**Response to Reviewers**

Major concerns were identified by reviewers in five areas of our proposal:

1. Innovation:

Our research aims to be innovative in the end goal of exploring DNA viruses as biological control for dealing with the significant issue of increasingly prevalent harmful algal blooms. The immediate result of this specific project is unlikely to be innovative due to the scope of the project. However, investigating the genes differentially regulated between susceptible and resistant algal species is the first step to determining mechanisms for algal persistence alongside DNA viruses.

1. Investigator Roles:

We chose to assign ourselves individual pipeline steps instead of all taking a proportion of the data for efficiency. If we run into time management concerns, as was questioned by the reviewers, we believe the shared-analysis approach may compound issues if one individual is consistently late with their contributions. While this could be solved by meeting together, coordinating schedules could also delay the pipeline. Other group members will aid in troubleshooting if analysis is halted at a given step. To reduce error and bias, all members will review the code and parameters. Due to concerns, we will revise our approach for internal conflict resolution; instead of voting, we will replace this strategy with group discussion and still pursue mediation in cases where a consensus cannot be reached.

1. Approach:

Reviewers requested more detailed information on pipeline steps and how they were being used. Quality control and trimming are omitted here as they were addressed in the proposal, with distinct goals for each. HISAT2 will be used to first generate a genomic index with consideration for exons and splice-sites. This genomic index will be used to align the transcripts in our RNA-Seq data to inform transcript assembly. As we are using a Eukaryotic data set, splicing and exon information is integral to the alignment process. Using only a fasta-formatted genomic reference would be adequate for prokaryotic data sets, but introns would obfuscate eukaryotic transcript alignment. Transcript assembly is to be conducted through StringTie to take our RNA-Seq reads and create transcripts from the splice-aware HISAT output. StringTie is conducted in three steps to assemble against the reference genome, merge transcripts, and then assemble again using the complete transcript list in order to create a ballgown appropriate output for expression quantitation while still allowing for novel transcript discovery. The StringTie output will then be imported into Ballgown (RStudio) to compile the output from each sample, gather transcript and gene information, FPKM, and the associated statistical outputs. Manipulation of the ballgown object and post-processing will allow for generation of figures to visualize the results of our analytical pipeline. As we are primarily looking for the genes over or under-expressed in our resistant clones compared to the susceptible samples, we will focus on differential expression analyses. We intend to sort our data based on these objectives and visually represent our results through a heat map and complementary figures.

1. Anticipated Errors and Alternate Steps:

We did not anticipate errors occurring in the software choice as packages were selected based on its reputation as an established analytical pipeline for RNA-Seq data. Although there are alternative packages that would be adequate for use with this data set, we anticipated errors arising from human error or shortfalls of the data itself. The genome contains 22 scaffolds for all 20 chromosomes, a plasmid, and the mitochondrion; this suggests completeness, but there still may be missing data or incomplete annotations. We attempted to account for this by using the most recent version of the genome and utilizing additional StringTie steps to allow discovery of novel transcripts. As errors within the HISAT2 and StringTie output are hard to detect prior to the generation of the read coverage tables, we also expedited our analysis so that we could go back and redo portions or adjust parameters if mapping or assembly did not go according to plan.

1. Analytical Goals: “There was not much discussion of full analysis of what genes are wanting to be viewed.”

We expect to see genes related to viral defense globally upregulated in all samples due to the treatment condition (large DNA virus inoculation). We are interested specifically in examining the differential gene expression that will result from viral resistance versus susceptibility. Due to the limited scope of this particular project, we expect this approach to be hypothesis generating for future experiments rather than to examine a pre-defined subset of genes. To examine differential expression, we plan to generate volcano plots and heat maps based on significant gene results. We hope to utilize these results to find new candidates for future studies and will inform following work based on the expression patterns observed here.

**Results**

*Data Retrieval, FastQC, and Trimming*

Paired-end RNA-Seq data for OtV5-inoculated *Ostreococcus tauri* was retrieved from NCBI BioProject PRJNA344946. The reference genome ([GCA\_000214015.2](https://www.ncbi.nlm.nih.gov/assembly/GCA_000214015.2)*;* version 140606) was retrieved from NCBI’s Genome database. Before being uploaded to Spruce, all files were renamed to designate Resistance/Susceptibility and isolate number to simplify analytical data structuring.

Initial FastQC was run for all transcriptomic data sets (4 susceptible, 19 resistant). Due to the high file size for the paired RNA-Seq data, FastQC was run on a single processor per node with all the available memory (36gb) specified to that node using the pvmem #PBS command (see pipeline code).

We did not conduct data trimming; based on our FastQC results (Fig 1), all sample data was comprised of only 100 bp reads and maintained high average phred quality across the entire length of the reads. No adapter sequences were found in any samples. Analysis proceeded with all four susceptible clones (S5 [SRA: SRS1720283], S4 [SRA: SRS1720285], S3 [SRA: SRS1720284], and S2 [SRA: SRS1720283]) and four resistant clones with high average sequence quality and large total read count (R4 [SRA: SRS1720271], R6 [SRA: SRS1720287], R8 [SRA: SRS1720288], and R14 [SRA: SRS1720291]). As all the isolates used in this study were clonal; this reduced the need for a large number of biological replicates in each condition.

Utilizing the largest of the high quality resistant datasets increased coverage- and thus sensitivity- of the downstream analysis. While coverage can be hard to estimate when comparing transcriptomic data sets to the genome, the combined size of the resistant and susceptible transcriptome sets comprised 546x genomic coverage (7.6Gpb resistant/13.9Mbp genome; each sample contributed approximately 135x coverage) and 342x coverage (4.76 Gbp susceptible/13.9Mbp genome; ~85x genomic coverage per sample), respectively.

