio analysis.

In the R Studio RNA-seq analysis (version 4.3.3), the quant.sf output files from the nf-core/rnaseq workflow are imported using tximport, which allows them to then be processed with the DESeq2 package. The DESeq2 dataset is then made from the resulting metadata, which uses shrinkage to produce shrunken log-fold-change (LFC) estimates.

The DESeq2 package applies multiple test correction using the Benjamin-Hochberg (BH) method which is applied automatically during the DESeq2 analysis and adjusts p-values for multiple testing. This

occurs automatically in the results() command, as well as the lfcShrinkage() command.[3] Multiple test correction was done automatically via the results() command on the DESeq2 produced dataset. The BH method sorts p-values in ascending order; then critical values for each p-value are calculated from rank, total tests, and set FDR cutoff; p-values are then compared to their critical value, rejecting associated null hypotheses. By ensuring that the resulting false positives is less than the FDR cutoff, the BH method controls the false discovery rate (FDR).[4]

The LFC package command lfcShrink() with the type= apeglm method was used to shrink the dispersion estimates. The apeglm method is used for log-fold change in DESeq2, In DESeq2, the prior (probability distribution) is typically normally distributed, but the apeglm method adapts DESeq and adopts a Bayesian method to adjust the prior to a heavy-tailed Cauchy distribution. This adjustment allows for better shrinkage of the dispersion estimates.[5]

Results

Sample <chr>	Total_Input_Reads <dbl>	Total_Mapped_Reads <dbl>	Mapping_Rate <chr>
Control 1	61235909	55663891	90.90%
Control 2	63764300	58717637	92.09%
Control 3	55938770	52054034	93.06%
Treated 1	57677275	53290204	92.39%
Treated 2	58907058	54595800	92.68%
Treated 3	46379548	42975036	92.66%

Figure 1. Table of total number of reads, total mapped reads, and mapping rate per sample

Mapping rate, as shown in the table above and in the MultiQC report, is calculated by taking the total number of input reads over the total number of reads successfully aligned to our reference transcriptome (total mapped reads). Mapping rate, represented here in percentage, is above 90% for all six samples, suggesting high read quality (Figure 1). To locate these scores, one may navigate to the meta_info.json file in the res directory per sample (/res/salmon/<samplename>/aux_info/meta_info.json), where the number of reads are reported for individual samples. In each meta_info file, num_processed is the total input reads, and num_mapped is the number of mapped reads.

feature_id <chr>	Gene_Symbol <chr>	baseMean <dbl>	log2FoldChange <dbl>	pvalue <dbl>	padj <dbl>
ENSG00000175334	BANF1	6423.055	1.6560248	3.957045e-142	6.399333e-138
ENSG00000163041	H3-3A	7971.784	1.6644920	1.849499e-120	1.495505e-116
ENSG00000196396	PTPN1	6618.461	1.1491386	1.042439e-106	5.619444e-103
ENSG00000105976	MET	9565.747	1.5682263	3.514053e-96	1.420731e-92
ENSG00000128595	CALU	22967.130	1.4817134	1.788641e-95	5.785182e-92
ENSG00000101384	JAG1	11784.403	1.3114163	6.977230e-89	1.880596e-85
ENSG00000124333	VAMP7	2741.573	1.4829540	1.039576e-88	2.401719e-85
ENSG00000117632	STMN1	16768.455	1.3402658	4.086288e-82	8.260431e-79
ENSG00000180398	MCFD2	20587.749	0.9086143	1.124360e-70	2.020349e-67
ENSG00000213281	NRAS	6961.708	1.1806800	1.476286e-69	2.387450e-66

Figure 2. Table with 10 most highly significant DEGs.

The 10 most highly significant differentially expressed genes are represented in the table above, sorted by ascending adjusted p-value. The res.lfcShrink.tbl_df dataframe was converted into a tibble, which was sorted by adjusted p-value (padj). The corresponding Ensembl ID, located in the feature_id column, were used to retrieve the Gene symbol so that the top 10 genes can be easily observed. Here we see that BANF1 is the most highly significant differentially expressed gene, followed by H3-3A, and PTPN1 (Figure 2).

