compbio_project

John Henry Cruz

10/14/2019

John Henry Cruz jec4968

Timeseries Pipeline on House Finch Data

SDS 358 *Dr. Woodward*

Times Series Pipeline Manuscript

Joyce Wang and John Henry Cruz

Introduction:

In nature, differential gene expression occurs, activating different genes within a cell. Sometimes, this leads to phenotypic changes in an organism. In different phenomena, these phenotypic changes can happen instantaneously or over a certain time period. But rarely do people study the trajectory of gene expression. Time series data are sets of gene expression data collected at different time points in chronological order, which is useful for discovering changes in gene expression as time progresses. For instance, collecting data at each time point of embryogenesis, the development of embryos, helps us identify the genes activated at different stages of this process. However, analyzing gene expression between two distinct stages of embryogenesis may easily disregard the subtle changes during each process within a stage. In this scenario, collecting and analyzing time series data will be more thorough due to its ability to discover changes in gene expression between each time point. Given the benefits of analyzing time series data, we created a time series pipeline to identify the changes in gene expression between different time points.

Using this pipeline, we identified key genes that contribute to the phenotypic changes of a subordinate, male Astatotilapia burtoni ascending into a dominant one (Burmeister et al 2005). for A. burtoni, social ascent from subordinate to dominant social status is a process of 1-5 days, meaning differential gene expression happened over the course of a few days and can not just be examined from two distinct time points (Maruska and Fernald 2010).

Different selective pressures that exist within the growth environments of house finches have shown to vary the end conditions after embryonic development. Not only were their survival rates affected, but the development timespan of females and the physiological characteristics of the males were affected. With the use of this pipeline, we aim to find a set of genes that may explain why male house finches that were exposed to mites not only grew larger and hatched faster compared to their non exposed counterparts.

Materials and Methods:

In the Carpodacus mexicanus experiment, the house finches have nests that are infested with Parapielus reedi, a nest mite, during the embryonic developmental stages. The 6 samples are split up into finches that were (3 samples) and were not (3 samples) in the presence of the nest mites. Of the 3 samples per the nest mite treatment are 3 samples from different time periods during the finch development. The 3 time periods are: Days 6-8, Day 8, Day 9. In each of these samples comes an abundance data set. This dataset contains transcript IDs with corresponding length, effect length, counts, and TPM values.

To analyze the gene expression of both datasets, we used an R environment. Figure 1 shows our workflow we during the time series data analysis. For samples that started off with raw data, we first collected the gene IDs. Gene id's were collected by gathering the transcript id's and translating that list into their corresponding gene id's. Once gene IDs were added to the data frame containing the raw data, TPM normalization was used to normalize the gene expression data. For TPM's that were under 2, those values were set to 0 to account for the effects of individual variations. We then collected the genes that are expressed in at least 90% of the samples. Once these genes are collected, a new filtered data frame was created with the previously collected gene IDs and their raw counts. The filtered data frame was then undergone either TMM (edgeR package) or RLE normalization (DESeq2 package). After normalization, PCAs were created to visually represent the data, showing potential outliers. With the normalized data, DESeq2, ctsGE, and WGCNA analysis was performed in order to find genes that are differentially expressed. Effect sizes were then calculated to further evaluate the significance of the differentially expressed target genes. Once these target genes were identified, GO analysis was conducted to search for possible pathways, which may explain the phenomena.

```
knitr::opts_chunk$set(echo = TRUE)
library(knitr)
library(ggplot2)
install.packages("tidyr")
```

```
## Installing package into '/stor/home/jhec819/R/x86_64-pc-linux-gnu-library/3.4'
## (as 'lib' is unspecified)
```

```
## Warning in install.packages("tidyr"): installation of package 'tidyr' had
## non-zero exit status
```

```
library(tidyr)
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.2.1 ──
```

```
## ✔ tibble  2.1.3     ✔ dplyr   0.8.3
## ✔ readr   1.3.1     ✔ stringr 1.4.0
## ✔ purrr   0.3.2     ✔ forcats 0.4.0
```

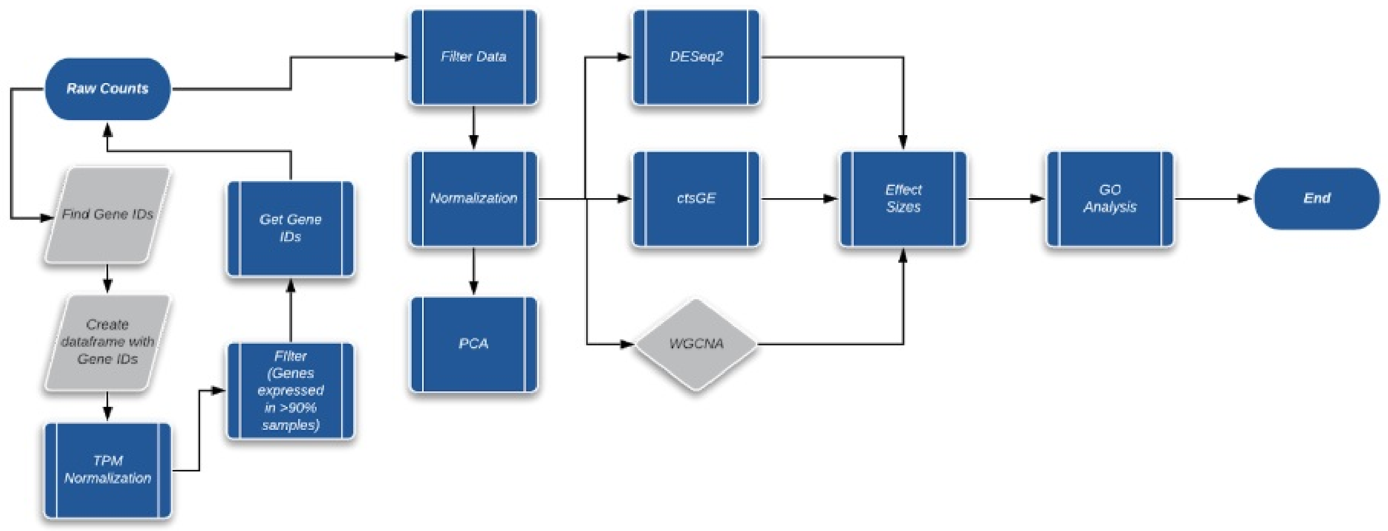
```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
library(readr)
install.packages("dplyr")
```

```
## Installing package into '/stor/home/jhec819/R/x86_64-pc-linux-gnu-library/3.4'
## (as 'lib' is unspecified)
```

```
## Warning in install.packages("dplyr"): installation of package 'dplyr' had
## non-zero exit status
```

```
library(dplyr)
```



Workflow Diagram

Part 1: Grab Gene ID's and Create Dataframe with Gene ID's and TPM's

Pre_6_7

```
abundance_pre_6_7 <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_Time
Series/John_Henry_Cruz/Zebrafinch/D6-7Pre_EVA68.fastq/abundance_pre_6_7.csv", header = T, sep =
"\t")
#load in the dataframe

pre_6_7_ti_list <- abundance_pre_6_7$target_id[1:18610]
## gather all the transcript id's and putting them into a list
pre_6_7_ti_list[1435]
```

```
## [1] ENSTGUT00000000664.1
## 18610 Levels: ENSTGUT00000000001.1 ... ENSTGUT00000019483.1
```

```
#test to see if the line above actually worked
write.table(pre_6_7_ti_list, file = "pre_6_7_ti_list", sep = "\t",
            row.names = FALSE)
#put list into txt file
```