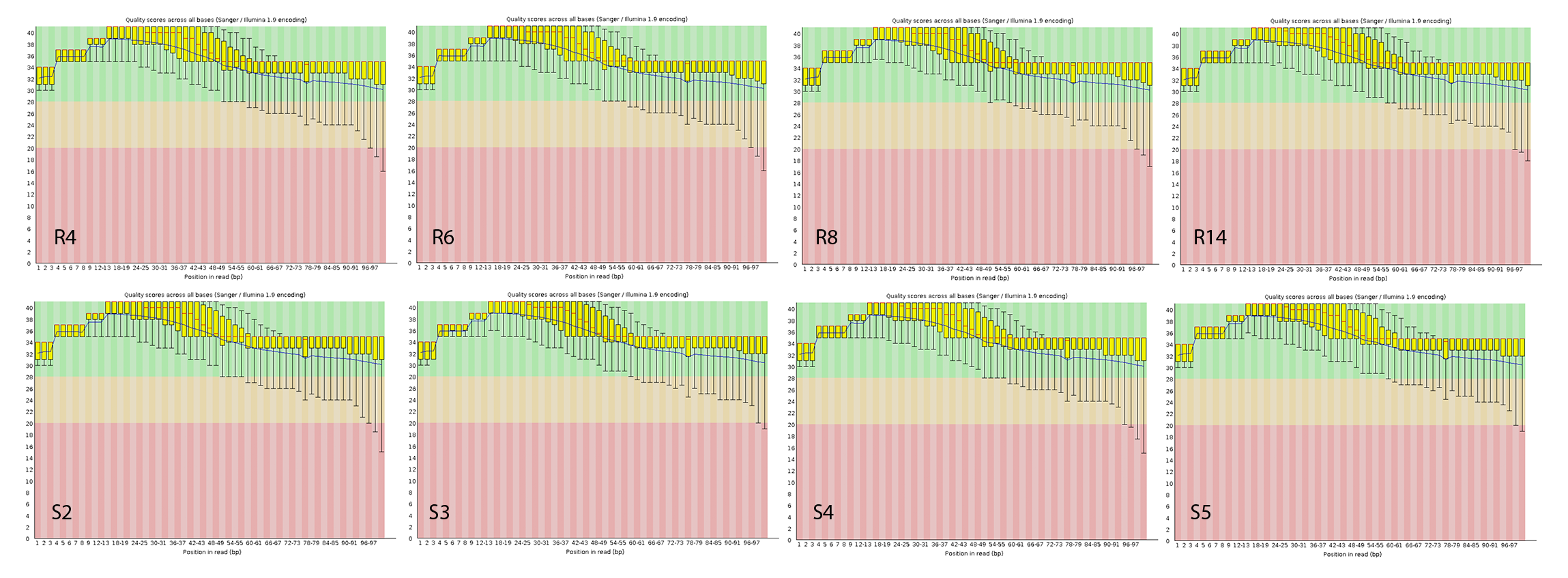


Figure 1. Average quality score across read length of samples used for analysis. The vertical axis shows phred scores along sequence length (horizontal). Average quality score is shown for all resistant samples (R4, R6, R8, and R14; top row) and susceptible samples (S2, S3, S4, and S5; bottom row) after initial FastQC.

*HISAT2*

*O. tauri* transcripts were re-acquired from the European Nucleotide Archive for the chosen run sets (SRA accessions as listed above) to obtain separated forward and reverse files for the selected isolates. The reference genome .gff file was used to generate exon and splice-site information for splice-aware mapping. HISAT2-build and the genomic fasta data was used in tandem with the exon and splice-site files to generate a genomic index. HISAT2 was run for each RNA-Seq data set to align the transcripts to the splice-aware indexed genome; alignments for each sample were output as .sam files.

*StringTie*

The HISAT2 output was converted from .sam files to .bam files using SamTools. SamTools version 0.1.19 required paired use of sort and view functions to first convert to bam and then sort the bam output. The sort.bam transcripts were then assembled using StringTie with the reference genome (with annotation data). Output file names were compiled into a text document (each row only containing a single file name; see Supplemental Data) and uploaded to Spruce; this document was used in the StringTie --merge function with the reference genome to merge initial stringtie output into a single .gtf file of all transcripts. The sorted bam files were then again run against the merged file (using the -B/e arguments) and output to sample-specific directories to generate .ctab and .gtf output for import and analysis in Ballgown. The final output was downloaded from the Spruce HPC environment for import into RStudio.

*Ballgown*

The StringTie output directories were downloaded from Spruce and placed into a parent directory. This file structure was used to import all .ctab and .gtf files into RStudio where the package ballgown was used to generate data tables and sort transcripts by gene annotation and FPKM values. Ballgown was conducted entirely in RStudio using R Markdown. A separate csv file (titled refrence.csv) was uploaded with information about our samples. This can be located at the bottom of the document. A Ballgown object was then created by Ballgown utilizing the output from Stringtie and refrence.csv. Next, we filtered the data to remove low-abundance genes. Transcripts and genes with significant differences between groups were identified fold changes were obtained. Transcripts and gene lists were then sorted by p-value and exported as a csv.

*Post-Processing*

Plots were first generated to examine data distribution before examining differential gene expression. Variance across gene abundance for each sample was evaluated by generating box plots for the adjusted FPKM (Fragments Per Kilobase of transcript per Million read pairs; adjusted using log2(FPKM+1) for cross-sample comparison) (Fig 2). Even data distribution and means between samples suggested no issues in individual RNA-Seq data sets. Length of transcripts and their distribution (Fig 3) showed most assembled transcripts were below 5kbp in length. Our transcripts were then compared to their associated genes to retrieve the number of transcripts per gene (Fig 4). By evaluating the number of transcripts associated with each gene, this approximated alternate splice variants and representation within our data set.