FDR < 0.05
<int>
3598

Figure 3. Number of statistically significant genes at the false discovery rate of 0.05.

The false discovery rate (FDR) of 0.05 was chosen as the cutoff to observe statistical significance in the dataset. Applying this FDR to the res.lfcShrink.tbl_df resulted in 3598 significant genes (Figure 3).

LFC < 0 count	LFC > 0 count
<dbl>	<dbl>
1667	1931

Figure 4. Number of genes with significantly higher and lower expression in RNAi vs. control for FDR 0.05

The number of genes that have higher expression after RNAi knockdown is found as 1931, while we see 1667 genes with decreased expression post-knockdown compared to the control (Figure 4).

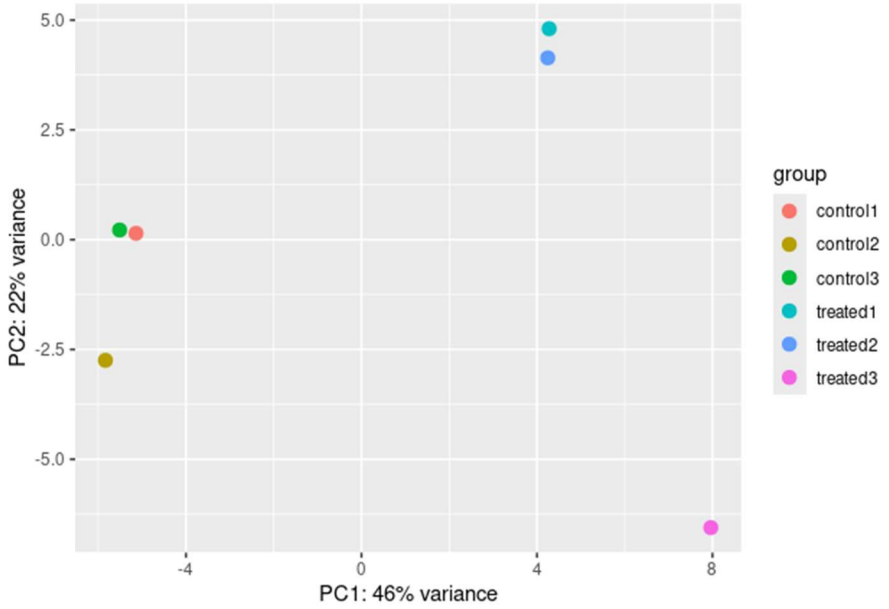

Figure 5. PCA plot showing relationships among all 6 samples.

The principal component analysis (PCA) plot shows comparison between PC1 (46% variance) and PC2 (22%) variance (Figure 5). All 3 control samples are clustered together in the PCA plot, suggesting similarity in gene expression levels, where control1 and 3 may have more similarity than control2 due to their proximity to each other. The positioning of the treated1 and 2 samples at positive coordinates suggests significant difference in gene expression compared to the control samples. The treated3 sample appears to be an outlier from the other treated and control samples, suggesting difference in gene expression to all other samples.

