RNA-Sequencing Workflow

Mark E. Pepin, MS

03/17/2018

Table of Contents

Code Authors: Mark E. Pepin **Contact**: pepinme@uab.edu **Institution**: University of Alabama at Birmingham
Location: 542 Biomedical Research Building 2, Birmingham, AL 35294

miRNA Data Analysis

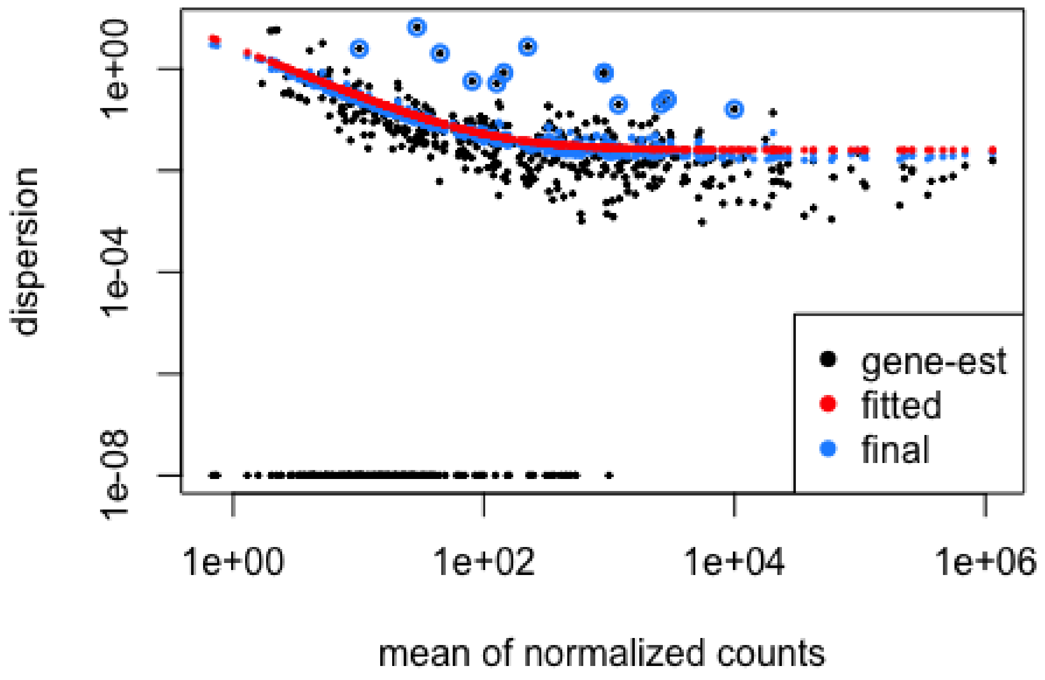
Data Pre-Processing

To determine the effect of miRNA, as well as identify the putative miRNA disregulated in a dose-dependent manner, miRNA-sequencing was performed on the same samples as before.

Count Normalization

DESeq2 (version 1.18.1) was used to perform the raw count normalization within R (version 3.4.2)

```
######### RUN DESeq2
dds_mir<-DESeqDataSetFromMatrix(countData=countData_mir.filtered, colData =
colData_mir.filtered, design= ~Treatment)
dds_mir

## class: DESeqDataSet
## dim: 623 6
## metadata(1): version
## assays(1): counts
## rownames(623): mmu-miR-1929-5p mmu-miR-1930-5p ... mmu-miR-215-3p
##   mmu-miR-7236-3p
## rowData names(0):
## colnames(6): Counts_TGL0 Counts_TGL1 ... Counts_TGH1 Counts_TGH2
## colData names(3): Sample_ID Transgene Treatment

dds_mir$Transgene<-relevel(dds_mir$Transgene, ref = "NO") # setting the
reference to wild-type genotype. only relevant for factors.
#Determine the Dispersion Relationship (determines which distribution to use
for the differential analysis) - should take about 2 minutes
dds_mir <- estimateSizeFactors(dds_mir)
dds_mir <- estimateDispersions(dds_mir)
plotDispEsts(dds_mir)
```



Plot Illustrating the corrective effect of count normalization on the dispersion estimates.

There appears to be a linear negative correlation between the mean and dispersion estimates, so the parametric "Wald" model should be an appropriate fit for differential expression analysis. Furthermore, we could get away with the parametric fit-type, but the run-time is not significantly impaired, allowing us to use the 'local' fit-type. NOTE: If it were nonlinear throughout, we would require a 'local' nonparametric fit-type.