When evaluating results, we chose to focus on the genes expressed rather than transcripts as 1) most genes contained a single transcript and 2) we are more interested in informing differences in general biological or metabolic function in future analyses rather than the differences in isoforms between conditions. From our gene results, we created a volcano plot (Fig 5) examining the log fold change of our expressed genes against their q-values. This plot located the most significant genes experiencing the highest fold change (highlighted in green). Our expressed genes were then sorted by significance (p<0.05) to examine differences in expression between all samples. A heatmap (Fig 6) was generated from the significantly expressed genes by using the associated FPKM value as a measure of relative expression. Our results show distinct clustering of significant genes as up and down regulated across samples. Clustering between samples sorted Resistant clone 6 (R6) with susceptible isolates.

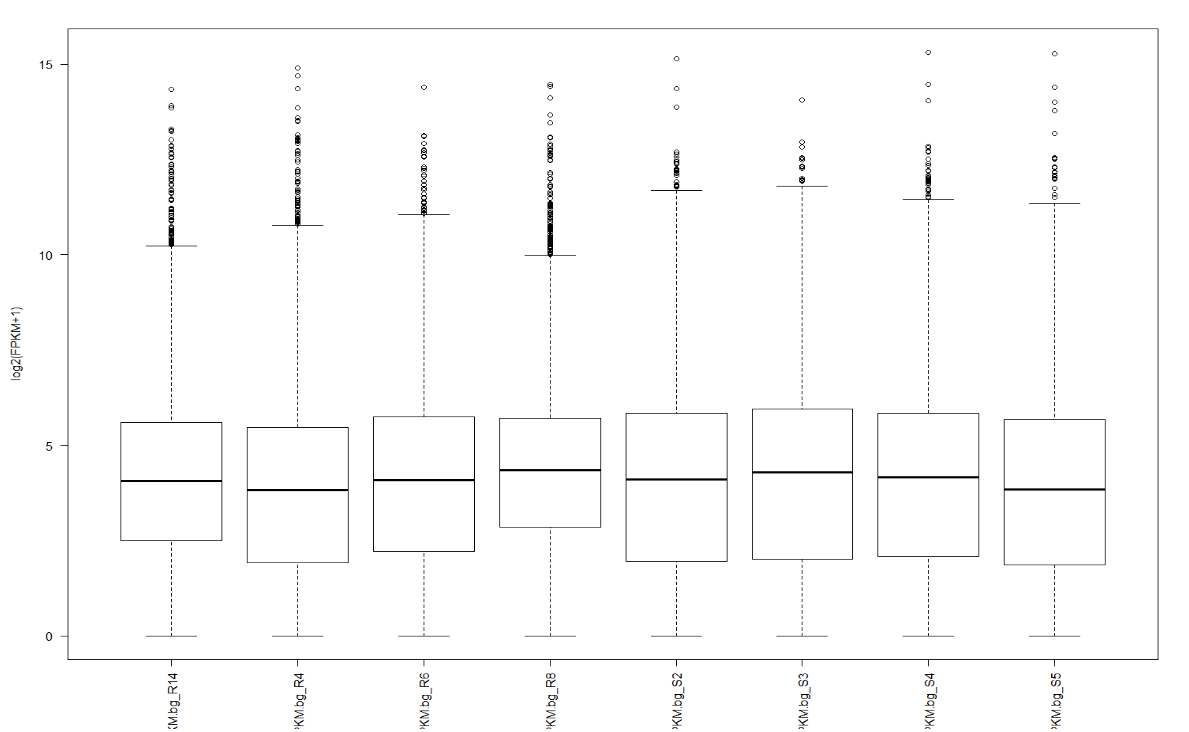


Figure 2: Evaluating Gene Abundance; Log2(FPKM+1) box plot across samples. FPKM was normalized to account for variance of transcriptomic samples.