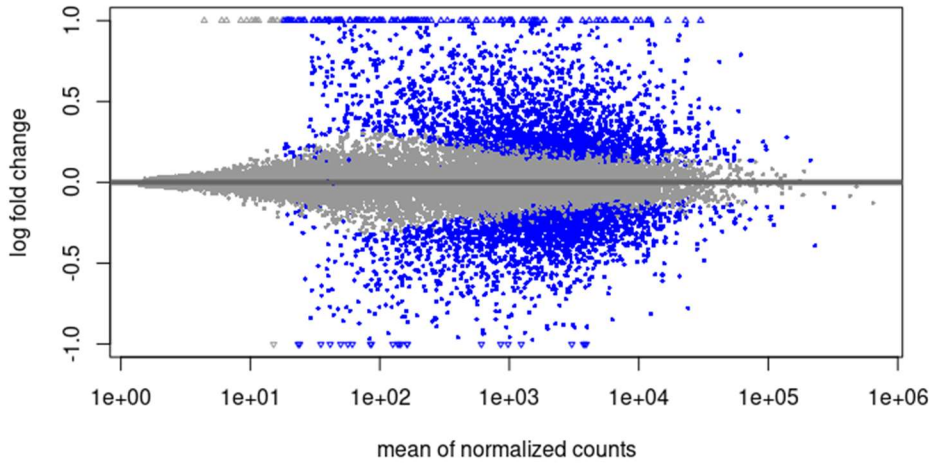

Figure 6. MA plot post-LFC shrinkage

The MA plot shows the log fold change (LFC) versus the mean expression for each gene, where the blue points represent statistically significant differentially expressed genes, and gray points are have no significant difference between control and treated samples. Post-LFC shrinkage, the magnitude of the log fold change is reduced significantly from pre-shrinkage, and the distribution of counts increases across the x-axis. From the plot above, we see that genes located closer to the 0.0 centerline on the y-axis are not significant, and we can infer that majority of the DEGs occur between a mean of 1e+01 and 1e+04, with several outliers beyond this maximum threshold. Positive vlaues on the y-axis indicate upregulation of the gene, and negative points indicate downregulated genes. Because we observe a fairly even distribution in both positive and negative directions on the y-axis, this suggests that there is similar upregulation and downregulation in genes for treatment vs control (Figure 6).

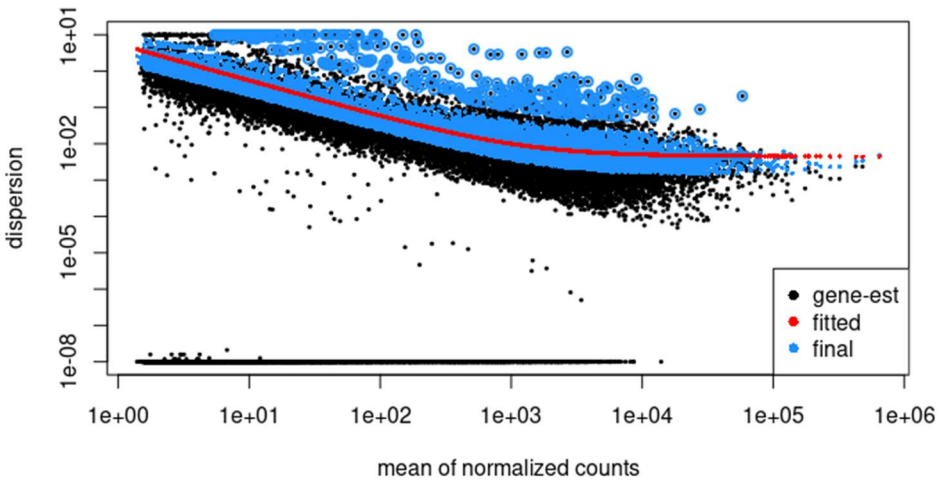

Figure 7. Dispersion by mean plot

Dispersion as a result of using DESeq2 is a parameter in variance, where dispersion is initially estimated using maximum likelihood estimation (MLE) and then shrunk toward the fitted line. Black points are original gene-wise dispersion estimates, blue points are values adjusted closer to the red fitted line, and black points circled in blue indicate genes with high dispersion that DESeq did not shrink due to potentially higher biological dispersion. We see above that the fitted line decreases gradually with increasing mean expression, suggesting that genes with higher expression have lower dispersion, supporting the negative binomial model that is used in the DESeq2 application (Figure 7).

