Transcription Analysis

Eli Markworth

2024-02-06

SRP Data: Expression Before and After Treatment

Libraries in Use

```
library(DESeq2)
library(tidyverse)
```

###Get Data

I am using the SRP data as supplied by Bioconductor. The scaled counts contain information on the genes and their expression levels. These expression levels are measured for each subject in the data. The metadata file will be used to add proper labels when creating the DESeq data object.

```
count.data <- read.csv("SRP026387_scaledcounts.csv")
head(count.data)
```

```
##           ensgene SRR923920 SRR923921 SRR923923 SRR923924 SRR923925 SRR923927
## 1 ENSG00000000003       159      1508      2062      1369      1625       401
## 2 ENSG00000000005         0         0        17         7         0         0
## 3 ENSG00000000419       416       312       505       404       609       376
## 4 ENSG00000000457       529       507       718       851       687       722
## 5 ENSG00000000460       214       197       330       279       249       213
## 6 ENSG00000000938       170       278       100       142       174       537
##   SRR923926 SRR923929 SRR923928 SRR923930 SRR923931 SRR923933 SRR923932
## 1      2024      1075       719      1104      1441      1125      1376
## 2         0         7         8         0         6         6         2
## 3       754       387       324       454       652       640       620
## 4       599       705       689       855      1016       883       671
## 5       217       281       220       184       379       383       269
## 6       214       206       255       110       253       147       185
```

```
count.metadata <- read.csv("SRP026387_metadata.csv")
head(count.metadata)
```

```
##          id replicate prepost
## 1 SRR923920        R1     Pre
## 2 SRR923921        R2     Pre
## 3 SRR923923        R4     Pre
## 4 SRR923924        R5     Pre
## 5 SRR923925        R6     Pre
## 6 SRR923927        R1    Post
```

I want to double check that the subject IDs are the same in each file.

```
names(count.data)[-1]
```

```
##  [1] "SRR923920" "SRR923921" "SRR923923" "SRR923924" "SRR923925" "SRR923927"
##  [7] "SRR923926" "SRR923929" "SRR923928" "SRR923930" "SRR923931" "SRR923933"
## [13] "SRR923932"
```

```
count.metadata$id
```

```
##  [1] "SRR923920" "SRR923921" "SRR923923" "SRR923924" "SRR923925" "SRR923927"
##  [7] "SRR923926" "SRR923929" "SRR923928" "SRR923930" "SRR923931" "SRR923933"
## [13] "SRR923932"
```

```
all(names(count.data)[-1] == count.metadata$id)
```

```
## [1] TRUE
```

###Make DESeq2 Object

We make the DESeq dataset object below, using the *prepost* variable as our design. The tidy variable is set to true, so that the row names become the gene names.

```
dds <- DESeqDataSetFromMatrix(countData = count.data,
                              colData = count.metadata,
                              design = ~prepost,
                              tidy = TRUE)
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
dds
```

```
## class: DESeqDataSet
## dim: 57408 13
## metadata(1): version
## assays(1): counts
## rownames(57408): ENSG00000000003 ENSG00000000005 ... ENSG00000282815
##   ENSG00000282816
## rowData names(0):
## colnames(13): SRR923920 SRR923921 ... SRR923933 SRR923932
## colData names(3): id replicate prepost
```

###Run DESeq Function

Here we make the *dds* object a proper DESeq object. We also view the results in two ways, the latter being properly formatted.

```
dds <- DESeq(dds)
```

```
## estimating size factors
```

```
## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 465 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing
```

```
# Showing Results Table
res <- results(dds)
head(results(dds, tidy=TRUE))
```

```
##               row    baseMean log2FoldChange     lfcSE        stat     pvalue
## 1 ENSG00000000003 1188.515370    0.469610834 0.4077774  1.15163526 0.24947099
## 2 ENSG00000000005    3.982705   -0.023596425 1.4422348 -0.01636102 0.98694638
## 3 ENSG00000000419  484.300067    0.004836295 0.2232363  0.02166447 0.98271561
## 4 ENSG00000000457  717.208074   -0.300744253 0.1763351 -1.70552698 0.08809619
## 5 ENSG00000000460  257.353171   -0.145787591 0.2001317 -0.72845835 0.46633306
## 6 ENSG00000000938  212.355071   -0.476748710 0.3634937 -1.31157343 0.18966410
##        padj
## 1 0.5173615
## 2        NA
## 3 0.9936122
## 4 0.2818457
## 5 0.7183530
## 6 0.4428848
```