sed -i ‘s/.1//g’ pre_6_7_ti_list.txt

remove “0.1” from the .txt file

sed -i ‘s/.2//g’ pre_6_7_ti_list.txt

remove “0.2” from the .txt file

sed -i ’s/“//g’ pre_6_7_ti_list.txt ######remove” " " from the .txt file

Gather Gene ID’s from Online using Ensembl

Output is a .txt file

```
pre_6_7_total_list <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_Tim
eSeries/John_Henry_Cruz/Zebrafinch/D6-7Pre_EVA68.fastq/pre_6_7_total_list.csv")
#load in the dataframe which is the product of Ensembl

pre_6_7_giti_df <- pre_6_7_total_list[, c("Transcript.stable.ID.version", "Gene.stable.ID")]
#pull two columns into a new dataframe

pre_6_7_giti_total <- left_join(abundance_pre_6_7, pre_6_7_giti_df, by = c("target_id"="Transcri
pt.stable.ID.version"))

length(unique(pre_6_7_giti_total$Gene.stable.ID))
```

```
## [1] 17894
```

```
#see if there are repeating gene id's
#we see that there are repeating gene id's

head(pre_6_7_giti_total %>% filter(duplicated(pre_6_7_giti_total$Gene.stable.ID)))
```

```
##              target_id length eff_length  est_counts         tpm
## 1 ENSTGUT00000017949.1    369        190 0.00000e+00 0.00000e+00
## 2 ENSTGUT00000017847.1    804        625 0.00000e+00 0.00000e+00
## 3 ENSTGUT00000017827.1    762        583 0.00000e+00 0.00000e+00
## 4 ENSTGUT00000017808.1    993        814 2.92971e-06 1.35037e-05
## 5 ENSTGUT00000017757.1    480        301 0.00000e+00 0.00000e+00
## 6 ENSTGUT00000017760.1    408        229 0.00000e+00 0.00000e+00
##       Gene.stable.ID
## 1 ENSTGUG00000017270
## 2 ENSTGUG00000017167
## 3 ENSTGUG00000017153
## 4 ENSTGUG00000017136
## 5 ENSTGUG00000017088
## 6 ENSTGUG00000017087
```

```
#find out which gene id's are repeated

pre_6_7_w_means <- pre_6_7_giti_total %>% group_by(Gene.stable.ID) %>% summarise(mean_lenght = m
ean(length), mean_eff_lenght = mean(eff_length), mean_est_counts = mean(est_counts), mean_tpm =
 mean(tpm))
#make new column name

pre_6_7_w_means
```

```
## # A tibble: 17,894 x 5
##    Gene.stable.ID     mean_lenght mean_eff_lenght mean_est_counts mean_tpm
##    <fct>                    <dbl>           <dbl>           <dbl>    <dbl>
##  1 ENSTGUG00000000001        1434            1255               6     17.9
##  2 ENSTGUG00000000002        1525            1346              24     66.9
##  3 ENSTGUG00000000003        2630            2451               8     12.2
##  4 ENSTGUG00000000004         678             499               0      0
##  5 ENSTGUG00000000005         504             325               0      0
##  6 ENSTGUG00000000006        1863            1684               9     20.1
##  7 ENSTGUG00000000007        5148            4969              16     12.1
##  8 ENSTGUG00000000008         363             184               0      0
##  9 ENSTGUG00000000010        1318            1139              51    168.
## 10 ENSTGUG00000000011         504             325               1     11.5
## # … with 17,884 more rows
```

```
#view dataframe
```

Post_6_7

```
abundance_post_6_7 <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_Tim
eSeries/John_Henry_Cruz/Zebrafinch/EVA-69_D6-7Post.fastq/abundance_post_6_7.csv", header = T, se
p = "\t")
#load in the dataframe

post_6_7_ti_list <- abundance_post_6_7$target_id[1:18610]
## gather all the transcript id's and putting them into a list
post_6_7_ti_list[1435]
```

```
## [1] ENSTGUT00000000664.1
## 18610 Levels: ENSTGUT00000000001.1 ... ENSTGUT00000019483.1
```

```
#test to see if the line above actually worked
write.table(post_6_7_ti_list, file = "post_6_7_ti_list.txt", sep = "\t",
            row.names = FALSE)
#put list into txt file
```

sed -i ‘s/.1//g’ post_6_7_ti_list.txt

remove “0.1” from the .txt file

sed -i ‘s/.2//g’ post_6_7_ti_list.txt

remove “0.2” from the .txt file

sed -i ’s/“//g’ post_6_7_ti_list.txt ###### remove” " " from the .txt file

Gather Gene ID’s from Online using Ensembl

Output is a .txt file

```
post_6_7_total_list <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_Ti
meSeries/John_Henry_Cruz/Zebrafinch/EVA-69_D6-7Post.fastq/post_6_7_total_list.csv")
#load in the dataframe which is the product of Ensembl

post_6_7_giti_df <- post_6_7_total_list[, c("Transcript.stable.ID.version", "Gene.stable.ID")]
#pull two columns into a new dataframe

post_6_7_giti_total <- left_join(abundance_post_6_7, post_6_7_giti_df, by = c("target_id"="Trans
cript.stable.ID.version"))

length(unique(post_6_7_giti_total$Gene.stable.ID))
```

```
## [1] 17894
```

```
#see if there are repeating gene id's
#we see that there are repeating gene id's

head(post_6_7_giti_total %>% filter(duplicated(post_6_7_giti_total$Gene.stable.ID)))
```

```
##              target_id length eff_length est_counts    tpm
## 1 ENSTGUT00000017949.1    369        190          1 26.084
## 2 ENSTGUT00000017847.1    804        625          0  0.000
## 3 ENSTGUT00000017827.1    762        583          0  0.000
## 4 ENSTGUT00000017808.1    993        814          0  0.000
## 5 ENSTGUT00000017757.1    480        301          0  0.000
## 6 ENSTGUT00000017760.1    408        229          0  0.000
##       Gene.stable.ID
## 1 ENSTGUG00000017270
## 2 ENSTGUG00000017167
## 3 ENSTGUG00000017153
## 4 ENSTGUG00000017136
## 5 ENSTGUG00000017088
## 6 ENSTGUG00000017087
```

```
#find out which gene id's are repeated

post_6_7_w_means <- post_6_7_giti_total %>% group_by(Gene.stable.ID) %>% summarise(mean_lenght =
mean(length), mean_eff_lenght = mean(eff_length), mean_est_counts = mean(est_counts), mean_tpm =
mean(tpm))
#make new column name

post_6_7_w_means
```

```
## # A tibble: 17,894 x 5
##    Gene.stable.ID     mean_lenght mean_eff_lenght mean_est_counts mean_tpm
##    <fct>                    <dbl>           <dbl>           <dbl>    <dbl>
##  1 ENSTGUG00000000001        1434            1255              12     47.4
##  2 ENSTGUG00000000002        1525            1346              23     84.7
##  3 ENSTGUG00000000003        2630            2451              21     42.5
##  4 ENSTGUG00000000004         678             499               0      0
##  5 ENSTGUG00000000005         504             325               0      0
##  6 ENSTGUG00000000006        1863            1684              19     55.9
##  7 ENSTGUG00000000007        5148            4969              34     33.9
##  8 ENSTGUG00000000008         363             184               0      0
##  9 ENSTGUG00000000010        1318            1139              47    205.
## 10 ENSTGUG00000000011         504             325               2     30.5
## # … with 17,884 more rows
```

```
#view dataframe
```

Post_8

```
abundance_post_8 <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_TimeS
eries/John_Henry_Cruz/Zebrafinch/Post8_08PSU12E4_EVA72.fastq/abundance_post_8.csv", header = T,
 sep = "\t")
#load in the dataframe

post_8_ti_list <- abundance_post_8$target_id[1:18610]
## gather all the transcript id's and putting them into a list
post_8_ti_list[1435]
```

```
## [1] ENSTGUT00000000664.1
## 18610 Levels: ENSTGUT00000000001.1 ... ENSTGUT00000019483.1
```

```
#test to see if the line above actually worked
write.table(post_8_ti_list, file = "post_8_ti_list.txt", sep = "\t",
            row.names = FALSE)
#put list into txt file
```