Differential Expression Analysis

```
##Pre-Filter to reduce the size of this dataset (according to the DESeq2
document reccomendations)
dds_mir <- dds_mir[ rowSums(counts(dds_mir)) > 1, ]
dds_mir

## class: DESeqDataSet
## dim: 588 6
## metadata(1): version
## assays(2): counts mu
## rownames(588): mmu-miR-1929-5p mmu-miR-1930-5p ... mmu-miR-215-3p
##   mmu-miR-7236-3p
## rowData names(9): baseMean baseVar ... dispOutlier dispMAP
## colnames(6): Counts_TGL0 Counts_TGL1 ... Counts_TGH1 Counts_TGH2
## colData names(4): Sample_ID Transgene Treatment sizeFactor

################Run DESeq2 differential quantification (Likelihood ratio test
(LRT), instead of the wald test - easier for large datasets)
dds_mir<-DESeq(dds_mir, test="Wald", fitType="parametric")
#compile the results tables
res_mir<-DESeq2::results(dds_mir)
resdf_mir<-data.frame(res_mir)
resdf_mir$external_gene_name<-row.names(resdf_mir)
resdf_mir$ensembl_gene_id<-resdf_mir$external_gene_name
```

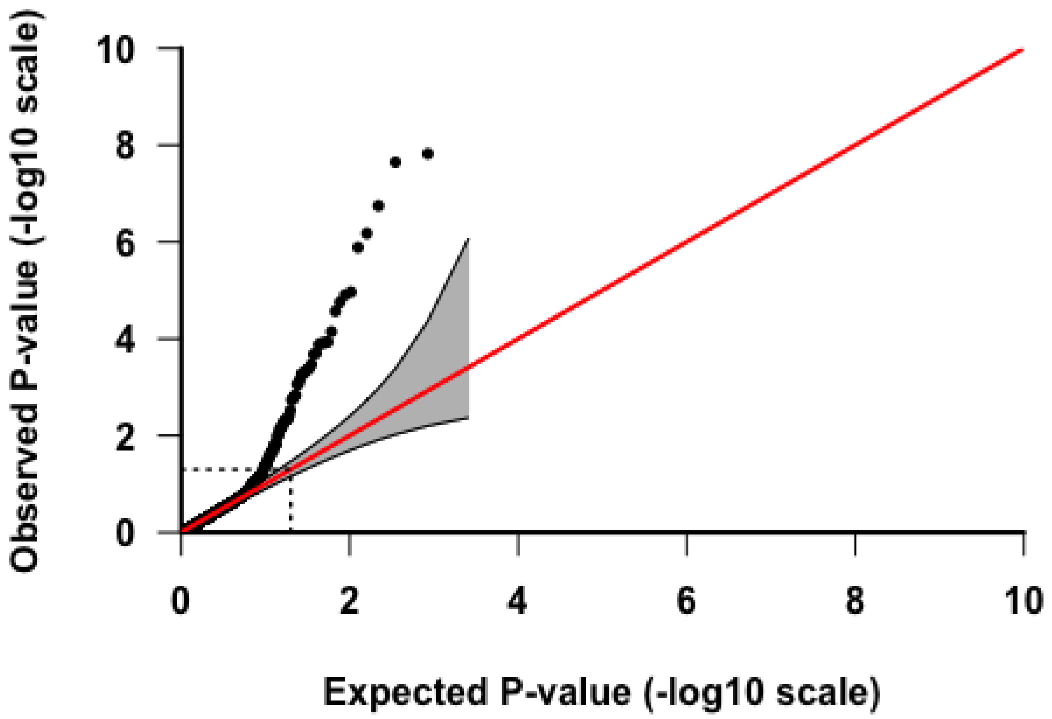
Once the differential Expression analysis was performed, the following were compiled into a results data matrix: Log2FoldChange, P-value, Bonferroni-Adjusted P-Value (Q-value), and normalized counts for each sample.

```
####Add normalized count data (for heatmap and sota)
normcount_mir<-counts(dds_mir)
normcount_mir<-as.data.frame(normcount_mir)
normcount_mir$ensembl_gene_id<-row.names(normcount_mir)
results_mir<-dplyr::left_join(resdf_mir, normcount_mir, by="ensembl_gene_id")
rownames(results_mir)<-results_mir$external_gene_name
```

QQ Plot

Before we examined the gene networks and pathways differentially regulated by NRF2 knockout, the first task was to determine whether transgene induction resulted in global changes. An effective way of determining this is the QQ plot, which compares the P-value distribution produced by the pairwise comparison (transgenic vs. WT mouse) to that of a random normal distribution. Below, it is evident that the two experimental groups produce robustly divergent expression patterns consistent with a true population difference worthy of differential expression analysis.

```
#Create Q-Q plot
results_mir<-results_mir[complete.cases(results_mir),]
write.csv(results_mir, "../2_Output/miRNA/results_miRNA_TGH.v.TGL.csv")
pQQ(results_mir$pvalue, lim=c(0,10))
```