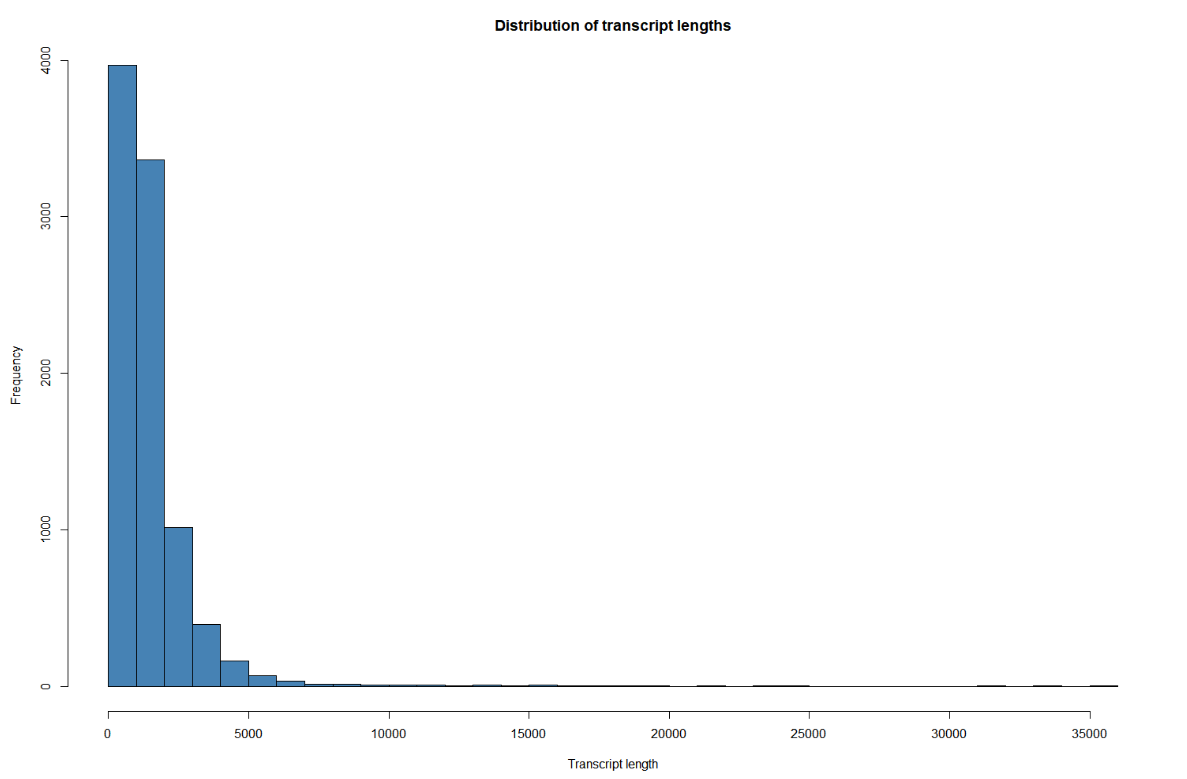


Figure 3. Distribution of Transcript Lengths. Length of transcripts across all samples were sorted by frequency (count).

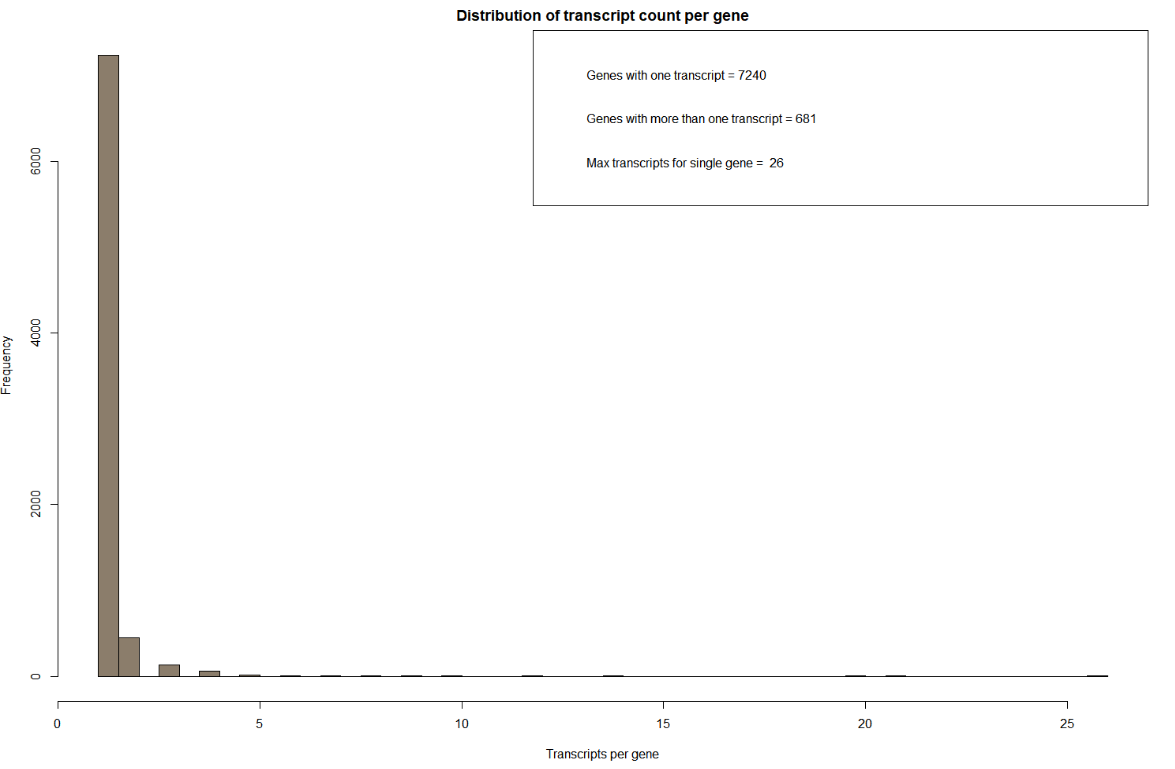


Figure 4. Distribution of Transcript Counts per Gene. Transcripts were aggregated and sorted by gene to examine the distribution of gene expression. A majority of genes (7,240) only had one transcript. 681 genes had more than one transcript with the highest number of transcripts per a single gene being 26.