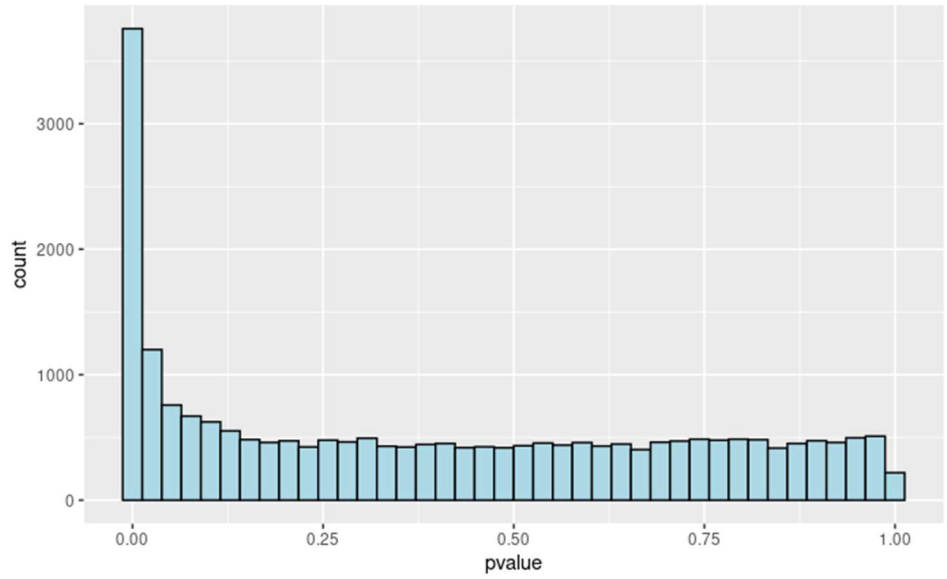

Figure 8. Histogram of raw p-value vs count

The histogram of raw p-values shows a high concentration of occurrence at and around a p-value of 0.0, suggesting the presence of many DEGs, likely with high significance. Beyond a p-value of 0.125, there appears to be no significant change in the number of genes, except for a decrease at p-value 1.00 where no DEGs are observed (Figure 8).

Discussion

The RNA-seq analysis of the MDA-MB-231 breast cancer cells with and without RNAi induced NRDE2 knockdown provides insight into molecular mechanisms, and significant DEGs upregulated and downregulated by the depletion of this gene. Despite issues observed in the FastQC portion of the MultiQC report, we observed that all treated and control samples had a mapping rate of over 90%, indicating high quality reads for use in analysis (Figure 1). The DESeq2 package was prominently used in this workflow to identify the top 10 significant DEGs between control and treated samples. After ranking in ascending order by adjusted p-value, the top 10 DEGs included: BANF1, H3-3A, PTPN1, MET, CALU, JAG1, VAMP7, STMN1, MCFD2, and NRAS (Figure 2). The principal component analysis (PCA) plot features distinct cluster patterns between control samples and treated, suggesting distinct differences in gene expression between conditions. The treated3 sample was observed as an outlier among the treated condition, highlighting variability post-NRDE2 depletion (Figure 5). Visualization of the amount of upregulated and downregulated DEGs can be seen in the MA plot for post-LFC shrinkage data. Here, the similar distribution of DEGs both negative and positive demonstrates the effects of NRDE2 depletion versus control in gene regulation (Figure 6). The dispersion by mean plot further shows the relationship of increasing gene expression levels experiencing gradual decrease in dispersion, which supports the statistical model used by DESeq2 (Figure 7). To further illustrate the amount and significance of DEGs in the dataset, a histogram of raw p-values was generated. From this plot, a high concentration of counts at low to 0 p-value provides support for the presence of significant DEGs (Figure 8). Overall, the RNA-seq analysis of breast cancer cells treated with RNAi knockdown of NRDE2 versus untreated control samples provides a foundation for understanding potential therapeutic targets, genes of interest, and molecular mechanisms related to the depletion of this gene.

Appendix

Executed to retrieve reference transcriptome: (bash)

```
latest_release=$(curl -s 'http://rest.ensembl.org/info/software?content-
type=application/json' | grep -o '"release":[0-9]*' | cut -d: -f2)



wget -L ftp://ftp.ensembl.org/pub/release-
${latest_release}/fasta/homo_sapiens/dna/Homo_sapiens.GRCh38.dna_sm.primary_assembly.fa.gz



wget -L ftp://ftp.ensembl.org/pub/release-
${latest_release}/gtf/homo_sapiens/Homo_sapiens.GRCh38.${latest_release}.gtf.gz
```