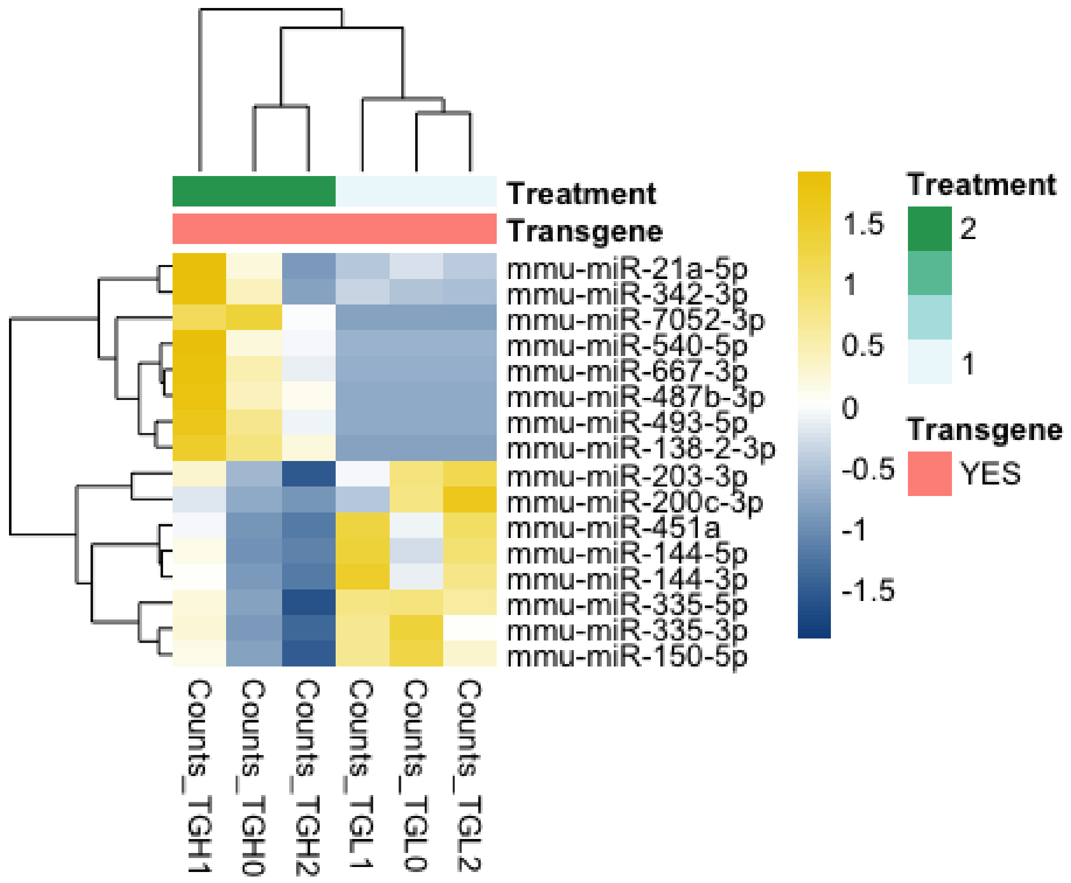


Heatmap Visualization of Transgenic vs. WT Comparison (P < 0.05)

In order to visualize the distribution of differentially expressed genes, as well as determine the effect of transgenic intensity (TGH vs. TGL), hierarchical clustering and heatmap visualization were performed at the Q < 0.05 statistical level. This analysis reveals that P < 0.05 is sufficient to separate all samples by genotype. In particular, it is apparent that the TGH group display more robust differential expression than TGL when compared to control, in a sort of dose-dependent manner.

```
library(pheatmap)
results_p05_mir<-filter(results_mir, padj<.01)
rownames(results_p05_mir)<-results_p05_mir$ensembl_gene_id
##Index file for annotating samples
rownames(colData_mir)<-colData_mir$Sample_ID
Index_mir<-dplyr::select(colData_mir, Transgene, Treatment)
```

```r
Index_mir<-as.data.frame(Index_mir)
counts_p05_mir<-dplyr::select(results_p05_mir, Counts_TGL0:Counts_TGH2)

paletteLength <- 100
myColor <- colorRampPalette(c("dodgerblue4", "white",
"gold2"))(paletteLength)
pheatmap(counts_p05_mir,
         cluster_cols=T,
         border_color=NA,
         cluster_rows=T,
         scale = 'row',
         show_colnames = T,
         show_rownames = T,
         color = myColor,
         annotation_col = Index_mir)
```



