# Appendix 2: XX vs XX\_DESeq\_Analysis

Here is an example of the code used for the pure temperature analysis. The temperatures XX can be replaced with any single tested temperature to make the code work for the other analyses of this type, and the outputs should go seamlessly into downstream analyses. Below is the example for the 25 °C pure temperature analysis.

## Load default parameters (Block 1)

knitr::opts\_chunk$set(echo = TRUE)

## Clear Global Environment (Block 2)

remove(list = ls())

## Setup and Installation (Block 3)

The first thing you need to do is install all of the packages necessary for the analysis. The following chunk of code is for installing the edgeR package and dependencies in a Windows environment. Unfortunatly when I tried installing all of the below packages on a MacBook Air, it did not work.

The first portion of the install was done using the CRAN database.

source("http://www.bioconductor.org/biocLite.R")  
biocLite("DESeq")  
biocLite("DESeq2", dependencies=TRUE)  
#biocLite("pasilla")  
#biocLite("IHC")  
biocLite("GenomeInfoDb")  
biocLite("SummarizedExperiment")  
biocLite("colorspace")  
biocLite("lazyeval")

## Accessing the Library and loading the read count table into the local environment (Block 4)

Since the remainder of the analysis does not differ in between Linux and Windows, the rest of the analysis was done in Windows. The first part of the actual analysis is to set the working directory. This can be done using the commands below:

#Set Library Directory  
#PC Path  
.libPaths(c("L:/RStudios/RPackRatLibLocations", "L:/RStudios/RPackRat\_2019\_04\_DESEQLibs"))  
  
#Set working directory  
#PC Path  
setwd("D:/Dropbox/Dropbox/Harrison Lab - Trevor Randall/RNASeq Analysis/RNASeqAnlyPkrat\_2020\_03/NullFiles/NullOutPut")  
#Laptop Path  
#Set Library Directory  
#.libPaths(c(""))  
#setwd("")  
#sink(file = "./RSessionRawRun.txt")

### Libraries needed (Block 5)

library("DESeq2")

**Loading and formatting the CF39 vs CF39S 25 C Dataset (Block 6)**

Load in the data from the file called “25 vs 25 Counts Table.csv”. This file should be located along the specified path.

#Load data data from my computer  
Exp25\_vs\_25 <- read.csv("../../SourceRNASEQCountsForDESEQ/25 vs 25 Counts Table.csv")  
TestSet <- Exp25\_vs\_25  
  
#Variables  
ColNameLables <- c("CF39S\_25-1","CF39S\_25-2", "CF39S\_25-3", "CF39C\_25-1","CF39C\_25-2", "CF39C\_25-3")

#### (Block 7)

# The dim command tells the dimensions of the dataframe.  
dim(TestSet)   
  
# The head command displays the first 6 lines of an R object in the console, unless the n parameter is used to specify the number to display.  
head(TestSet)   
  
# Another way of viewing the data. Note that the diminsions of the dataframe can also be seen here.  
str(TestSet)

Note that the data object Exp25\_vs\_25 has 7 columns. Further note that the first of the 7 columns is the locus. The locus contains 4 repeating loci, appended with a designation to indicate if the loci is for the “Intergenic\_sense”, “Intergenic\_antisense”, “Genic\_sense”, and “Genic\_antisense”. These were re-added in excel prior to exporting as a csv file to get the source file for R.

Finally, remove the unwanted Designation column, and replace the column numbers with the locus names.

#### (Block 8)

# The following command is used to label all of the rows using the gene/locus names. Essentially you are overwriting the column names with the locus names, and then deleting the column that contained the locus names.  
TestSet <- data.frame(TestSet[,-1], row.names=TestSet[,1])   
head(TestSet)  
dim(TestSet)

#### (Block 9)

# How to rename table columns, using the ColNamesLable string Variable  
colnames(TestSet) <- ColNameLables   
  
head(TestSet)

This should leave a read count table in the correct format to be read into a R list called a DGEList.

#### (Block 10)

# A method of using equal length names to name the groups in the treatment. Essentially the colnames command is extracting the pre-exsisting column names from the TestSet dataframe, substring is extracting the first 2 characters of the column names, as specified by the characters inclusive between character 1 and 2 (so SC and ST), and finally these characters are being turned into factors.  
TreatmentNames <- substring(colnames(x = TestSet), first = 1, last = 5)  
TreatmentNames <- unique(TreatmentNames)  
Treatment <- factor(substring(colnames(x = TestSet), first = 1, last = 5))   
  
Treatment # This tells you that you have a factor of 6 with 2 levels.

#### (Block 11)

expt\_design <- data.frame(row.names = colnames(Treatment), condition = c("CF39S","CF39S","CF39S","CF39C","CF39C","CF39C"))  
expt\_design  
  
Check1 <- TestSet  
names <- colnames(Check1)  
dataCheck <- cbind(names, expt\_design)  
head(dataCheck)  
  
if((all(rownames(dataCheck$names) %in% colnames(TestSet)))==(all(rownames(dataCheck$names) == colnames(TestSet))))  
{  
 TestSetCkd <- TestSet  
} else {  
 FailedMessage <- "The order of the file is not correct"  
 FailedMessage  
}  
  
conditions = expt\_design$condition  
conditions  
  
DataDFDESeq <-   
 DESeqDataSetFromMatrix(countData = TestSetCkd, colData = expt\_design, design = ~condition)

## Prefiltering Data and cahnge reference levels

We want to keep rows that have 10 or more reads in them

#### (Block 12)

keep <- rowSums(counts(DataDFDESeq)) >= 10  
DataDFDESeq <- DataDFDESeq[keep,]  
  