###Summary of Results

We can see a summary of the results with *summary*. It shows low outliers, but many low count values.

```
summary(res)
```

```
##
## out of 44437 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 2681, 6%
## LFC < 0 (down)     : 2231, 5%
## outliers [1]       : 250, 0.56%
## low counts [2]     : 16911, 38%
## (mean count < 5)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

###Sorting Results by P-Value

We can sort the genes by p-value, the genes with the lowest p-values being those that we are most sure had a difference between pre and post-treatment.

```
res <- res[order(res$padj),]
head(res)
```

```
## log2 fold change (MLE): prepost Pre vs Post
## Wald test p-value: prepost Pre vs Post
## DataFrame with 6 rows and 6 columns
##                    baseMean log2FoldChange     lfcSE      stat      pvalue
##                   <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000151503  6082.3980        3.45898  0.271621  12.73458 3.79904e-37
## ENSG00000249599    76.2770        3.62074  0.320647  11.29197 1.43774e-29
## ENSG00000228278   282.8524        7.47858  0.680100  10.99630 3.98156e-28
## ENSG00000116133 16803.5013        2.25829  0.250077   9.03040 1.71040e-19
## ENSG00000278709   361.7065        3.04660  0.347900   8.75711 2.00316e-18
## ENSG00000229314    98.5453        6.71850  0.775893   8.65905 4.75711e-18
##                         padj
##                    <numeric>
## ENSG00000151503 1.03721e-32
## ENSG00000249599 1.96266e-25
## ENSG00000228278 3.62349e-24
## ENSG00000116133 1.16743e-15
## ENSG00000278709 1.09380e-14
## ENSG00000229314 2.16464e-14
```

We can also filter for the genes with the biggest changes of expression, and had significant p-values (even though many of these such genes had significant p-values anyways).

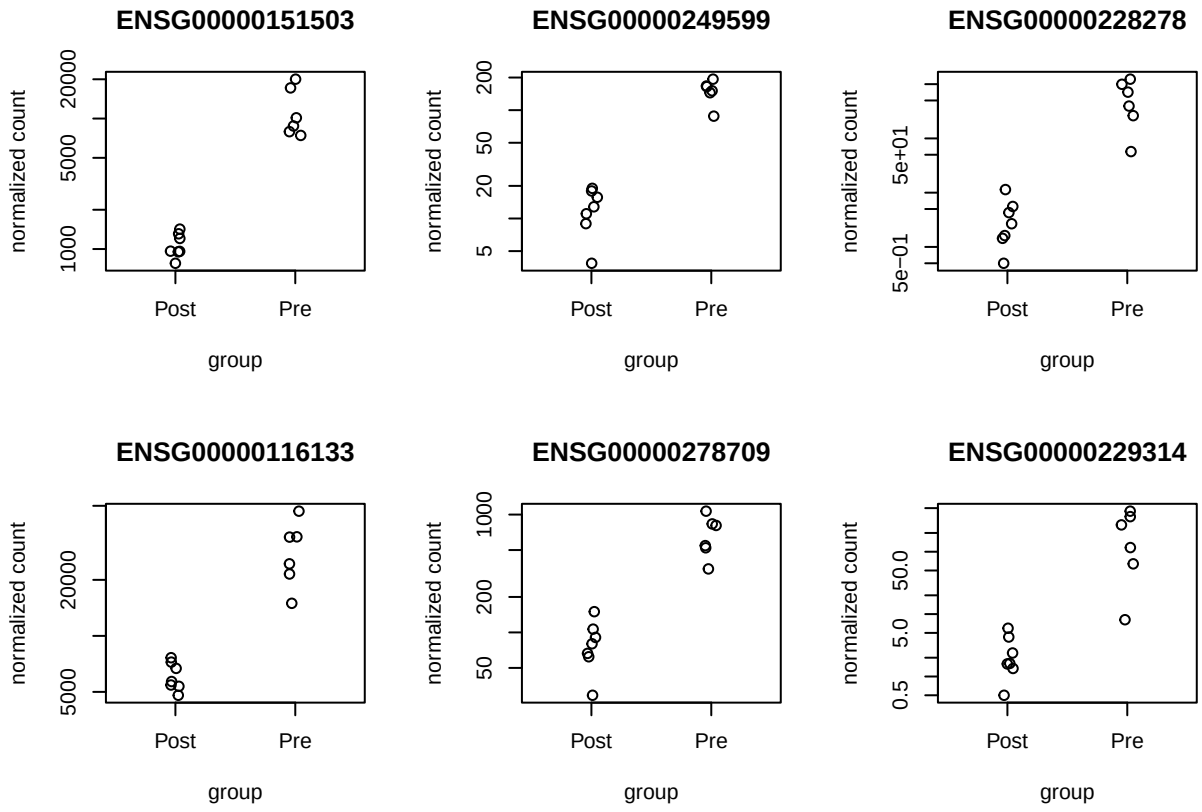
```
res.LFC <- res[which(res$padj < 0.05),]
res.LFC <- res.LFC[rev(order(abs(res.LFC$log2FoldChange))),]
head(res.LFC)
```