Executed to copy and run download.sh of fastq files and txt file: (bash)

```
cp /scratch/work/courses/BI7653/ProjectA.2024/download.sh .
sbatch download.sh

cp /scratch/work/courses/BI7653/ProjectA.2024/project_fastqs.txt .
cat project_fastqs.txt | less
```

Creating the samplesheet.csv file: (bash, using nano)

```
cp /scratch/work/courses/BI7653/hw9.2024/rnaseq.json .
nano rnaseq_samplesheet.csv

#in nano, made samplesheet for single-end sample files, using hw9 paired-end samplesheet for
reference

sample,fastq_1,strandedness
control1,/scratch/knt2039/ngs.finalA/SRR7819990.fastq.gz,auto
control2,/scratch/knt2039/ngs.finalA/SRR7819991.fastq.gz,auto
control3,/scratch/knt2039/ngs.finalA/SRR7819992.fastq.gz,auto
treated1,/scratch/knt2039/ngs.finalA/SRR7819993.fastq.gz,auto
treated2,/scratch/knt2039/ngs.finalA/SRR7819994.fastq.gz,auto
treated3,/scratch/knt2039/ngs.finalA/SRR7819995.fastq.gz,auto
```

nf-core/rnaseq Script: (bash, executed with slurm / sbatch)

```
#!/bin/bash
#
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=2
#SBATCH --time=24:00:00
#SBATCH --mem=8GB
#SBATCH --job-name=rnaseq_analysis
#SBATCH --mail-type=FAIL
#SBATCH --mail-user=knt2039@nyu.edu

module load nextflow/23.04.1


sample_sheet="/scratch/knt2039/ngs.finalA/rnaseq_samplesheet.csv"
genome_ref_fasta="/scratch/knt2039/ngs.finalA/Homo_sapiens.GRCh38.dna_sm.primary_assembly.fa.
       gz"
genome_annotation="/scratch/knt2039/ngs.finalA/Homo_sapiens.GRCh38.111.gtf.gz"
json_file="/scratch/knt2039/ngs.finalA/rnaseq_final.json"

nextflow run nf-core/rnaseq -r 3.14.0 \
    --input "$sample_sheet" \
    --fasta "$genome_ref_fasta" \
    --gtf "$genome_annotation" \
    --outdir res \
    --skip_trimming false \
    --extra_salmon_quant_args "--gcBias true" \
    -profile nyu_hpc \
    -params-file "$json_file"
```

Code in R Studio to run RNA-seq Analysis:

```
```{r}
library(tximport)
library(DESeq2)
#set up filepaths to find quant.sf files in salmon directory
netid <- 'knt2039'
sample_names <- c('control1','control2','control3','treated1','treated2','treated3')
sample_condition <- c(rep('control',3),rep('treated',3))
files <- file.path("/scratch",netid,"ngs.finalA","res","salmon",sample_names,'quant.sf')
names(files) <- sample_names
```

```{r}
#set up data as df
tx2gene <-
read.table(file.path("/scratch",netid,"ngs.finalA","res","salmon","tx2gene.tsv"),header=F,sep
="\t")
```

```{r}
#use tximport to obtain gene counts and other info
txi <- tximport(files, type="salmon", tx2gene=tx2gene)
```