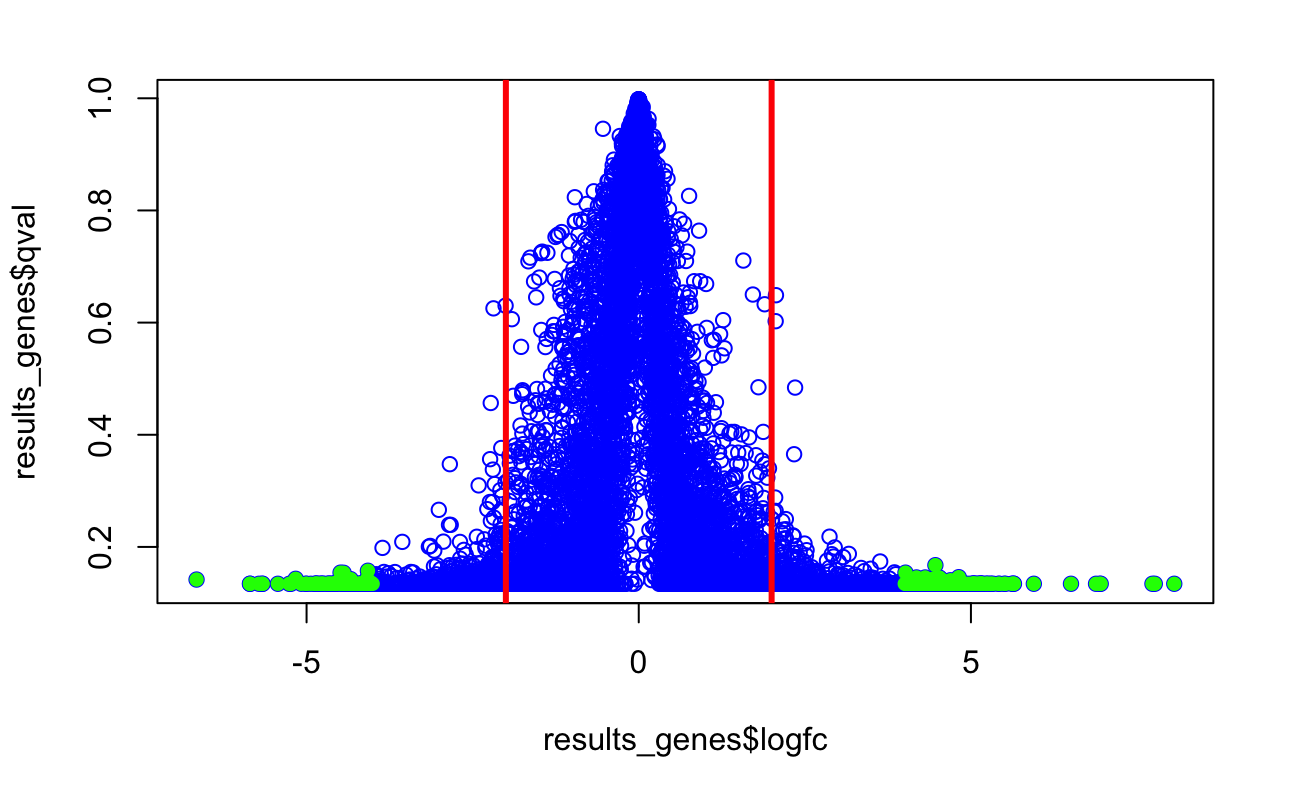


Figure 5. Volcano Plot. Fold change is plotted against the q-value of expressed genes. Genes highlighted in green are those displaying the highest fold change at the lowest q-values (indicating significance). Red lines represent 2x fold change.