```
## log2 fold change (MLE): prepost Pre vs Post
## Wald test p-value: prepost Pre vs Post
## DataFrame with 6 rows and 6 columns
##                   baseMean log2FoldChange     lfcSE      stat      pvalue
##                  <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000228278 282.85242        7.47858  0.680100  10.99630 3.98156e-28
## ENSG00000104760   9.39077        6.79573  1.131138   6.00787 1.87976e-09
## ENSG00000229314  98.54531        6.71850  0.775893   8.65905 4.75711e-18
## ENSG00000248809   7.96401        6.56136  1.134880   5.78155 7.40164e-09
## ENSG00000228740   6.27509        6.21434  1.350557   4.60132 4.19830e-06
## ENSG00000278406  11.01685        6.18427  0.994526   6.21830 5.02554e-10
##                        padj
##                   <numeric>
## ENSG00000228278 3.62349e-24
## ENSG00000104760 4.54170e-07
## ENSG00000229314 2.16464e-14
## ENSG00000248809 1.41323e-06
## ENSG00000228740 2.27425e-04
## ENSG00000278406 1.64994e-07
```

###Plotting Counts, Before and After Treatment

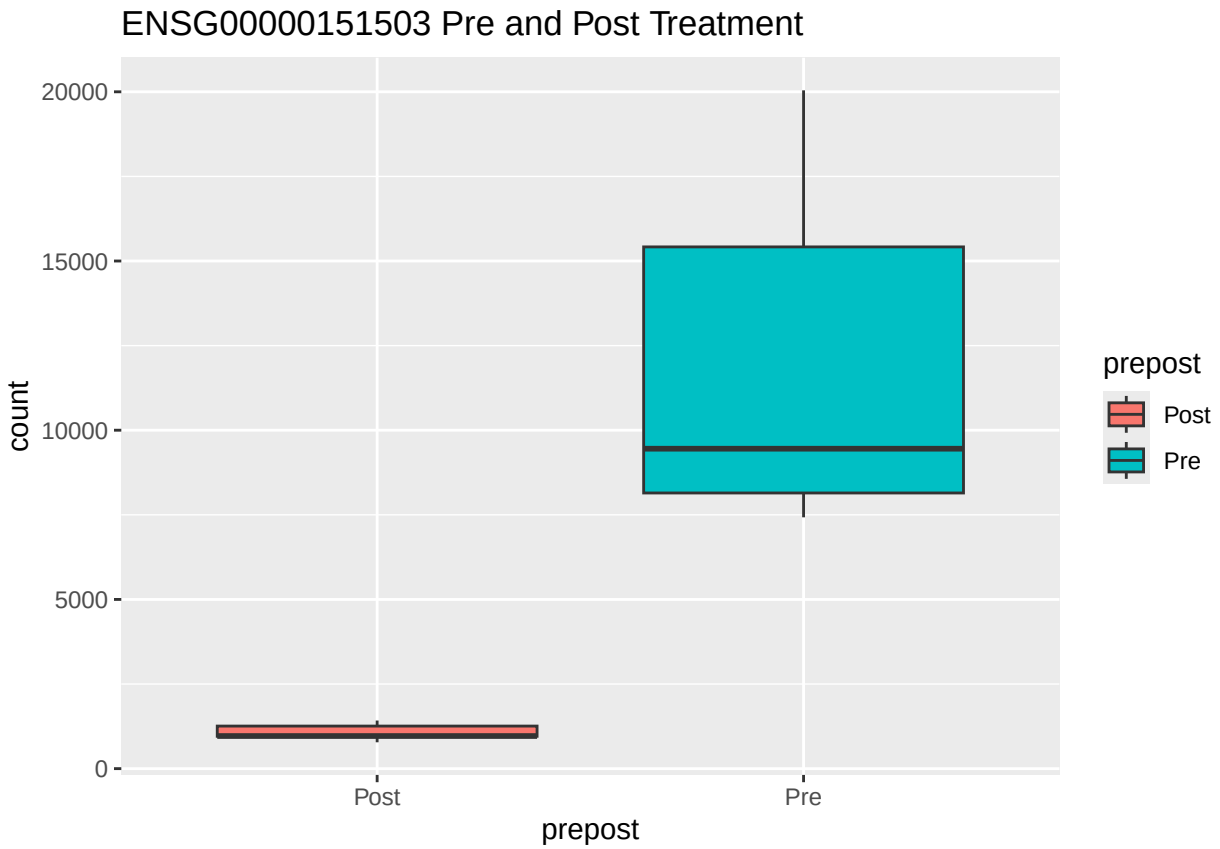
To the genes that had the most significant p-values, we can compare counts.