sed -i ‘s/.1//g’ post_8_ti_list.txt

remove “0.1” from the .txt file

sed -i ‘s/.2//g’ post_8_ti_list.txt

remove “0.2” from the .txt file

sed -i ’s/“//g’ post_8_ti_list.txt ###### remove” " " from the .txt file

Gather Gene ID’s from Online using Ensembl

Output is a .txt file

```
post_8_total_list <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_Time
Series/John_Henry_Cruz/Zebrafinch/Post8_08PSU12E4_EVA72.fastq//post_8_total_list.csv")
#load in the dataframe which is the product of Ensembl

post_8_giti_df <- post_8_total_list[, c("Transcript.stable.ID.version", "Gene.stable.ID")]
#pull two columns into a new dataframe

post_8_giti_total <- left_join(abundance_post_8, post_8_giti_df, by = c("target_id"="Transcript.
stable.ID.version"))

length(unique(post_8_giti_total$Gene.stable.ID))
```

```
## [1] 17894
```

```
#see if there are repeating gene id's
#we see that there are repeating gene id's

head(post_8_giti_total %>% filter(duplicated(post_8_total_list$Gene.stable.ID)))
```

```
##              target_id length eff_length est_counts       tpm
## 1 ENSTGUT00000017952.1   1632  1453.0000    0.00000   0.00000
## 2 ENSTGUT00000017841.1   1706  1527.0000    3.00000   6.88357
## 3 ENSTGUT00000017828.1   2658  2479.0000    3.00000   4.24010
## 4 ENSTGUT00000017801.1    903   724.0000   24.00000 116.14600
## 5 ENSTGUT00000017745.1    189    20.4424    0.00000   0.00000
## 6 ENSTGUT00000017751.1    477   298.0000    3.96057  46.56640
##       Gene.stable.ID
## 1 ENSTGUG00000017269
## 2 ENSTGUG00000017160
## 3 ENSTGUG00000017145
## 4 ENSTGUG00000017126
## 5 ENSTGUG00000017077
## 6 ENSTGUG00000017075
```

```
#find out which gene id's are repeated

post_8_w_means <- post_8_giti_total %>% group_by(Gene.stable.ID) %>% summarise(mean_lenght = mea
n(length), mean_eff_lenght = mean(eff_length), mean_est_counts = mean(est_counts), mean_tpm = me
an(tpm))
#make new column name

post_8_w_means
```

```
## # A tibble: 17,894 x 5
##    Gene.stable.ID     mean_lenght mean_eff_lenght mean_est_counts mean_tpm
##    <fct>                    <dbl>           <dbl>           <dbl>    <dbl>
##  1 ENSTGUG00000000001        1434            1255               9     25.1
##  2 ENSTGUG00000000002        1525            1346              29     75.5
##  3 ENSTGUG00000000003        2630            2451              24     34.3
##  4 ENSTGUG00000000004         678             499               0      0
##  5 ENSTGUG00000000005         504             325               0      0
##  6 ENSTGUG00000000006        1863            1684              12     25.0
##  7 ENSTGUG00000000007        5148            4969              27     19.0
##  8 ENSTGUG00000000008         363             184               0      0
##  9 ENSTGUG00000000010        1318            1139              47    145.
## 10 ENSTGUG00000000011         504             325               0      0
## # … with 17,884 more rows
```

```
#view dataframe
```

Post_9

```
abundance_post_9 <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_TimeS
eries/John_Henry_Cruz/Zebrafinch/Post9_08PSU12E2_EVA70.fastq/abundance_post_9.csv", header = T,
 sep = "\t")
#load in the dataframe

post_9_ti_list <- abundance_post_9$target_id[1:18610]
## gather all the transcript id's and putting them into a list
post_9_ti_list[1435]
```

```
## [1] ENSTGUT00000000664.1
## 18610 Levels: ENSTGUT00000000001.1 ... ENSTGUT00000019483.1
```

```
#test to see if the line above actually worked
write.table(post_9_ti_list, file = "post_9_ti_list.txt", sep = "\t",
            row.names = FALSE)
#put list into txt file
```

sed -i ‘s/.1//g’ post_9_ti_list.txt

remove “0.1” from the .txt file

sed -i ‘s/.2//g’ post_9_ti_list.txt

remove “0.2” from the .txt file

sed -i ’s/“//g’ post_9_ti_list.txt ###### remove” " " from the .txt file

Gather Gene ID’s from Online using Ensembl

Output is a .txt file

```
post_9_total_list <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_Time
Series/John_Henry_Cruz/Zebrafinch/Post9_08PSU12E2_EVA70.fastq/post_9_total_list.csv")
#load in the dataframe which is the product of Ensembl

post_9_giti_df <- post_9_total_list[, c("Transcript.stable.ID.version", "Gene.stable.ID")]
#pull two columns into a new dataframe

post_9_giti_total <- left_join(abundance_post_9, post_9_giti_df, by = c("target_id"="Transcript.
stable.ID.version"))

length(unique(post_9_giti_total$Gene.stable.ID))
```

```
## [1] 17894
```

```
#see if there are repeating gene id's
#we see that there are repeating gene id's

head(post_9_giti_total %>% filter(duplicated(post_9_total_list$Gene.stable.ID)))
```

```
##              target_id length eff_length est_counts       tpm
## 1 ENSTGUT00000017952.1   1632  1453.0000    0.00000   0.00000
## 2 ENSTGUT00000017841.1   1706  1527.0000    2.00000   6.57853
## 3 ENSTGUT00000017828.1   2658  2479.0000    2.00000   4.05221
## 4 ENSTGUT00000017801.1    903   724.0000    6.00000  41.62470
## 5 ENSTGUT00000017745.1    189    20.4424    0.00000   0.00000
## 6 ENSTGUT00000017751.1    477   298.0000    6.96057 117.31800
##       Gene.stable.ID
## 1 ENSTGUG00000017269
## 2 ENSTGUG00000017160
## 3 ENSTGUG00000017145
## 4 ENSTGUG00000017126
## 5 ENSTGUG00000017077
## 6 ENSTGUG00000017075
```

```
#find out which gene id's are repeated

post_9_w_means <- post_9_giti_total %>% group_by(Gene.stable.ID) %>% summarise(mean_lenght = mea
n(length), mean_eff_lenght = mean(eff_length), mean_est_counts = mean(est_counts), mean_tpm = me
an(tpm))
#make new column name

post_9_w_means
```

```
## # A tibble: 17,894 x 5
##    Gene.stable.ID     mean_lenght mean_eff_lenght mean_est_counts mean_tpm
##    <fct>                    <dbl>           <dbl>           <dbl>    <dbl>
##  1 ENSTGUG00000000001        1434            1255               8     32.0
##  2 ENSTGUG00000000002        1525            1346              13     48.5
##  3 ENSTGUG00000000003        2630            2451              10     20.5
##  4 ENSTGUG00000000004         678             499               0      0
##  5 ENSTGUG00000000005         504             325               0      0
##  6 ENSTGUG00000000006        1863            1684              11     32.8
##  7 ENSTGUG00000000007        5148            4969              14     14.2
##  8 ENSTGUG00000000008         363             184               2     54.6
##  9 ENSTGUG00000000010        1318            1139              34    150.
## 10 ENSTGUG00000000011         504             325               0      0
## # … with 17,884 more rows
```

```
#view dataframe
```

Pre_8

```
abundance_pre_8 <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_TimeSe
ries/John_Henry_Cruz/Zebrafinch/Pre8_08CBS6E3_Eva79.fastq/abundance_pre_8.csv", header = T, sep
 = "\t")
#load in the dataframe

pre_8_ti_list <- abundance_pre_8$target_id[1:18610]
## gather all the transcript id's and putting them into a list
pre_8_ti_list[1435]
```

```
## [1] ENSTGUT00000000664.1
## 18610 Levels: ENSTGUT00000000001.1 ... ENSTGUT00000019483.1
```

```
#test to see if the line above actually worked
write.table(pre_8_ti_list, file = "pre_8_ti_list.txt", sep = "\t",
            row.names = FALSE)
#put list into txt file
```