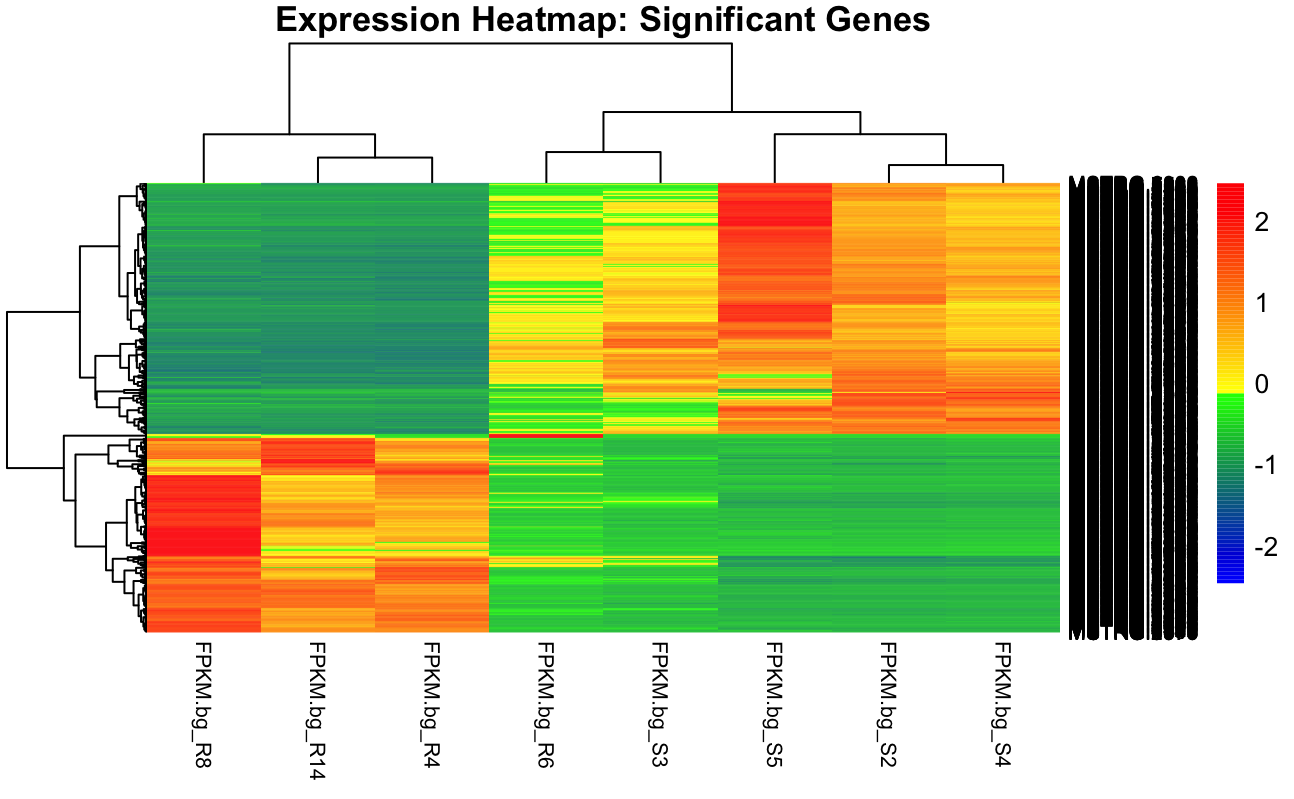


Figure 6. Heat map of significantly expressed genes (p<0.05). Degree of change calculated by FPKM values for each treatment. The yellow-to-red spectrum represents upregulated genes while green-to-blue represents downregulation.

**Discussion**

*Outcomes*

A very low proportion of our genes (8.6%) expressed more than a single transcript as captured by our analysis. Many genes are often expressed at very low levels within the cellular environment; we did not expect to recover a high proportion of splice variants in our analysis. Most expressed transcripts were below 5kbp in length, but small amounts of long transcripts can be observed. This does not suggest marked error in transcript assembly.

Differences in expression were ultimately sorted by p-value and/or adjusted p-value (q-value) to discern relevant results. We attempted to discern potentially significant genes through several different tools. The first method was in generating a volcano plot; this examined fold change of genes across q-value. By sorting the data in this manner, we were able to highlight the most significant, highly expressed genes of our data set. As q-values were higher than expected (sorting by q<0.05 returned no results), we also examined gene expression by unadjusted p-value. Significantly expressed genes were selected at p<0.05 for heat map generation. With this definition, we retrieved 1,290 significantly expressed genes. Clustering of genes revealed distinct groups of up and downregulated genes between susceptible and resistant algae, which suggests distinct biological functions may be differentially regulated between conditions under OtV5 infection. One resistant sample showed diverging gene expression patterns in our heat map; hierarchical clustering suggests sample R6 is more similar in expression pattern to the susceptible samples. This sample will be revisited in culture and sequence data. Future work will utilize more of the available resistant data sets to determine whether other resistant algal clones display susceptible-like patterns of gene expression when exposed to OtV5. Analysis may benefit from the generation of additional susceptible data sets, as well; increased sample size will allow for greater sensitivity, reduce the chance of false positives (reduce q-values), and increase result significance.

Overall, the suggestion that distinct biological processes may be active between resistant versus susceptible *O. tauri* creates a starting point to evaluate the molecular bases of algal viral resistance. To further examine this trend in the data, future work will involve Gene Ontology annotation on significant genes from the volcano plot analysis as well as pathway analysis for genes determined significantly expressed and present in the heat map. This will allow us to explore the different cellular activity characterized by large DNA virus resistance and susceptibility.

*Troubleshooting*

Jobs for FastQC initially timed out in the queue even with a single (interleaved) file run. This was fixed by specifying only one processor per node and directing all the available memory (36gb) to that processor. As FastQC command line cannot specify threads, this accelerated analysis and solved the walltime termination.

The latter portion of our pipeline was also rerun due to skipping the complete HISAT2 index building during our first attempt. Due to the annotated genome file being a .gff instead of .gtf, HISAT2 was unable to properly extract splice-site and exon data. Dr. Driscoll manually retrieved this information from the *O. tauri* reference, as described in the pipeline code. Once these files were generated, the genomic index was rebuilt and the remainder of the pipeline completed.

Some trouble was also encountered during file import using Ballgown. The ballgown compiler required a .csv file with the sample directory names as they appeared in the parent ballgown directory. There were multiple errors that directory order did not agree; we had to use the order function to sort the first column to successfully import our StringTie output. Other small errors were encountered in R due to it being a less familiar language.

*Learning Outcomes*

As many steps of this process were used before (FastQC/Trimmomatic/Samtools) or were reminiscent of previous processes (mapping a file to a reference genome), we encountered fewer technical issues in the analytical pipeline than in previous projects. However, post-processing in R created a learning curve. Though several group members had some RStudio experience, the errors were often more ambiguous than those encountered during Spruce job submission. Ultimately, we were able to work past them, but the inclusion of the additional language exposed us to new material during this project. Working in R slowed our post-processing and provided a challenge. This biggest learning opportunity came from post-processing; having to design our data analysis and figure generation approach made us think critically about our data set and what we wanted to say about our output. In previous work, interpretable output was generated directly from our analysis pipeline. Though there were additional ways in which we could choose to organize and display our results, this project required much more hands-on design. This is a skill that will be universally applicable to future research.

**Pipeline Code**

#! /bin/bash

## FastQC

# The following PBS argument was modified as follows:

#PBS -l nodes=1:ppn=1,pvmem=36gb

# FastQC for interleaved data sets

fastqc r3.fastq.gz

fastqc r4.fastq.gz

fastqc r5.fastq.gz

fastqc r6.fastq.gz

fastqc r8.fastq.gz

fastqc r12.fastq.gz

fastqc r12a.fastq.gz

fastqc r13.fastq.gz

fastqc r13a.fastq.gz

fastqc r14.fastq.gz

fastqc r14a.fastq.gz

fastqc r15.fastq.gz

fastqc r15a.fastq.gz

fastqc r16.fastq.gz

fastqc r24.fastq.gz

fastqc r25.fastq.gz

fastqc r26.fastq.gz

fastqc r27.fastq.gz

fastqc r28.fastq.gz

fastqc s2.fastq.gz

fastqc s3.fastq.gz

fastqc s4.fastq.gz

fastqc s5.fastq.gz

## HISAT2

# exon building from gff file

awk '{if ($3=="exon") {print $1"\t"$4-1"\t"$5-1"\t"$7}}' GCF\_000214015.3\_version\_140606\_genomic.gff > tauri.exon