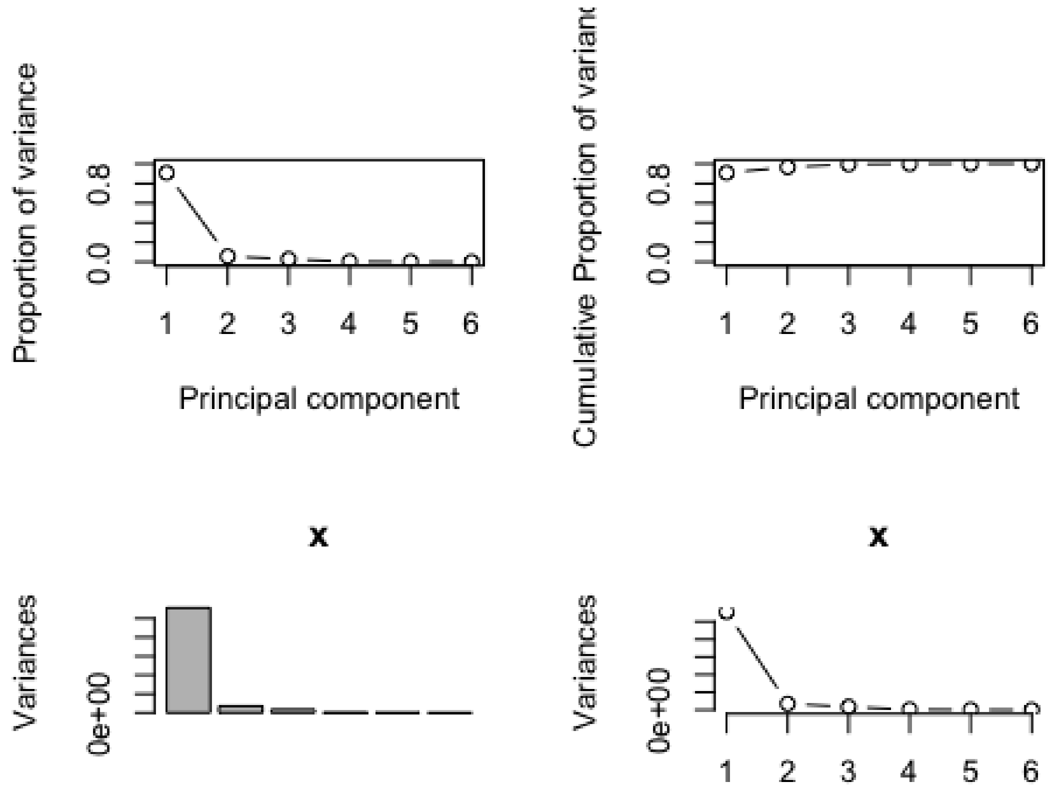
Principal Components Analysis

Once we established that the populations under consideration truly display divergene expression patterns, we sought to determine whether unbiased global gene expression patterns recapitulate TGL versus TGH phenotypes within the transgenic group. To accomplish this, an unsupervised Principal Components Analysis (PCA) was initially used with normalized counts.

PCA Features

Before running the principal components analysis, it was necessary to first determine the number of PC's required to account for 80% of the variance, a machine-learning algorithmm benchmark that provides sufficient confidence in the analysis.

```r
#Plot Features of the PCA
library(readxl)
library(dplyr)
library(plotly)
##Import the data to be used for PCA
results_dmir<-dplyr::select(results_mir, Counts_TGL0:Counts_TGH2)
results_dmir<-results_dmir[order(-rowSums(results_dmir)),]
#transpose the dataset (required for PCA)
data.pca_dmir<-t(results_dmir)
data.pca_dmir<-as.data.frame(data.pca_dmir)
##Import the data to be used for annotation
rownames(colData_mir)<-colData_mir$Sample_ID
Index_mir<-dplyr::select(colData_mir, Transgene, Treatment)
Index_mir<-as.data.frame(Index_mir)
##merge the file
data.pca_Final_dmir<-merge(Index_mir, data.pca_dmir, by=0)
rownames(data.pca_Final_dmir)<-data.pca_Final_dmir$Row.names
pca.comp_dmir<-
prcomp(data.pca_Final_dmir[,(ncol(Index_mir)+2):ncol(data.pca_Final_dmir)])

pcaCharts=function(x) {
    x.var <- x$sdev ^ 2
    x.pvar <- x.var/sum(x.var)
    par(mfrow=c(2,2))
    plot(x.pvar,xlab="Principal component",
         ylab="Proportion of variance", ylim=c(0,1), type='b')
    plot(cumsum(x.pvar),xlab="Principal component",
         ylab="Cumulative Proportion of variance",
         ylim=c(0,1),
         type='b')
    screeplot(x)
    screeplot(x,type="l")
    par(mfrow=c(1,1))
}
pcaCharts(pca.comp_dmir)
```