sed -i ‘s/.1//g’ pre_8_ti_list.txt

remove “0.1” from the .txt file

sed -i ‘s/.2//g’ pre_8_ti_list.txt

remove “0.2” from the .txt file

sed -i ’s/“//g’ pre_8_ti_list.txt ###### remove” " " from the .txt file

Gather Gene ID’s from Online using Ensembl

Output is a .txt file

```
pre_8_total_list <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_TimeS
eries/John_Henry_Cruz/Zebrafinch/Pre8_08CBS6E3_Eva79.fastq//pre_8_total_list.csv")
#load in the dataframe which is the product of Ensembl

pre_8_giti_df <- pre_8_total_list[, c("Transcript.stable.ID.version", "Gene.stable.ID")]
#pull two columns into a new dataframe

pre_8_giti_total <- left_join(abundance_pre_8, pre_8_giti_df, by = c("target_id"="Transcript.sta
ble.ID.version"))

length(unique(pre_8_giti_total$Gene.stable.ID))
```

```
## [1] 17894
```

```
#see if there are repeating gene id's
#we see that there are repeating gene id's

head(pre_8_giti_total %>% filter(duplicated(pre_8_total_list$Gene.stable.ID)))
```

```
##              target_id length eff_length est_counts       tpm
## 1 ENSTGUT00000017952.1   1632  1453.0000    1.00000   3.12138
## 2 ENSTGUT00000017841.1   1706  1527.0000    3.00000   8.91036
## 3 ENSTGUT00000017828.1   2658  2479.0000    0.00000   0.00000
## 4 ENSTGUT00000017801.1    903   724.0000   16.00000 100.22900
## 5 ENSTGUT00000017745.1    189    20.4424    0.00000   0.00000
## 6 ENSTGUT00000017751.1    477   298.0000    4.74216  72.17270
##         Gene.stable.ID
## 1 ENSTGUG00000017269
## 2 ENSTGUG00000017160
## 3 ENSTGUG00000017145
## 4 ENSTGUG00000017126
## 5 ENSTGUG00000017077
## 6 ENSTGUG00000017075
```

```
#find out which gene id's are repeated

pre_8_w_means <- pre_8_giti_total %>% group_by(Gene.stable.ID) %>% summarise(mean_lenght = mean
(length), mean_eff_lenght = mean(eff_length), mean_est_counts = mean(est_counts), mean_tpm = mea
n(tpm))
#make new column name

pre_8_w_means
```

```
## # A tibble: 17,894 x 5
##    Gene.stable.ID     mean_lenght mean_eff_lenght mean_est_counts mean_tpm
##    <fct>                    <dbl>           <dbl>           <dbl>    <dbl>
##  1 ENSTGUG00000000001        1434            1255               9    32.5
##  2 ENSTGUG00000000002        1525            1346              17    57.3
##  3 ENSTGUG00000000003        2630            2451               7    13.0
##  4 ENSTGUG00000000004         678             499               1     9.09
##  5 ENSTGUG00000000005         504             325               0     0
##  6 ENSTGUG00000000006        1863            1684              12    32.3
##  7 ENSTGUG00000000007        5148            4969              24    21.9
##  8 ENSTGUG00000000008         363             184               0     0
##  9 ENSTGUG00000000010        1318            1139              34   135.
## 10 ENSTGUG00000000011         504             325               3    41.9
## # … with 17,884 more rows
```

```
#view dataframe
```

Pre_9

```
abundance_pre_9 <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_TimeSe
ries/John_Henry_Cruz/Zebrafinch/Pre9_08CBS6E2_EVA71.fastq/abundance_pre_9.csv", header = T, sep
 = "\t")
#load in the dataframe

pre_9_ti_list <- abundance_pre_9$target_id[1:18610]
## gather all the transcript id's and putting them into a list
pre_9_ti_list[1435]
```

```
## [1] ENSTGUT00000000664.1
## 18610 Levels: ENSTGUT00000000001.1 ... ENSTGUT00000019483.1
```

```
#test to see if the line above actually worked
write.table(pre_9_ti_list, file = "pre_9_ti_list.txt", sep = "\t",
            row.names = FALSE)
#put list into txt file
```

sed -i ‘s/.1//g’ pre_9_ti_list.txt

remove “0.1” from the .txt file

sed -i ‘s/.2//g’ pre_9_ti_list.txt

remove “0.2” from the .txt file

sed -i ’s/“//g’ pre_9_ti_list.txt ###### remove” " " from the .txt file

Gather Gene ID’s from Online using Ensembl

Output is a .txt file

```
pre_9_total_list <- read.csv("/stor/work/Hofmann/Shared/Undergraduate_Students/FRI_BigData_TimeS
eries/John_Henry_Cruz/Zebrafinch/Pre9_08CBS6E2_EVA71.fastq/pre_9_total_list.csv")
#load in the dataframe which is the product of Ensembl

pre_9_giti_df <- pre_9_total_list[, c("Transcript.stable.ID.version", "Gene.stable.ID")]
#pull two columns into a new dataframe

pre_9_giti_total <- left_join(abundance_pre_9, pre_9_giti_df, by = c("target_id"="Transcript.sta
ble.ID.version"))

length(unique(pre_9_giti_total$Gene.stable.ID))
```

```
## [1] 17894
```

```
#see if there are repeating gene id's
#we see that there are repeating gene id's

head(pre_9_giti_total %>% filter(duplicated(pre_9_total_list$Gene.stable.ID)))
```

```
##              target_id length eff_length est_counts       tpm
## 1 ENSTGUT00000017952.1   1632  1453.0000     0.0000   0.00000
## 2 ENSTGUT00000017841.1   1706  1527.0000     4.0000   9.06992
## 3 ENSTGUT00000017828.1   2658  2479.0000     1.0000   1.39671
## 4 ENSTGUT00000017801.1    903   724.0000    20.0000  95.64760
## 5 ENSTGUT00000017745.1    189    20.4424     0.0000   0.00000
## 6 ENSTGUT00000017751.1    477   298.0000    11.8514 137.70000
##       Gene.stable.ID
## 1 ENSTGUG00000017269
## 2 ENSTGUG00000017160
## 3 ENSTGUG00000017145
## 4 ENSTGUG00000017126
## 5 ENSTGUG00000017077
## 6 ENSTGUG00000017075
```

```
#find out which gene id's are repeated

pre_9_w_means <- pre_9_giti_total %>% group_by(Gene.stable.ID) %>% summarise(mean_lenght = mean
(length), mean_eff_lenght = mean(eff_length), mean_est_counts = mean(est_counts), mean_tpm = mea
n(tpm))
#make new column name

pre_9_w_means
```

```
## # A tibble: 17,894 x 5
##    Gene.stable.ID     mean_lenght mean_eff_lenght mean_est_counts mean_tpm
##    <fct>                    <dbl>           <dbl>           <dbl>    <dbl>
##  1 ENSTGUG00000000001        1434            1255              19    52.4
##  2 ENSTGUG00000000002        1525            1346              32    82.3
##  3 ENSTGUG00000000003        2630            2451              21    29.7
##  4 ENSTGUG00000000004         678             499               1     6.94
##  5 ENSTGUG00000000005         504             325               0     0
##  6 ENSTGUG00000000006        1863            1684               8    16.4
##  7 ENSTGUG00000000007        5148            4969              31    21.6
##  8 ENSTGUG00000000008         363             184               0     0
##  9 ENSTGUG00000000010        1318            1139              44   134.
## 10 ENSTGUG00000000011         504             325               5    53.3
## # … with 17,884 more rows
```

```
#view dataframe
```

Get the TPM Values that we will be manipulating

```
post_6_7_tpm <- post_6_7_w_means %>% select(Gene.stable.ID, mean_tpm) %>% rename("post_6_7"=mean
_tpm)
post_8_tpm <- post_8_w_means %>% select(Gene.stable.ID, mean_tpm) %>% rename("post_8"=mean_tpm)
post_9_tpm <- post_9_w_means %>% select(Gene.stable.ID, mean_tpm) %>% rename("post_9"=mean_tpm)
pre_6_7_tpm <- pre_6_7_w_means %>% select(Gene.stable.ID, mean_tpm) %>% rename("pre_6_7"=mean_tp
m)
pre_8_tpm <- pre_8_w_means %>% select(Gene.stable.ID, mean_tpm) %>% rename("pre_8"=mean_tpm)
pre_9_tpm <- pre_9_w_means %>% select(Gene.stable.ID, mean_tpm) %>% rename("pre_9"=mean_tpm)

#for each sample, take the gene id's and the tpm values and make a new dataframe
#rename the mean_tpm column to the sample name
#use gene id's as a way to join the datasets together
```

```
test <- left_join(post_6_7_tpm, post_8_tpm, by = "Gene.stable.ID")
test <- left_join(test, pre_6_7_tpm, by = "Gene.stable.ID")
test <- left_join(test, pre_8_tpm, by = "Gene.stable.ID")
test <- left_join(test, pre_9_tpm, by = "Gene.stable.ID")
test <- left_join(test, post_9_tpm, by = "Gene.stable.ID")
tpm_df <- test
#put all the sample's tpm values in one dataframe with the gene id's

sum(is.na(tpm_df$post_6_7)) + sum(is.na(tpm_df$post_8)) + sum(is.na(tpm_df$post_9)) + sum(is.na
(tpm_df$pre_6_7)) + sum(is.na(tpm_df$pre_8)) + sum(is.na(tpm_df$pre_9))
```

```
## [1] 0
```

```
#make sure that no tpm columns have na values to confirm joining was a success

head(tpm_df)
```

```
## # A tibble: 6 x 7
##   Gene.stable.ID     post_6_7 post_8 pre_6_7 pre_8 pre_9 post_9
##   <fct>                 <dbl>  <dbl>   <dbl> <dbl> <dbl>  <dbl>
## 1 ENSTGUG00000000001     47.4   25.1    17.9 32.5  52.4    32.0
## 2 ENSTGUG00000000002     84.7   75.5    66.9 57.3  82.3    48.5
## 3 ENSTGUG00000000003     42.5   34.3    12.2 13.0  29.7    20.5
## 4 ENSTGUG00000000004      0      0       0    9.09  6.94    0
## 5 ENSTGUG00000000005      0      0       0    0     0       0
## 6 ENSTGUG00000000006     55.9   25.0    20.1 32.3  16.4    32.8
```