```{r}
#create metadata table. First level,denominator, is reference for log2FC
metadata.df <- data.frame(sample = factor(sample_names),
                          condition = factor(sample_condition,levels = c('control','treated'))
)
metadata.df
```
```

```r
#make DESeq object using gene-level counts from Salmon-inferred TPMs per transcript.
dds <- DESeqDataSetFromTximport(txi,
                                colData = metadata.df,
                                design = ~ condition)
dds
```

```r
#extract matrix and view head
#needs magrittr package to run
library(magrittr)
counts(dds) %>%
  head()
```

```r
#show matrix dimensions
counts(dds) %>%
  dim()
```

```r
keep <- rowSums(counts(dds)) >= 10
dds <- dds[keep,]
counts(dds) %>%
  dim()
```

```r
#DESeq wrapper
dds <- DESeq(dds)
```

```r
#extract normalized count matrix
counts(dds,normalized=T) %>%
  head()
```

```r
# convert matrix to tibble, to simplify tidy operations
library(tidyverse)
normalizedcounts.tbl_df <- counts(dds,normalized=T) %>%
  as.data.frame() %>%
  rownames_to_column(var = 'feature_id') %>%
  as_tibble()

normalizedcounts.tbl_df
```

```r
normalizedcounts.long.tbl_df <- normalizedcounts.tbl_df %>%
                                pivot_longer( cols = -feature_id,
                                              names_to = 'sample',
                                              values_to = 'normalized_count')
normalizedcounts.long.tbl_df
```

```r
#visualize normalized counts in histogram per sample
normalizedcounts.long.tbl_df %>%
  ggplot(aes(x = normalized_count)) +
  geom_histogram(binwidth = 20) +
  xlim(0,1000) +
  ylim(0,7500) +
  facet_wrap( ~ sample, ncol = 4)
```

```r
#mean and variance for each gene. Plot avg of normalized counts and variance
normalizedcounts.long.tbl_df %>%
  filter(is.finite(normalized_count)) %>%
  group_by(feature_id) %>%
  summarise(mean = mean(normalized_count),
            variance = var(normalized_count)) %>%
  ggplot(aes(x = mean,y = variance)) +
    geom_point(size = .6) +
    scale_y_log10(limits = c(1,1e9)) +
    scale_x_log10(limits = c(1,1e9)) +
    geom_abline(intercept = 0, slope = 1, color="dark blue")
```

```r
#make the dds object into DESeqTransform class
rld <- rlog(dds)
rld
```

```r
#plotPCA(rld)
#differentiates samples seen in plotPCA(rld)
plotPCA(rld, intgroup = "sample")
```

```r
#numerator=treated, denominator=control in LFC
res <- results(dds, contrast = c('condition','treated','control'))
```

```r
#using apeglm for LFC shrinkage
resultsNames(dds)
res.lfcShrink <- lfcShrink(dds, coef = 'condition_treated_vs_control',type = 'apeglm')
```

```r
#no LFC shrinkage applied
plotMA(res)
```

```r
#LCF shrinkage applied
plotMA(res.lfcShrink)
```

```r
#dispersion plot
plotDispEsts(dds)
```

```r
mcols(res.lfcShrink)$description
```

```r
library(tidyverse)
#make shrunken DESeqResults into tibble
res.lfcShrink %>%
  as_tibble() %>%
  summarise(padj_NA = sum(is.na(padj)),
            padj_notNA = sum(!is.na(padj)))
```

```r
metadata(res.lfcShrink)$filterThreshold
```

```r
#histogram of shrunken data
res.lfcShrink %>%
  as_tibble() %>% # coerce to tibble
  ggplot(aes(pvalue)) +
  geom_histogram(fill="light blue",color='black',bins = 40)
```

```r
library(tidyverse)
#convert res.lfcShrink to tibble
res_df <- res.lfcShrink %>%
  as_tibble()

head(res_df)
```

```r
#produce table with reads and mapping rate for each sample
Sample <- c("Control 1", "Control 2", "Control 3", "Treated 1", "Treated 2", "Treated 3")
Total_Input_Reads <- c(61235909, 63764300, 55938770, 57677275, 58907058, 46379548)
Total_Mapped_Reads <- c(55663891, 58717637, 52054034, 53290204, 54595800, 42975036)
Mapping_Rate <- c("90.90%", "92.09%", "93.06%", "92.39%", "92.68%", "92.66%")
df <- data.frame(Sample, Total_Input_Reads, Total_Mapped_Reads, Mapping_Rate)
print(df)
```

```r
#top 10 genes, refined
res.lfcShrink.tbl_df <- res.lfcShrink %>%
  as.data.frame() %>%
  rownames_to_column(var = "feature_id") %>%
  as_tibble()
res.lfcShrink.sorted <- res.lfcShrink.tbl_df %>%
  arrange(padj)
top_10_genes <- head(res.lfcShrink.sorted, 10)
print(top_10_genes)
```

````
```{r}
#add gene symbol to dataframe
library(org.Hs.eg.db)
feature_id <- (top_10_genes[["feature_id"]])
gene_symbols <- mapIds(org.Hs.eg.db, keys = feature_id, column = "SYMBOL", keytype =
"ENSEMBL")
top_10_genes$Gene_Symbol = gene_symbols
#isolate columns of interest for top 10
top_10_df <- top_10_genes[, c("feature_id", "Gene_Symbol", "baseMean", "log2FoldChange",
"pvalue", "padj")]
print(top_10_df)
```
```{r}
#FDR top 10 genes (not used)
res.lfcShrink.tbl_df %>%
  filter(padj < 0.05) %>%
  arrange(padj)
```