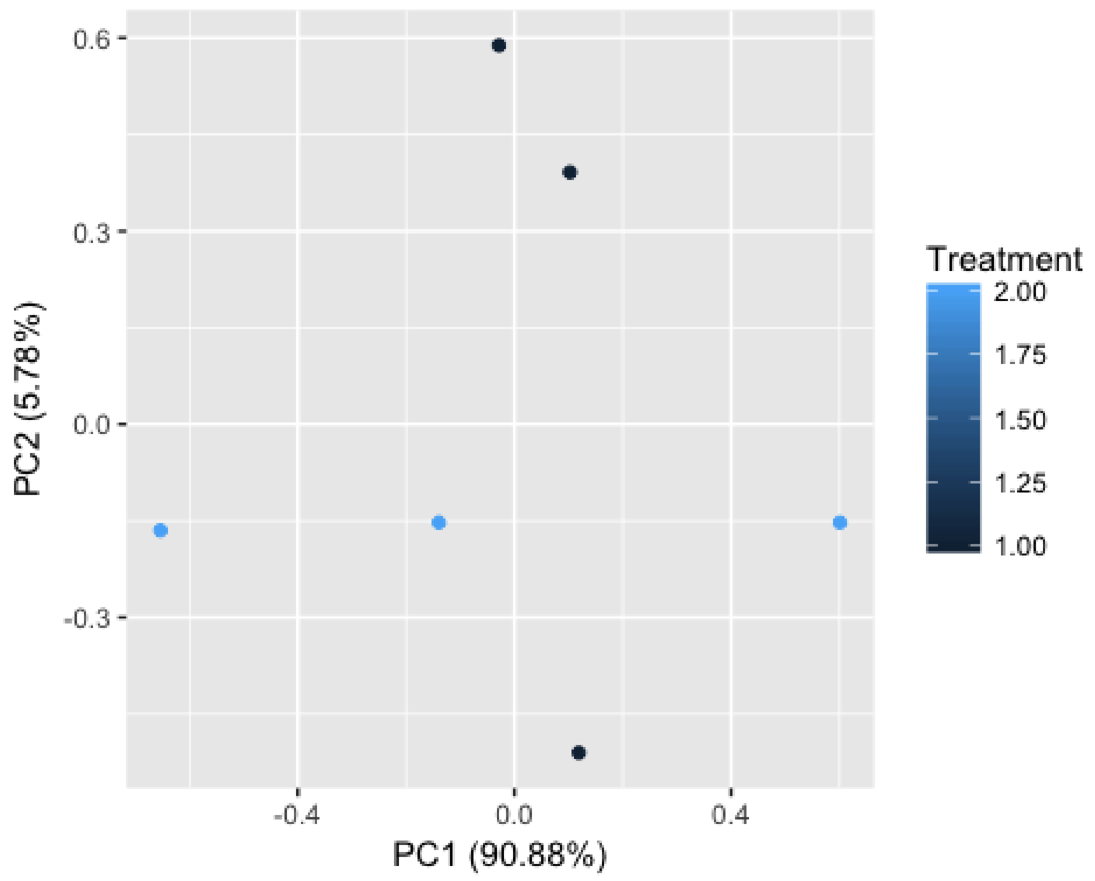
3-Dimensional PCA of miRNA

From the previous calculations, it is seens that again only 2 principal components are necessary (accounting for >80% cumulative variance). Nonetheless, below is a 3-D PCA to ensure that all groups are characterize to higher-degree of stringency.

```
##Create a 3D-PCA for Inspection
PCs_mir<-merge(pca.comp_dmir$x, Index_mir, by=0)
rownames(PCs_mir)<-PCs_mir$Row.names
ax_text<-list(
  family = "times",
  size = 12,
  color = "black")
t <- list(
  family = "times",
  size = 14,
  color = "black")
p_mir <- plot_ly(PCs_mir, x = ~PC1, y = ~PC2, z = ~PC3,
   marker = list(color = ~Treatment,
                 colorscale = c('#FFE1A1', '#683531'),
                 showscale = TRUE)) %>%
  add_markers() %>%
  layout(scene = list(
     xaxis = list(title = 'PC1', zerolinewidth = 4,