Filter TPM values for low cutoffs and Sample Abundance

```
tpm_df_frequency <- tpm_df %>% mutate_at(vars(post_6_7:post_9), function(x) ifelse(x >= 2,1,0))
#make a dataframe for presence and absense where presence is 1 and absence is 0

tpm_df_w_cutoff <- tpm_df %>%
  mutate(post_6_7_tpm = case_when(post_6_7 < 2 ~ 0, TRUE ~ post_6_7)) %>%
  mutate(post_8_tpm = case_when(post_8 < 2 ~ 0, TRUE ~ post_8)) %>%
  mutate(post_9_tpm = case_when(post_9 < 2 ~ 0, TRUE ~ post_9)) %>%
  mutate(pre_6_7_tpm = case_when(pre_6_7 < 2 ~ 0, TRUE ~ pre_6_7)) %>%
  mutate(pre_8_tpm = case_when(pre_8 < 2 ~ 0, TRUE ~ pre_8)) %>%
  mutate(pre_9_tpm = case_when(pre_9 < 2 ~ 0, TRUE ~ pre_9)) %>% select(1,8:13)
#make a dataframe of the raw counts but filtering data that is less than 2 and changing those va
lues to 0

frequency_list <- rowSums(tpm_df_frequency[2:7])
#make a list of the sums of the elements in each row of the presence absense dataframe to see ho
w frequently we see that gene throughout the samples

frequency_df <- data.frame(frequency = matrix(unlist(frequency_list), nrow=17894, byrow=T),strin
gsAsFactors=FALSE)
#make a dataframe of the row sums of the frequency of the presence of the gene throughout the sa
mples

tpm_df_frequency_sums <- cbind.data.frame(tpm_df_frequency, frequency_df)
#make a dataframe with the frequency of the genes across all samples and the dataframe of presen
ce absence

tpm_df_frequency_sums_4_up <- tpm_df_frequency_sums[tpm_df_frequency_sums$frequency > 3,]
#make a new dataframe with only the genes that were seen at least 4 times

tpm_df_frequency_sums_5_up <- tpm_df_frequency_sums[tpm_df_frequency_sums$frequency > 4,]
#make a new dataframe with only the genes that were seen at least 5 times


frequency_4_up_list <- tpm_df_frequency_sums_4_up$Gene.stable.ID[1:9845]
#get list of the gene names that were seen at least 4 times

frequency_5_up_list <- tpm_df_frequency_sums_5_up$Gene.stable.ID[1:8782]
#get list of the gene names that were seen at least 5 times
```

Create Dataframes to be Used in DESeq2

```
tpm_df_filtered_4up <- left_join(tpm_df_frequency_sums_4_up, tpm_df_w_cutoff, by ="Gene.stable.I
D")
#make a new dataframe of only the raw counts with the genes that were seen at least 4 times, mak
ing the filtered dataframe

tpm_df_filtered_4up <- tpm_df_filtered_4up %>% select(1,9:14)
#take only the columns with the tpm values and the gene names

tpm_df_filtered_5up <- left_join(tpm_df_frequency_sums_5_up, tpm_df_w_cutoff, by ="Gene.stable.I
D")
#make a new dataframe of only the raw counts with the genes that were seen at least 4 times, mak
ing the filtered dataframe

tpm_df_filtered_5up <- tpm_df_filtered_5up %>% select(1, 9:14)
#take only the columns with the tpm values and the gene names
```

Create Dataframe To Do Wrangling Part of the Project

```
full_data <- tpm_df_w_cutoff %>% pivot_longer(c("post_6_7_tpm","pre_6_7_tpm","post_8_tpm","pre_8
_tpm","post_9_tpm","pre_9_tpm"), names_to = "sample", values_to = "tpm") %>% select(Gene.stable.
ID, sample, tpm)
#set up dataframe so that a column is "sample"" and its elements are the sample names

head(full_data)
```

```
## # A tibble: 6 x 3
##   Gene.stable.ID     sample         tpm
##   <fct>              <chr>        <dbl>
## 1 ENSTGUG00000000001 post_6_7_tpm  47.4
## 2 ENSTGUG00000000001 pre_6_7_tpm   17.9
## 3 ENSTGUG00000000001 post_8_tpm    25.1
## 4 ENSTGUG00000000001 pre_8_tpm     32.5
## 5 ENSTGUG00000000001 post_9_tpm    32.0
## 6 ENSTGUG00000000001 pre_9_tpm     52.4
```

```
#see what is looks like

pre_6_7_w_samp <- pre_6_7_w_means %>% mutate_at(vars(mean_tpm), function(x) ifelse(x > 0, 'pre_6
_7_tpm', 'pre_6_7_tpm')) %>% rename('sample'=mean_tpm)
#change the tpm values to say the sample that it's from

post_6_7_w_samp <- post_6_7_w_means %>% mutate_at(vars(mean_tpm), function(x) ifelse(x > 0, 'pos
t_6_7_tpm', 'post_6_7_tpm')) %>% rename('sample'=mean_tpm)
#change the tpm values to say the sample that it's from


pre_8_w_samp <- pre_8_w_means %>% mutate_at(vars(mean_tpm), function(x) ifelse(x > 0, 'pre_8_tp
m', 'pre_8_tpm')) %>% rename('sample'=mean_tpm)
#change the tpm values to say the sample that it's from


post_8_w_samp <- post_8_w_means %>% mutate_at(vars(mean_tpm), function(x) ifelse(x > 0, 'post_8_
tpm', 'post_8_tpm')) %>% rename('sample'=mean_tpm)
#change the tpm values to say the sample that it's from


pre_9_w_samp <- pre_9_w_means %>% mutate_at(vars(mean_tpm), function(x) ifelse(x > 0, 'pre_9_tp
m', 'pre_9_tpm')) %>% rename('sample'=mean_tpm)
#change the tpm values to say the sample that it's from


post_9_w_samp <- post_9_w_means %>% mutate_at(vars(mean_tpm), function(x) ifelse(x > 0, 'post_9_
tpm', 'post_9_tpm')) %>% rename('sample'=mean_tpm)
#change the tpm values to say the sample that it's from


sampledf <- rbind(post_6_7_w_samp, post_8_w_samp)
sampledf <- rbind(sampledf, post_9_w_samp)
sampledf <- rbind(sampledf, pre_6_7_w_samp)
sampledf <- rbind(sampledf, pre_8_w_samp)
sampledf <- rbind(sampledf, pre_9_w_samp)
#combine all of te dataframes vertically, so by rows
#join the dataframes by sample alphabetically

head(sampledf)
```

```
## # A tibble: 6 x 5
##   Gene.stable.ID     mean_lenght mean_eff_lenght mean_est_counts sample
##   <fct>                    <dbl>           <dbl>           <dbl> <chr>
## 1 ENSTGUG00000000001        1434            1255              12 post_6_7_…
## 2 ENSTGUG00000000002        1525            1346              23 post_6_7_…
## 3 ENSTGUG00000000003        2630            2451              21 post_6_7_…
## 4 ENSTGUG00000000004         678             499               0 post_6_7_…
## 5 ENSTGUG00000000005         504             325               0 post_6_7_…
## 6 ENSTGUG00000000006        1863            1684              19 post_6_7_…
```

```
full_data <- full_data %>% arrange(sample)
#have the dataframe go alphabetical order in the samples

compiled_df <- left_join(full_data,sampledf)
```

```
## Joining, by = c("Gene.stable.ID", "sample")
```

```
#combine the dataframe with the length values to the dataframe with the tpm values

compiled_df <- compiled_df %>% mutate(tpm_presence = tpm/1)
#make new column in the dataframe to change to the presence absence column

compiled_df <- compiled_df %>% mutate_at(vars(tpm_presence), function(x) ifelse(x > 0, 'yes', 'n
o'))
#change the values that are absent to false and the values that are present to true
```

Tidying and Joining

Much of the tidying that was done was working across samples to make sure that the values that I wanted were kept and pivoted into an oreintations where all samples and Gene ID's could have their tpm values in one row. After getting the gene id's from the transcript id's, next steps were to gather each samples TPM values so that I could filter the data and create a presence/absence dataframe. The gene id's collected after filtering were then crossed with the raw data to create the filtered dataframe. The dataframes then needed to be pivoted to have columns for the gene id, sample, and tpm. A left join was used to merge the abundance dataframe data(length and tpms) to the dataframe with the gene ids and sample names because I needed to keep the gene id's and tpm values. Since both dataframes were perectly aligned with gene id's and sample names, the left join created no NAs.