```
par(mfrow=c(2,3))

plotCounts(dds, gene="ENSG00000151503", intgroup="prepost")
plotCounts(dds, gene="ENSG00000249599", intgroup="prepost")
plotCounts(dds, gene="ENSG00000228278", intgroup="prepost")
plotCounts(dds, gene="ENSG00000116133", intgroup="prepost")
plotCounts(dds, gene="ENSG00000278709", intgroup="prepost")
plotCounts(dds, gene="ENSG00000229314", intgroup="prepost")
```



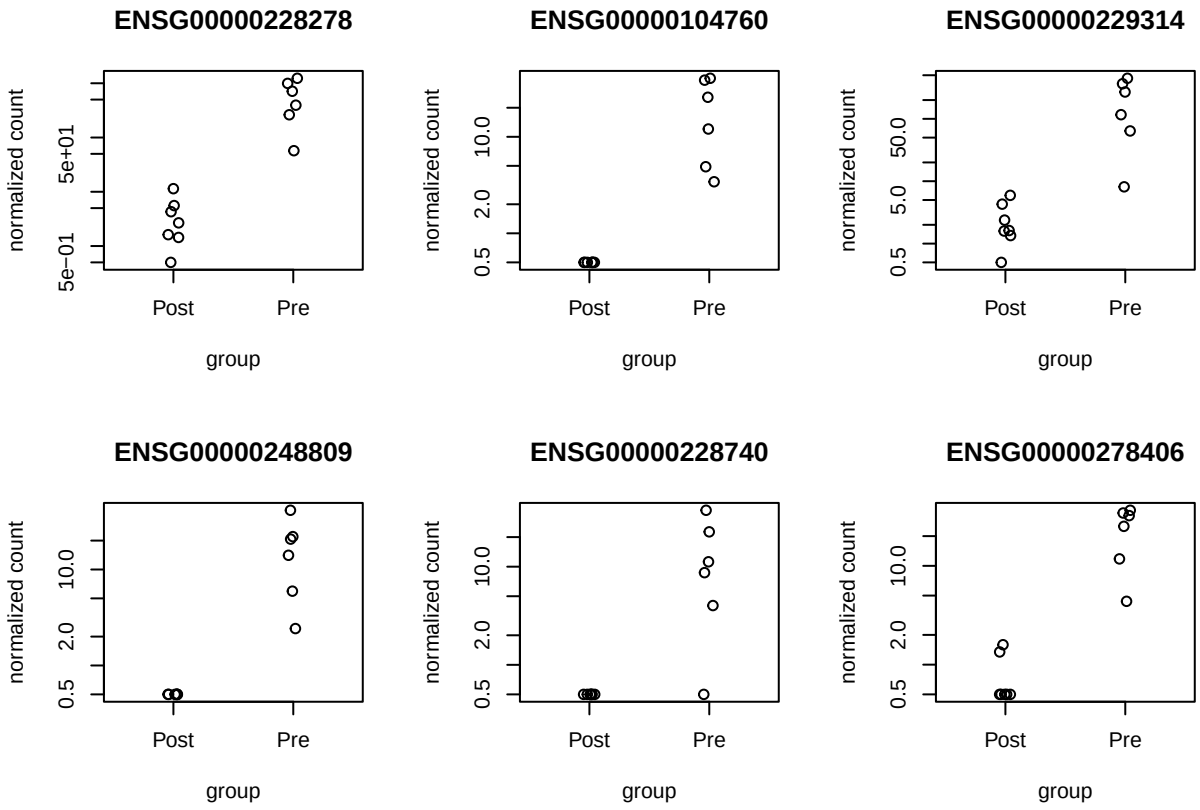
Using ggplot to enhance our visual aid.