        zerolinecolor="darkgrey", linecolor="darkgrey",
        linewidth=4, titlefont=t, tickfont=ax_text),
     yaxis = list(title = 'PC2', zerolinewidth = 4,
        zerolinecolor="darkgrey", linecolor="darkgrey",
        linewidth=4, titlefont=t, tickfont=ax_text),
    zaxis = list(title = 'PC3', zerolinewidth = 4,
        zerolinecolor="darkgrey",  linecolor="darkgrey",
        linewidth=4, titlefont=t, tickfont=ax_text)),
  annotations = list(
          x = 1.13,
          y = 1.03,
          text = 'Transgene',
          xref = '1',
          yref = '0',
          showarrow = FALSE))
p_mir #must comment out for PDF generation via knitr (Pandoc)
```

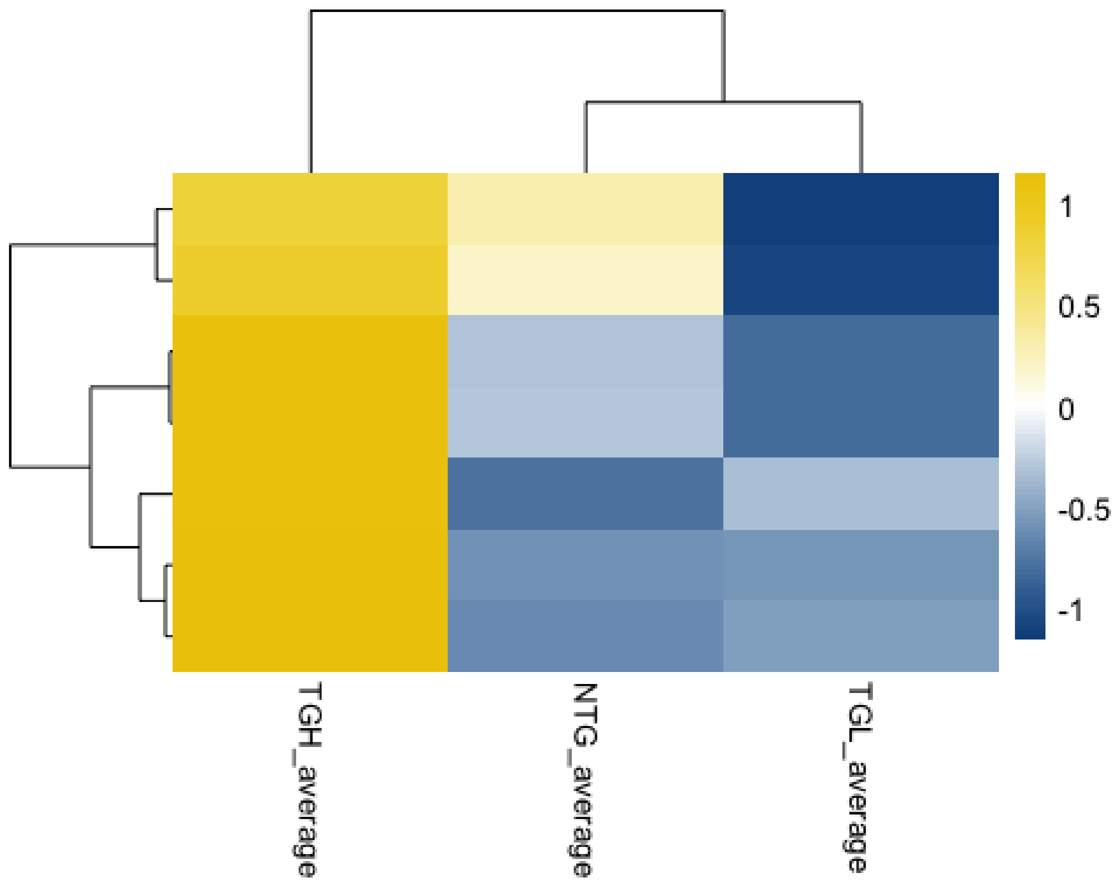
2-D PCA

```
library(ggfortify)
library(cluster)
autoplot(pca.comp_dmir, data = data.pca_Final_dmir, colour = "Treatment")
```