# introns added to gff file

gt gff3 -retainids -addintrons < GCF\_000214015.3\_version\_140606\_genomic.gff > tauri\_with\_introns.gff

awk '{if ($3=="intron") {print $1"\t"$4-2"\t"$5"\t"$7}}' tauri\_with\_introns.gff > tauri.ss

# Hisat2 index building

hisat2-build --ss tauri.ss --exon tauri.exon -f gene.fna gene\_ind

# Hisat2

hisat2 -p 12 --dta -x gene\_ind -1 R14\_1.fastq.gz -2 R14\_2.fastq.gz -S R14.sam

hisat2 -p 12 --dta -x gene\_ind -1 R8\_1.fastq.gz -2 R8\_2.fastq.gz -S R8.sam

hisat2 -p 12 --dta -x gene\_ind -1 R6\_1.fastq.gz -2 R6\_2.fastq.gz -S R6.sam

hisat2 -p 12 --dta -x gene\_ind -1 R4\_1.fastq.gz -2 R4\_2.fastq.gz -S R4.sam

hisat2 -p 12 --dta -x gene\_ind -1 S2a\_1.fastq.gz -2 S2a\_2.fastq.gz -S S2.sam

hisat2 -p 12 --dta -x gene\_ind -1 S3a\_1.fastq.gz -2 S3a\_2.fastq.gz -S S3.sam

hisat2 -p 12 --dta -x gene\_ind -1 S4a\_1.fastq.gz -2 S4a\_2.fastq.gz -S S4.sam

hisat2 -p 12 --dta -x gene\_ind -1 S5a\_1.fastq.gz -2 S5a\_2.fastq.gz -S S5.sam

## Samtools: sam to bam conversion and sorting

# Run as two processes due to samtools version

samtools view -bS -o R14.bam R14.sam

samtools sort R14.bam R14.sort

samtools view -bS -o R4.bam R4.sam

samtools sort R4.bam R4.sort

samtools view -bS -o R6.bam R6.sam

samtools sort R6.bam R6.sort

samtools view -bS -o R8.bam R8.sam

samtools sort R8.bam R8.sort

samtools view -bS -o S5.bam S5.sam

samtools sort S5.bam S5.sort

samtools view -bS -o S4.bam S4.sam

samtools sort S4.bam S4.sort

samtools view -bS -o S3.bam S3.sam

samtools sort S3.bam S3.sort

samtools view -bS -o S2.bam S2.sam

samtools sort S2.bam S2.sort

## StringTie

# ST assembly

stringtie R14.sort.bam -p 12 -G gene.gff -l R14 -o R14.gtf

stringtie R4.sort.bam -p 12 -G gene.gff -l R4 -o R4.gtf

stringtie R6.sort.bam -p 12 -G gene.gff -l R6 -o R6.gtf

stringtie R8.sort.bam -p 12 -G gene.gff -l R8 -o R8.gtf

stringtie S2.sort.bam -p 12 -G gene.gff -l S2 -o S2.gtf

stringtie S3.sort.bam -p 12 -G gene.gff -l S3 -o S3.gtf

stringtie S4.sort.bam -p 12 -G gene.gff -l S4 -o S4.gtf

stringtie S5.sort.bam -p 12 -G gene.gff -l S5 -o S5.gtf

# ST merge

stringtie --merge -p 12 -G gene.gff -o stringtie\_merged.gtf mergelist.txt

# ST -Be for ballgown

stringtie R14.sort.bam -p 12 -G stringtie\_merged.gtf -B -e -o bg\_R14/R14\_bg.gtf

stringtie R4.sort.bam -p 12 -G stringtie\_merged.gtf -B -e -o bg\_R4/R4\_bg.gtf

stringtie R6.sort.bam -p 12 -G stringtie\_merged.gtf -B -e -o bg\_R6/R6\_bg.gtf

stringtie R8.sort.bam -p 12 -G stringtie\_merged.gtf -B -e -o bg\_R8/R8\_bg.gtf

stringtie S2.sort.bam -p 12 -G stringtie\_merged.gtf -B -e -o bg\_S2/S2\_bg.gtf

stringtie S3.sort.bam -p 12 -G stringtie\_merged.gtf -B -e -o bg\_S3/S3\_bg.gtf

stringtie S4.sort.bam -p 12 -G stringtie\_merged.gtf -B -e -o bg\_S4/S4\_bg.gtf

stringtie S5.sort.bam -p 12 -G stringtie\_merged.gtf -B -e -o bg\_S5/S5\_bg.gtf

## RStudio

# Loading packages for Ballgown

library(usethis)

library(ballgown)

library(RColorBrewer)

library(genefilter)

library(dplyr)

library(devtools)

# Importing covariate data .csv

library(readr)

Refrence <- read\_csv("Refrence.csv", col\_names = FALSE)

View(Refrence)

# sorting the csv by sample to agree with the directory ordering

reference = Refrence[order(Refrence$X1),]

# creating the ballgown object from structured stringtie output

BGoutput1 <- ballgown(dataDir = 'C:/Users/Naomi/Desktop/Ballgown', samplePattern = "bg\_", pData= data.frame(reference))

BGoutput1\_filt <- subset(BGoutput1,"rowVars(texpr(BGoutput1)) >1", genomesubset= TRUE)'