Wrangling Calculations

```
compiled_df_wranglin <- compiled_df %>% mutate(log_length = log10(mean_lenght))
#make new column which is the log of the lengths of the DNA sequences

compiled_df_wranglin %>% select(2,4,5) %>% filter(sample == 'post_6_7_tpm') %>% summarise(mean(m
ean_lenght))
```

```
## # A tibble: 1 x 1
##   `mean(mean_lenght)`
##                 <dbl>
## 1               1410.
```

```
#grab the sample, mean length, and mean effect length columns, filter for only post_6_7 samples
 and find the mean of the length of the DNA sequences

compiled_df_wranglin %>% select(2,4,5) %>% filter(sample=='post_9_tpm') %>% summarise(mean(mean_
lenght))
```

```
## # A tibble: 1 x 1
##   `mean(mean_lenght)`
##                 <dbl>
## 1               1410.
```

```
#grab the sample, mean length, and mean effect length columns, filter for only post_9 samples an
d find the mean of the length of the DNA sequences

compiled_df_wranglin %>% select(2,4,5) %>% filter(sample=='pre_8_tpm') %>% summarise(mean(mean_l
enght))
```

```
## # A tibble: 1 x 1
##   `mean(mean_lenght)`
##                 <dbl>
## 1               1410.
```

```
#grab the sample, mean length, and mean effect length columns, filter for only post_6_7 samples
 and find the mean of the length of the DNA sequences

compiled_df_wranglin %>% summarise(mean(mean_eff_lenght))
```

```
## # A tibble: 1 x 1
##   `mean(mean_eff_lenght)`
##                     <dbl>
## 1                   1232.
```

```
#find the mean effect length of all the samples

compiled_df_wranglin %>% summarise(sd(mean_lenght))
```

```
## # A tibble: 1 x 1
##   `sd(mean_lenght)`
##               <dbl>
## 1             1372.
```

```
#find the standard deviation of all the mean lengths of all the samples

compiled_df_wranglin %>% select(2,4,5) %>% filter(sample=='pre_6_7_tpm') %>% summarise(sd(mean_l
enght))
```

```
## # A tibble: 1 x 1
##   `sd(mean_lenght)`
##               <dbl>
## 1             1372.
```

```
#grab the sample, mean length, and mean effect length columns, filter for only pre_6_7 samples a
nd find the standard of the length of the DNA sequences

compiled_df_wranglin %>% group_by(tpm_presence) %>% filter(sample == "post_8_tpm") %>% summarise
(mean(tpm))
```

```
## # A tibble: 2 x 2
##   tpm_presence `mean(tpm)`
##   <chr>              <dbl>
## 1 no                   0
## 2 yes                 93.4
```

```
#group be having tpms expressed or not, filter by only the post_8 sample and find the mean tpm's
in that sample

compiled_df_wranglin %>% group_by(tpm_presence) %>% filter(sample == "pre_6_7_tpm") %>% summaris
e(mean(tpm>10))
```

```
## # A tibble: 2 x 2
##   tpm_presence `mean(tpm > 10)`
##   <chr>                   <dbl>
## 1 no                      0
## 2 yes                     0.712
```

```
#group be having tpms expressed or not, filter by only the pre_6_7 sample and find the mean tp
m's in that sample that were greater than 10

compiled_df_wranglin %>% group_by(tpm_presence) %>% filter(sample == "post_8_tpm") %>% arrange(d
esc(mean_eff_lenght)) %>% summarise(mean(tpm))
```

```
## # A tibble: 2 x 2
##   tpm_presence `mean(tpm)`
##   <chr>              <dbl>
## 1 no                   0
## 2 yes                 93.4
```

```
#group be having tpms expressed or not, filter by only the post_8 sample and find the mean tpm's
in that sample aftering arranging the mean effect length in decreasing numerical order

compiled_df_wranglin %>% mutate(tpm_pctile = ntile(tpm,100))%>%  group_by(tpm_presence) %>% filt
er(sample == 'post_9_tpm') %>% select(8,9) %>% arrange(tpm_pctile) %>% summarise(mean(tpm_pctil
e))
```

```
## Adding missing grouping variables: `tpm_presence`
## # A tibble: 2 x 2
##   tpm_presence `mean(tpm_pctile)`
##   <chr>                     <dbl>
## 1 no                         18.0
## 2 yes                        71.7
```

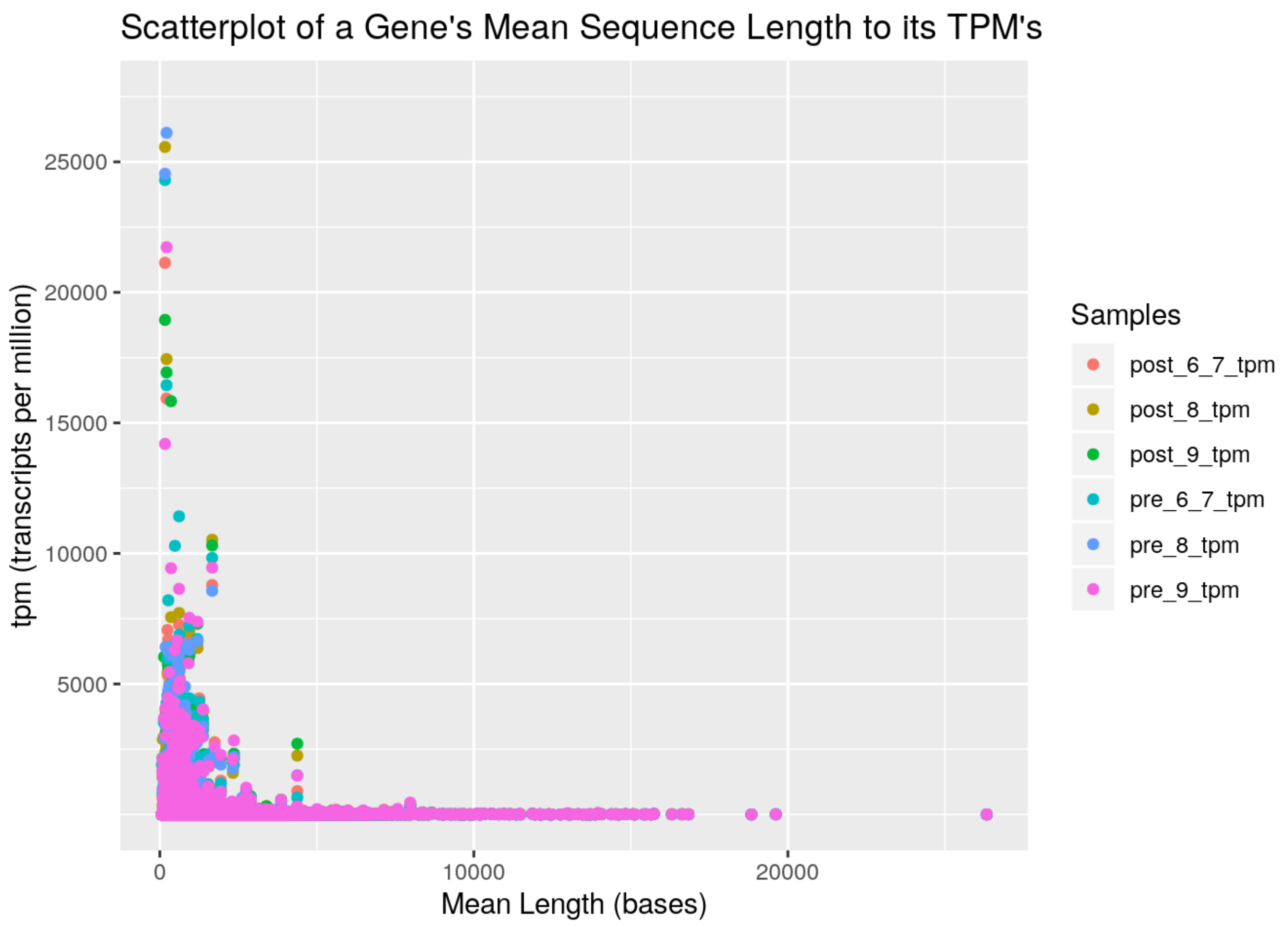
```
#create a new column with the percentile that the corresponding tpm value is seen, group the dat
a be having and not having tpm present, filter for only the post_9 sample, grab the columns cont
aining the log length counts and the newly created column and arrange this column in increasing
 numerical order and then find the mean of the tpm_percentile.
```