```
plotCounts(dds, gene="ENSG00000151503", intgroup="prepost", returnData = TRUE) %>%
  ggplot(aes(prepost, count)) +
  geom_boxplot(aes(fill=prepost)) +
  ggtitle("ENSG00000151503 Pre and Post Treatment")
```



We can also compare counts from the genes that had the largest change in expression.

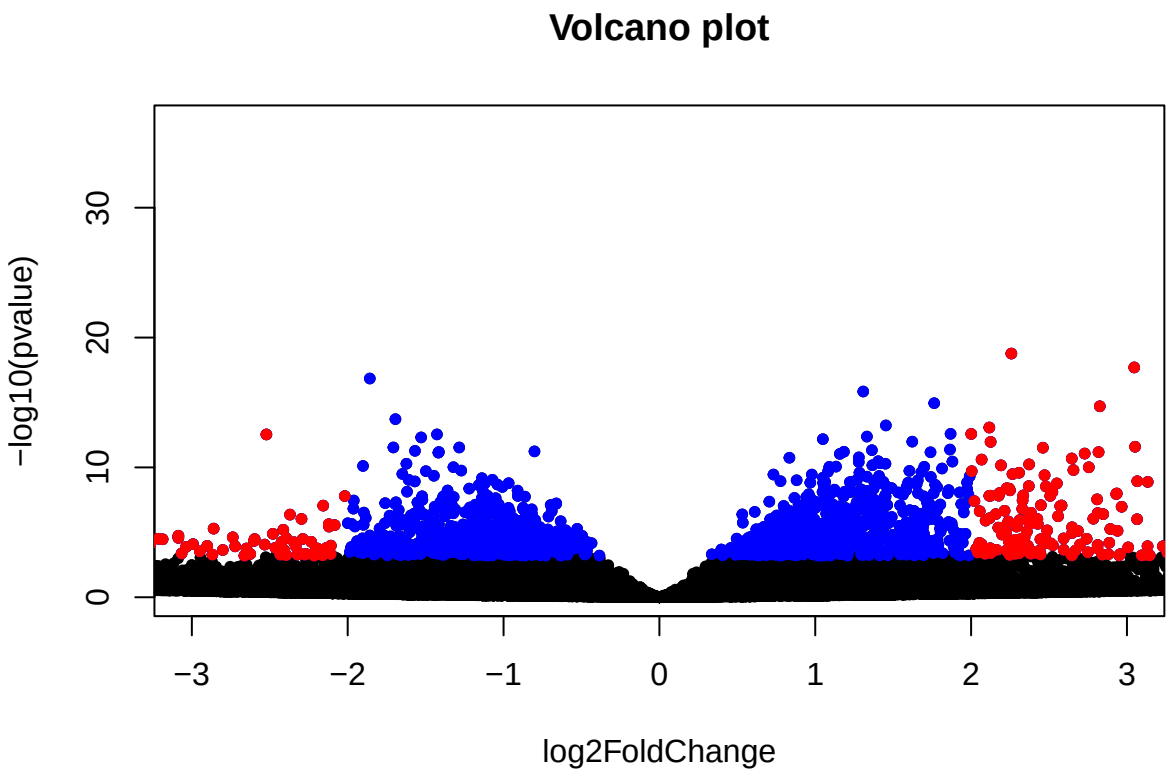
```
par(mfrow=c(2,3))