# creating new objects from the filtered ballgown object to examine transcripts and genes separately

results\_transcripts = stattest(BGoutput1\_filt, feature = "transcript", covariate = "X2", getFC = TRUE, meas = "FPKM" )

results\_genes = stattest(BGoutput1\_filt, feature = "gene", covariate = "X2", getFC = TRUE, meas = "FPKM" )

results\_transcripts = data.frame(geneNames = ballgown:: geneNames(BGoutput1\_filt), geneIDs= ballgown :: geneIDs(BGoutput1\_filt), results\_transcripts)

# sorting by p value

results\_transcripts = arrange(results\_transcripts, pval)

results\_genes = arrange(results\_genes, pval)

# normalizing FPKM

fpkm = texpr(BGoutput1, meas = "FPKM")

fpkm = log2(fpkm +1)

# generating the box plot to illustrate sample variance

boxplot(fpkm, col=as.numeric(reference$X2), las = 2, ylab='log2(FPKM+1)')

##Figures 3 (In R Markdown)

sampleNames(BGoutput1)

pData1 <- data.frame(id=sampleNames(BGoutput1), group=rep(c(Refrence$X2), each=4))'

tx\_table = texpr(BGoutput1, "all")

fpkm1 = texpr(BGoutput1, meas="FPKM")

transcript\_det = stattest(BGoutput1, feature="transcript", covariate= "Sample", getFC=TRUE, meas="FPKM")

gene\_deg = stattest(BGoutput1, feature="gene", covariate="Sample", getFC=TRUE, meas="FPKM")

bg1 = ballgown (dataDir="C:/Users/Naomi/Desktop/Ballgown", samplePattern="bg\_", meas='all')

bg1@dirs

transcript\_fpkm = texpr(bg1, 'FPKM')

transcript\_cov = texpr(bg1, 'cov')

whole\_tx\_table = texpr(bg1, 'all')

transcript\_gene\_table = indexes(bg1)$t2g

hist(whole\_tx\_table$length, breaks=50, xlab="Transcript length", main="Distribution of transcript lengths", col="steelblue")

##Figure 4 (In R Markdown)

counts=table(transcript\_gene\_table[,"g\_id"])

c\_one = length(which(counts == 1))

# genes with more than one transcript

c\_more\_than\_one = length(which(counts > 1))

# what is the maximum number of transcripts per gene

c\_max = max(counts)

# plot above data

hist(counts, breaks=50, col="bisque4", xlab="Transcripts per gene", main="Distribution of transcript count per gene")

# add legend

legend\_text = c(paste("Genes with one transcript =", c\_one), paste("Genes with more than one transcript =", c\_more\_than\_one), paste("Max transcripts for single gene = ", c\_max))

legend("topright", legend\_text)

## Volcano Plot

# Filtering threshold: 16x fold change

fc\_sig\_results\_genes=which(abs(results\_genes$logfc)>4)

fc\_sig\_results\_genes\_plot=results\_genes[fc\_sig\_results\_genes,]

plot(results\_genes$logfc,results\_genes$qval, col="blue", pch=1)

abline(v=c(2,-2), col="red", lwd=3)

abline(h=0.05, col="red",lwd=3)

# highlighting genes

points(fc\_sig\_results\_genes\_plot$logfc,fc\_sig\_results\_genes\_plot$qval, col="green", pch=16)

## Heatmap generation

# gene results sorted for genes with p-value < 0.05

sigpi = which(results\_genes[,"pval"]<0.05)

# significant genes (p<0.05) stored in sigp

sigp = results\_genes[sigpi,]

sigde = which(abs(sigp[,"logfc"]) >= 2)

# significant genes further sorted based on up/downregulation greater than 4 fold

sig\_tn\_de = sigp[sigde,]

sig\_gene\_expression=gene\_expression[rownames(gene\_expression) %in% sig\_tn\_de$id,]

# columns 9 and 10 correspond to resistant and susceptible summary: removed

sig\_gene\_expression=sig\_gene\_expression[,-c(9:10)]

rownames(pheno\_data)=pheno\_data[,1]

phenotype\_table=subset(pheno\_data, select = -c(group) )

library(pheatmap)

# creating color gradient

m <- matrix(c(rnorm(1000)), ncol=100)

distmat <- dist(t(m))

makeColorRampPalette <- function(colors, cutoff.fraction, num.colors.in.palette)

{ stopifnot(length(colors) == 4)

ramp1 <- colorRampPalette(colors[1:2])(num.colors.in.palette \* cutoff.fraction)

ramp2 <- colorRampPalette(colors[3:4])(num.colors.in.palette \* (1 - cutoff.fraction))

return(c(ramp1, ramp2))}

cutoff.distance <- 4

cols <- makeColorRampPalette(c("blue", "green","yellow", "red"), cutoff.distance / max(distmat),100)

pheatmap(as.matrix(sig\_gene\_expression), scale = "row", clustering\_distance\_rows = "correlation", clustering\_method = "complete", main="Expression Heatmap: Significant Genes", fontsize\_col=8, fontsize\_row = 8, color = c("blue","yellow"))

--------

Mergelist.txt contents:

R14.gtf

R8.gtf

R6.gtf

R4.gtf

S2.gtf

S3.gtf

S4.gtf

S5.gtf

---

Phenotype .csv contents:

|  |  |
| --- | --- |
| **X1** | **X2** |
| bg\_R14 | resistant |
| bg\_R4 | resistant |
| bg\_R6 | resistant |
| bg\_R8 | resistant |
| bg\_S2 | susceptible |
| bg\_S3 | susceptible |
| bg\_S4 | susceptible |
| bg\_S5 | susceptible |