Wrangling Calculations

For samples post_6_7, post_9, and pre_8, all of their mean of the mean lenghts of the sequences that were coded were the same at 1410 bases long. The total mean of the mean effect length for all the samples was 1232 bases and the mean standard deviation for all of the samples is 1372 base. The standard deviation of the mean length of the pre_6_7 sample was 1372 bases long. When grouping my TPM presence, the mean tpm values for the post_8 sample was 93.4 tpm's(transcripts per million) for tpm present and 0 for tpm absent. When looking at tpm presence, looking at tpm's greater than 10, the mean tpm values was 0.712 tpm's for present and 0 tpm's for absent. When looking at the mean tpm values for the post_8 sample, we saw the mean to be 93.4 tpm's for present and 0 tpm's for absent. When taking the mean value of the percentile position for the tpm values, for present, we saw the mean tpm percentile to be 71.7 and for absent we saw 18.

Plot 1

```
ggplot(compiled_df_wranglin, aes(mean_lenght, tpm)) + geom_point(aes(color=sample)) + ggtitle("S
catterplot of a Gene's Mean Sequence Length to its TPM's") + labs(y = "tpm (transcripts per mill
ion)", x = "Mean Length (bases)", color = "Samples") + scale_y_continuous(lim=c(0,27500), breaks
= c(5000,10000,15000,20000,25000))
```



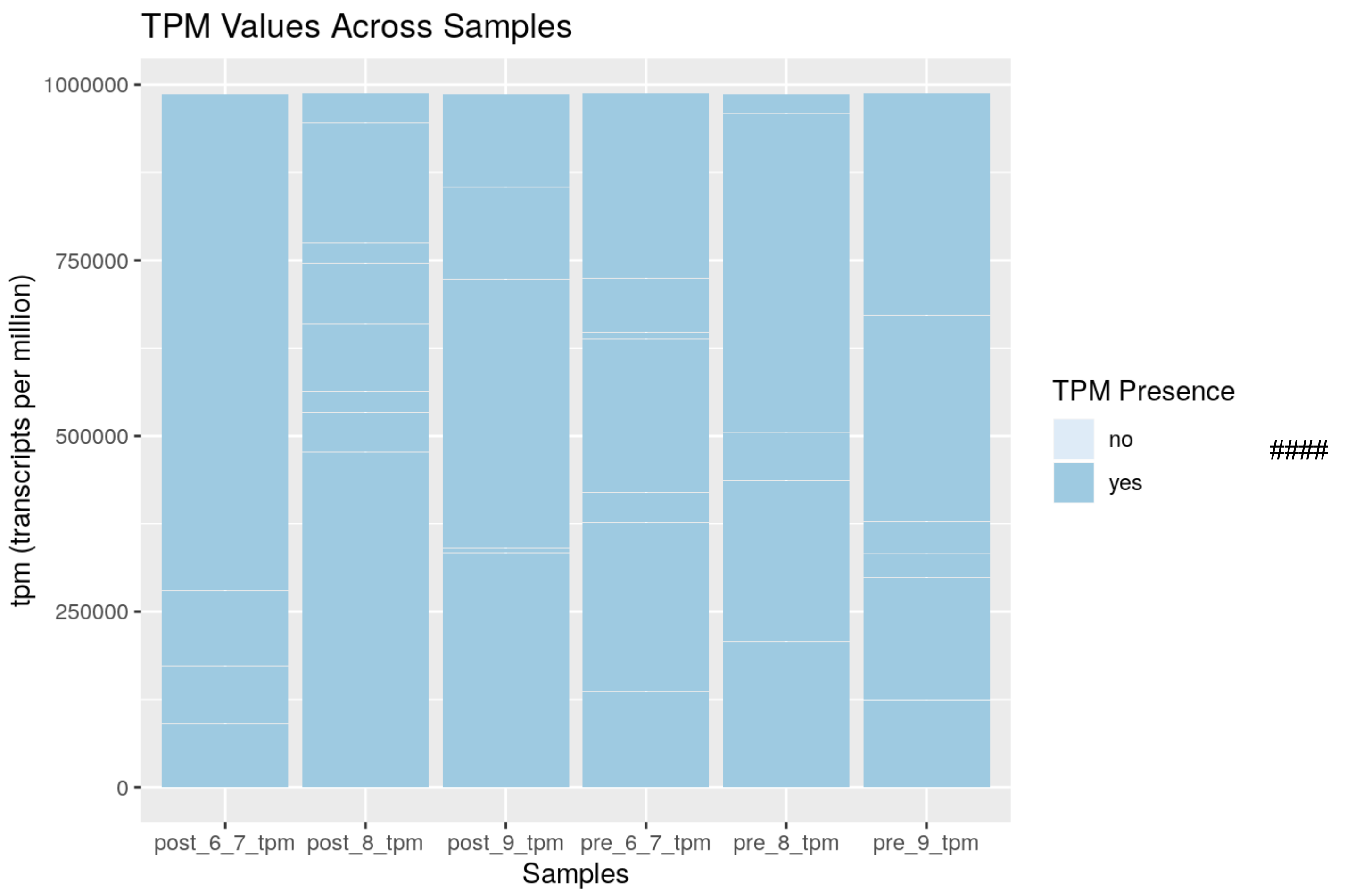
####

Plot 1 Analysis ##### This plot was to see if there was a relationship between the mean length of the base sequence of a gene and its corresponding tpm’s(transcripts per million), which is a measurement of that gene’s expression. The colors correspond to the 6 different samples that gene’s and their tpm values tie to. There seems to be a sort of inverse relationship between mean length of a gene’s sequence and its tpm values, where shorter mean lenghts correspond to higher tpm values. There are a few points a little outside of this trend, but overall the trend is inverse.

Plot 2

```
ggplot(compiled_df_wranglin, aes(x = sample, fill=tpm_presence)) + geom_bar(aes(y = tpm), stat =
"identity", fun.y = 'mean') + scale_fill_brewer() + ggtitle("TPM Values Across Samples") + labs
(y = "tpm (transcripts per million)", x = "Samples", fill = "TPM Presence")
```

```
## Warning: Ignoring unknown parameters: fun.y
```



Plot 2 ##### Looking at the 6 samples, the 6 samples all have fairly similar tpm values meaning similar expression values. This means that there may not be a significant different between pre and post sample's expressions. This is one of the reasons why TimeSeries Analysis is very important. Looking at a gene over 2 disctinct time periods may leave out gene's that work significantly over a period of time. If we made results from this plot, no genes would be found significant.