plotCounts(dds, gene="ENSG00000228278", intgroup="prepost")
plotCounts(dds, gene="ENSG00000104760", intgroup="prepost")
plotCounts(dds, gene="ENSG00000229314", intgroup="prepost")
plotCounts(dds, gene="ENSG00000248809", intgroup="prepost")
plotCounts(dds, gene="ENSG00000228740", intgroup="prepost")
plotCounts(dds, gene="ENSG00000278406", intgroup="prepost")
```



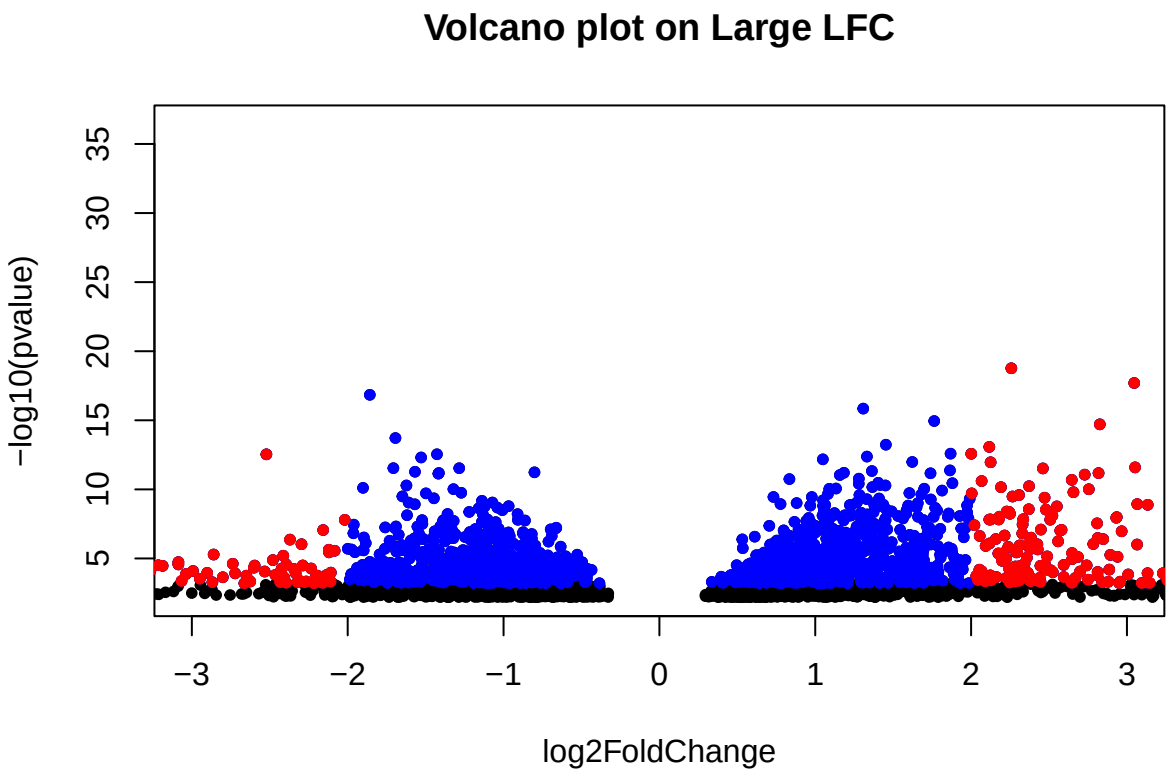
###Volcano Plot

The data in red are statistically significant and have an expression change of 4 times or more.