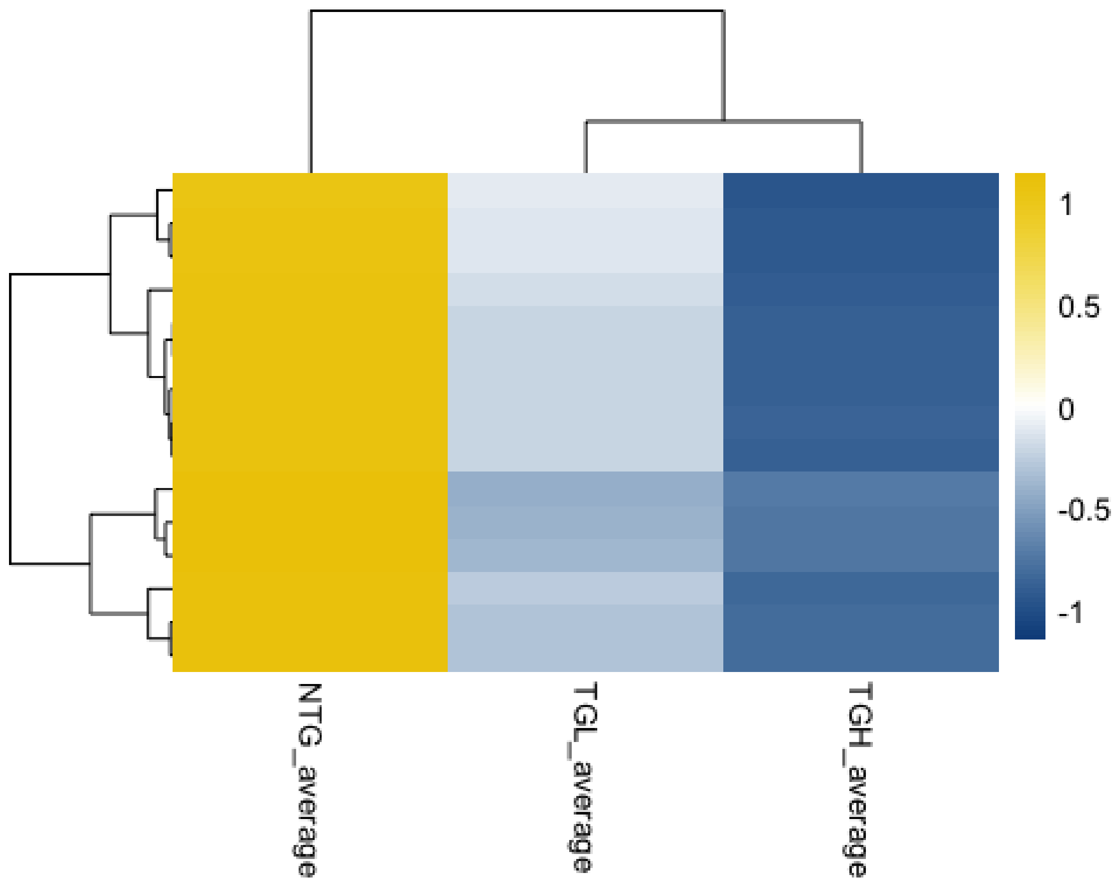


Merging the Data (Transgenic Effect and Dose Effect)

```r
library(dplyr)
##
Data_miRNA<-
read.csv("../3_Results/miRNA/_Merge_VennPlex/VP.OUT_TG.Dose_180315.p05.csv")
vector_0<-Data_miRNA$region_0
vector_2<-Data_miRNA$region_2
vector_6<-Data_miRNA$region_6
vector_8<-Data_miRNA$region_8
vector_7<-Data_miRNA$region_7
vector_3<-Data_miRNA$region_3
vector_5<-Data_miRNA$region_5

##Import the data.frame to be subsetted (T2_ND in this case) - Try the
Methylation (to facilitate further subsetting)
results_TG<-read.csv("../2_Output/miRNA/results_miRNA_TG.v.NTG.csv")
rownames(results_TG)<-results_TG$X
results_TG.filter<-dplyr::select(results_TG, ensembl_gene_id,
baseMean_TG=baseMean, log2FoldChange_TG=log2FoldChange, lfcSE_TG=lfcSE,
stat_TG=stat, pvalue_TG=pvalue, padj_TG=padj, Counts_control0:Counts_TGH2)
results_TG.filter<-dplyr::mutate(results_TG.filter,
NTG_average=((Counts_control0 + Counts_control1 + Counts_control2 +
Counts_control2)/3))
results_TG.filter<-dplyr::mutate(results_TG.filter, TGL_average=((Counts_TGL0
+ Counts_TGL1 + Counts_TGL2)/3))
results_TG.filter<-dplyr::mutate(results_TG.filter, TGH_average=((Counts_TGH0
+ Counts_TGH1 + Counts_TGH2)/3))

results_Dose<-read.csv("../2_Output/miRNA/results_miRNA_TGH.v.TGL.csv")
rownames(results_Dose)<-results_Dose$X
results_Dose.filter<-dplyr::select(results_Dose, ensembl_gene_id,
baseMean_TG=baseMean, log2FoldChange_TG=log2FoldChange, lfcSE_TG=lfcSE,
stat_TG=stat, pvalue_TG=pvalue, padj_TG=padj, Counts_TGL0:Counts_TGH2)

#Use Dplyr to subset the dataframe based on the vector (Transgenic Effect)
Annotated_0<-filter(results_TG.filter, ensembl_gene_id %in% vector_0)
Annotated_2<-filter(results_TG.filter, ensembl_gene_id %in% vector_2)
Annotated_6<-filter(results_TG.filter, ensembl_gene_id %in% vector_6)
Annotated_8<-filter(results_TG.filter, ensembl_gene_id %in% vector_8)
Annotated_7<-filter(results_TG.filter, ensembl_gene_id %in% vector_7)
Annotated_3<-filter(results_Dose.filter, ensembl_gene_id %in% vector_3)
Annotated_5<-filter(results_Dose.filter, ensembl_gene_id %in% vector_5)
#Add the fold-changes for dose effect
Annotated_6.Full<-left_join(Annotated_6, results_Dose, by="ensembl_gene_id")