```{r}
#number of statistically significant genes at your chosen FDR (0.05)
res.lfcShrink.tbl_df %>%
  summarise(`FDR < 0.05` = sum(padj < 0.05,na.rm = T))
```

```{r}
#the number of genes with signficantly higher and lower expression in RNAi vs. control
according to your chosen FDR
res.lfcShrink.tbl_df %>%
  mutate(`LFC < 0` = case_when(log2FoldChange < 0 & padj < 0.05 ~ 1, # add a column "LFC < 0"
and set to 1 if gene has LFC < 0 and FDR < 0.05
                               TRUE ~ 0)) %>%                          # and set to zero
otherwise
  mutate(`LFC > 0` = case_when(log2FoldChange > 0 & padj < 0.05 ~1,  # add a column "LFC < 0"
and set to 1 if gene has LFC > 0 and FDR < 0.05
                               TRUE ~ 0)) %>%                          # and set to zero
otherwise
  summarise(`LFC < 0 count`= sum(`LFC < 0`),
            `LFC > 0 count` = sum( `LFC > 0` ))
```
````

MultiQC nf-core/rnaseq software versions:

Process Name	Software	Version
CUSTOM_DUMPSOFTWAREVERSIONS	python	3.11.7
	yaml	5.4.1
CUSTOM_GETCHROMSIZES	getchromsizes	1.16.1
DESEQ2_QC_PSEUDO	bioconductor-deseq2	1.28.0
	r-base	4.0.3
FASTQC	fastqc	0.12.1
FQ_SUBSAMPLE	fq	0.9.1 (2022-02-22)
GTF2BED	perl	5.26.2
GTF_FILTER	python	3.9.5
GUNZIP_FASTA	gunzip	1.10
GUNZIP_GTF	gunzip	1.10
MAKE_TRANSCRIPTS_FASTA	rsem	1.3.1
	star	2.7.10a
SALMON_INDEX	salmon	1.10.1
SALMON_QUANT	salmon	1.10.1
SE_GENE	bioconductor-summarizedexperiment	1.24.0
	r-base	4.1.1
TRIMGALORE	cutadapt	3.4
	trimgalore	0.6.7
TX2GENE	python	3.9.5
TXIMPORT	bioconductor-tximeta	1.12.0
	r-base	4.1.1
Workflow	Nextflow	23.04.1
	nf-core/rnaseq	3.14.0

References

1. Jiao AL, Perales R, Umbreit NT, Haswell JR, Piper ME, Adams BD, Pellman D, Kennedy S, Slack FJ. 2019. Human nuclear RNAi-defective 2 (NRDE2) is an essential RNA splicing factor. RNA **25**: 352–363.doi:10.1261/rna.069773.118
2. nf-core/rnaseq Usage. https://nf-co.re/rnaseq/3.14.0/docs/usage#explicit-reference-file-specification-recommended
3. Love, M. I., Huber, W., & Anders, S. (2020). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, 21(1), 1-21. https://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html
4. Ignatiadis, Nikolaos, Bernd Klaus, Judith Zaugg, and Wolfgang Huber. 2016. “Data-Driven Hypothesis Weighting Increases Detection Power in Genome-Scale Multiple Testing.” *Nature Methods*. http://dx.doi.org/10.1038/nmeth.3885.
5. 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