DataDFDESeq$condition <- relevel(DataDFDESeq$condition, ref = "CF39C")  
  
DataDFDESeq$condition <- droplevels(DataDFDESeq$condition)

#Differential Expression Analysis and File Output

#### (Block 13)

DataDFDESeqRes <- DESeq(DataDFDESeq)  
Res <- results(DataDFDESeqRes)  
Res  
  
d2 <- Res  
ID <- rownames(Res)  
rownames(d2) <- NULL  
Res2 <- cbind(ID, d2)  
  
ResDataTable = as.data.frame(Res2)  
ResDataTable  
  
  
write.table(ResDataTable,file = "DESeqNew\_results.txt", sep = "\t", row.names = F)

#Log Fold change shrinkage for visualization and ranking

#### (Block 14)

resultsNames(DataDFDESeqRes)  
  
ResLFC <- lfcShrink(DataDFDESeqRes, coef = "condition\_CF39S\_vs\_CF39C", type = "apeglm")  
ResLFC  
  
d2 <- ResLFC  
ID <- rownames(ResLFC)  
rownames(d2) <- NULL  
ResLFC\_Alt <- cbind(ID, d2)  
colnames(ResLFC\_Alt) <- c("locus", "baseMean", "log2FoldChange", "lfcSE", "pvalue", "padj")  
ResLFC\_Alt  
write.table(ResLFC\_Alt,file = "DESeq\_results\_LCFShrinkResults-apeglm.txt", sep = "\t", row.names = FALSE)  
  
ResNorm <- lfcShrink(DataDFDESeqRes, coef = "condition\_CF39S\_vs\_CF39C", type = "normal")  
ResNorm  
  
d2 <- ResNorm   
ID <- rownames(ResNorm)  
rownames(d2) <- NULL  
ResNorm\_Alt <- cbind(ID, d2)  
colnames(ResNorm\_Alt) <- c("locus", "baseMean", "log2FoldChange", "lfcSE", "stat", "pvalue", "padj")  
ResNorm\_Alt  
write.table(ResNorm\_Alt,file = "DESeq\_results\_LCFShrinkResults-normal.txt", sep = "\t", row.names = FALSE)  
  
#If you use this data cite ashy package  
ResAsh <- lfcShrink(DataDFDESeqRes, coef = "condition\_CF39S\_vs\_CF39C", type = "ashr")  
ResAsh  
  
d2 <- ResAsh  
ID <- rownames(ResAsh)  
rownames(d2) <- NULL  
ResAsh\_Alt <- cbind(ID, d2)  
colnames(ResAsh\_Alt) <- c("locus", "baseMean", "log2FoldChange", "lfcSE", "pvalue", "padj")  
ResAsh\_Alt  
write.table(ResAsh\_Alt,file = "DESeq\_results\_LCFShrinkResults-ashr.txt", sep = "\t", row.names = FALSE)

### Figure Generation (Block 15)

### MA plot

dev.off() # Turns off any previously on graphical devices  
png(filename = "MyDESeq MA plot.png",width = 11, height = 9, res = 300, units = "in")  
plotMA(DataDFDESeqRes, ylim=c(-4,4))  
dev.off()

### Plot other shrinkage plots (Block 16)

dev.off() # Turns off any previously on graphical devices  
png(filename = "MyDESeq ResLFC MA plot.png",width = 11, height = 9, res = 300, units = "in")  
  
plotMA(ResLFC, ylim=c(-4,3))  
#idx <- identify(Res$baseMean, Res$log2FoldChange)  
#rownames(Res)[idx] #Allows you to lable (and identify) certain points on the graph  
  
par(mfrow=c(1,3), mar=c(4,4,2,1))  
xlim <- c(1,1e5); ylim <- c(-3,3)  
plotMA(ResLFC, xlim=xlim, ylim=ylim, main="apeglm")  
plotMA(ResNorm, xlim=xlim, ylim=ylim, main="normal")  
plotMA(ResAsh, xlim=xlim, ylim=ylim, main="ashr")  
  
par(mfrow = c(1,1)) # Good practice to immediatly reset plot sizing  
  
dev.off()

### Plot Counts (for looking at individual genes) (Block 17)

#par(mfrow=c(1,1))  
#plotCounts(DataDFDESeqRes, gene = which.min(Res$padj), intgroup = "condition")

### Session Info (Block 18)

sink(file = "./SessionInfo.txt")  
sessionInfo()  
sink(file = NULL)

### References (Block 19)

The citation function can be used to who you should be citing.

sink(file = "./SessionCitations.txt")  
  
citation("DESeq2")  
citation("apeglm")  
citation("lattice")  
citation("stringi")  
  
# Cite if ResAsh plot was used  
citation("ashr")  
  
# Cite if apeglm plot was used  
citation("apeglm")  
  
sink(file = NULL)

Love, M.I., Huber, W., Anders, S. (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biology, 15:550. 10.1186/s13059-014-0550-8

Modelled from the following page (dated 2019/01/04): <https://www.bioconductor.org/packages/devel/bioc/vignettes/DESeq2/inst/doc/DESeq2.html>

Modelled from the following page (dated 2019/01/04): <https://rstudio-pubs-static.s3.amazonaws.com/13301_6641d73cfac741a59c0a851feb99e98b.html?fbclid=IwAR1h2pL9V1tblTk4z9E1JX9udrel2ILE_WKKp2M8Kjd6HB0ZC6wwctFu61Y>