PCA's

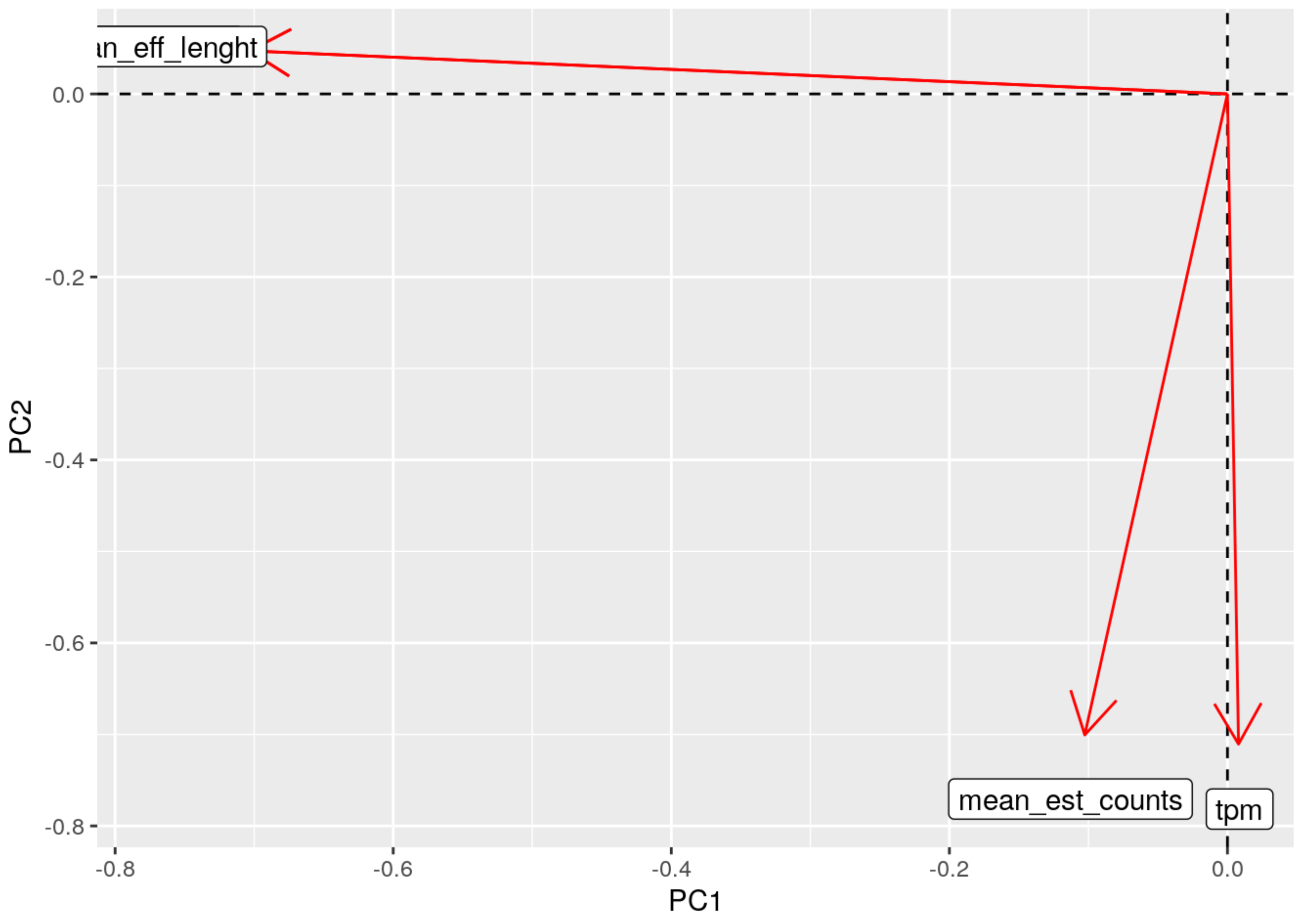
```
compiled_df_pca <- compiled_df %>% ungroup(sample) %>% select_if(is.numeric) %>% scale %>% prcom
p()

names(compiled_df_pca)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

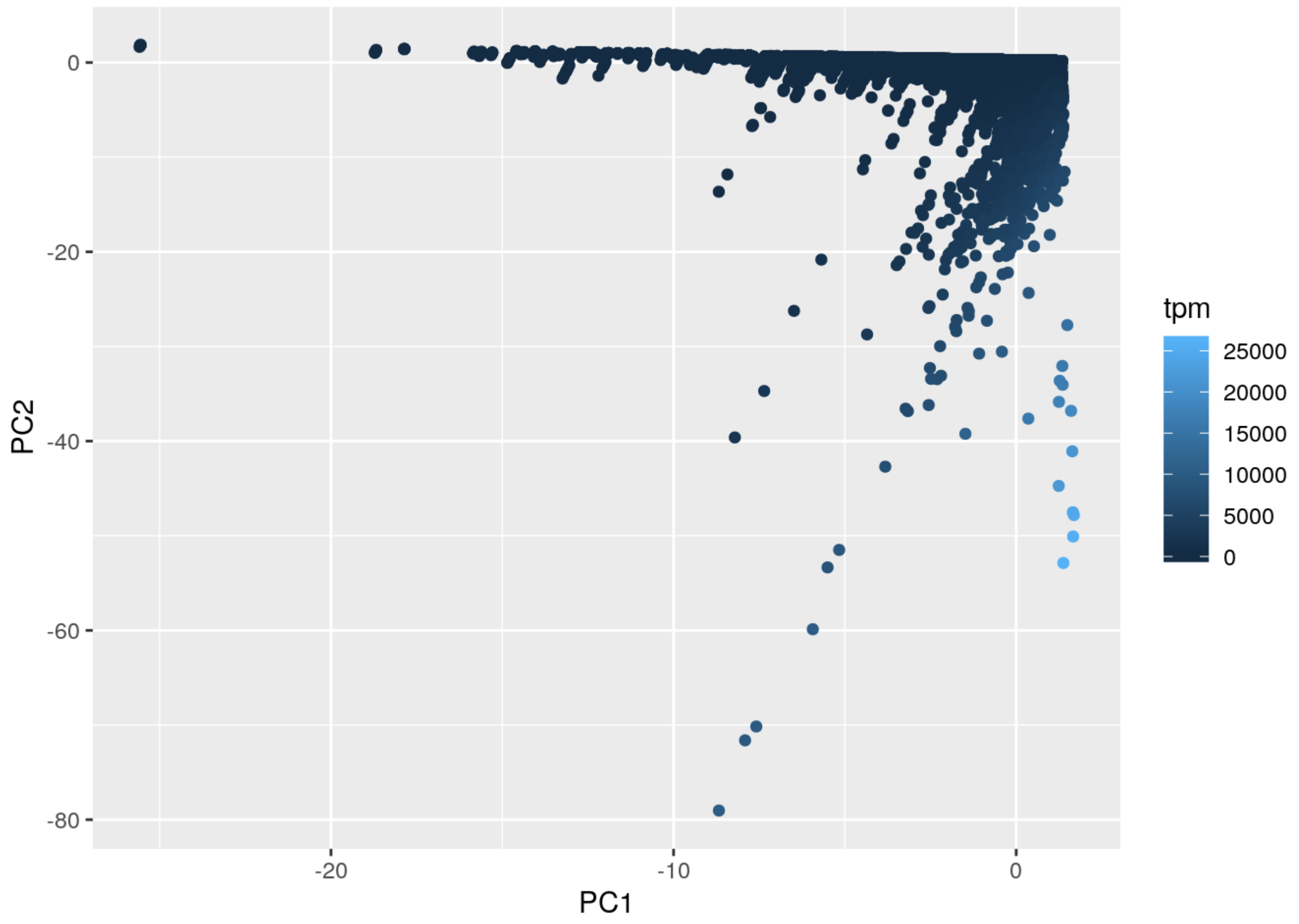
```
summary(compiled_df_pca)
```

```
## Importance of components:
##                          PC1   PC2     PC3      PC4
## Standard deviation     1.418 1.263 0.62620 0.003894
## Proportion of Variance 0.503 0.399 0.09803 0.000000
## Cumulative Proportion  0.503 0.902 1.00000 1.000000
```

```
compiled_df_pca$rotation[,1:2]%>%as.data.frame%>%rownames_to_column%>%
  ggplot()+geom_hline(aes(yintercept=0),lty=2)+
  geom_vline(aes(xintercept=0),lty=2)+ylab("PC2")+xlab("PC1")+
  geom_segment(aes(x=0,y=0,xend=PC1,yend=PC2),arrow=arrow(),col="red")+
  geom_label(aes(x=PC1*1.1,y=PC2*1.1,label=rowname))
```



```
compiled_df_pca$x%>%as.data.frame%>%mutate(tpm=compiled_df$tpm)%>%
  ggplot(aes(PC1,PC2,col=tpm))+geom_point()
```



PCA Analysis

Looking at the PCA and the Loading Plot, we see three distinct directions that is caused by variance in the data. These 3 factors are mean effect length, mean estimated counts, and tpm. The color plot of the PCA shows lighter colors to be higher tpm's and darker colors to be lower tpm's. We do see that there is much clustering in one concentration, but a linear trend in the 3 directions caused by the three factors. There is some clustering in the lighter colors which is notated to be gene's that showed higher tpm values. Across PC1, outliers were seen with gene's that showed low tpm values. Across PC2, the outliers seen either had a high estimated counts or high tpm values.