Annotated_7.Full<-left_join(Annotated_7, results_Dose, by="ensembl_gene_id")
Annotated_8.Full<-left_join(Annotated_8, results_Dose, by="ensembl_gene_id")

# library(xlsx)
# write.xlsx(Annotated_0, "../3_Results/miRNA.Dose_Venn.p05.xlsx",
sheetName="TG_UP",
#   col.names=TRUE, row.names=FALSE, append=TRUE)
# write.xlsx(Annotated_2, "../3_Results/miRNA.Dose_Venn.p05.xlsx",
sheetName="TG_DOWN",
#   col.names=TRUE, row.names=FALSE, append=TRUE)
# write.xlsx(Annotated_3, "../3_Results/miRNA.Dose_Venn.p05.xlsx",
sheetName="DOSE_UP",
#   col.names=TRUE, row.names=FALSE, append=TRUE)
# write.xlsx(Annotated_5, "../3_Results/miRNA.Dose_Venn.p05.xlsx",
sheetName="DOSE_DOWN",
#   col.names=TRUE, row.names=FALSE, append=TRUE)
# write.xlsx(Annotated_6.Full, "../3_Results/miRNA.Dose_Venn.p05.xlsx",
sheetName="BOTH_UP",
#   col.names=TRUE, row.names=FALSE, append=TRUE)
# write.xlsx(Annotated_8.Full, "../3_Results/miRNA.Dose_Venn.p05.xlsx",
sheetName="BOTH_DOWN",
#   col.names=TRUE, row.names=FALSE, append=TRUE)
# write.xlsx(Annotated_7, "../3_Results/miRNA.Dose_Venn.p05.xlsx",
sheetName="BOTH_INVERSE",
#   col.names=TRUE, row.names=FALSE, append=TRUE)

##Overlap_UP
library(RColorBrewer)
library(pheatmap)
Index<-read.csv("../1_Input/Index_Justin.csv")
rownames(Index)<-Index$Sample_ID
Index<-dplyr::select(Index, -Sample_ID)
heatmap_both.up<-select(Annotated_6, NTG_average:TGH_average)
rownames(heatmap_both.up)<-Annotated_6$ensembl_gene_id
heatmap_both.up<-data.matrix(heatmap_both.up)

paletteLength <- 100
myColor <- colorRampPalette(c("dodgerblue4", "white",
"gold2"))(paletteLength)
pheatmap(heatmap_both.up,
         cluster_cols=T,
         border_color=NA,
         cluster_rows=T,
         scale = 'row',
         show_colnames = T,
         show_rownames = F,
         color = myColor)
```



```
heatmap_both.down<-select(Annotated_8, NTG_average:TGH_average)
rownames(heatmap_both.down)<-Annotated_8$ensembl_gene_id
heatmap_both.down<-data.matrix(heatmap_both.down)
pheatmap(heatmap_both.down,
         cluster_cols=T,
         border_color=NA,
         cluster_rows=T,
         scale = 'row',
         show_colnames = T,
         show_rownames = F,
         color = myColor)
```


1
0.5
0
-0.5
-1
NTG_average
TGL_average
TGH_average