```
par(mfrow=c(1,1))

with(res, plot(log2FoldChange, -log10(pvalue), pch=20, main="Volcano plot", xlim=c(-3,3)))
with(subset(res, padj<.01 ), points(log2FoldChange, -log10(pvalue), pch=20, col="blue"))
with(subset(res, padj<.01 & abs(log2FoldChange)>2), points(log2FoldChange, -log10(pvalue), pch=20, col=
```



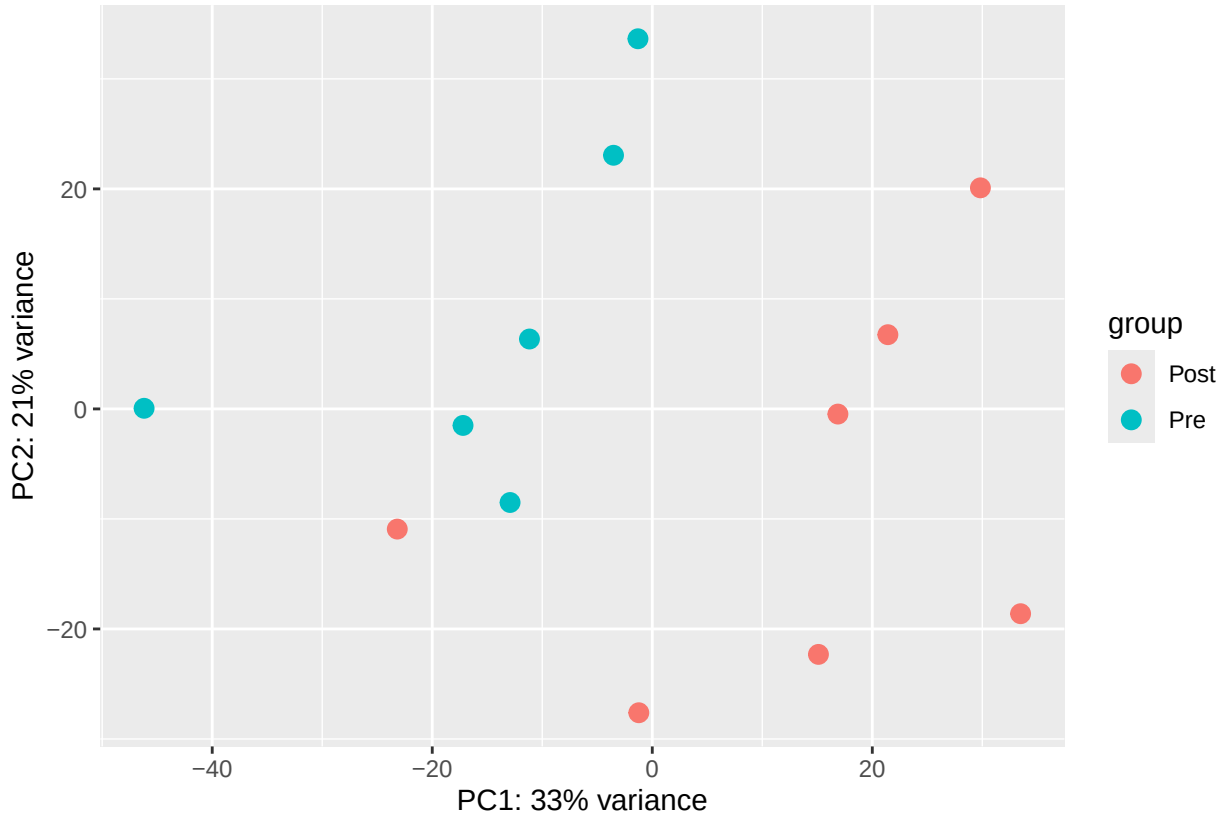

```
par(mfrow=c(1,1))

with(res.LFC, plot(log2FoldChange, -log10(pvalue), pch=20, main="Volcano plot on Large LFC", xlim=c(-3,3

with(subset(res.LFC, padj<.01 ), points(log2FoldChange, -log10(pvalue), pch=20, col="blue"))
with(subset(res.LFC, padj<.01 & abs(log2FoldChange)>2), points(log2FoldChange, -log10(pvalue), pch=20, c
```



###Principle Components Analysis

```
vstdata <- vst(dds, blind=TRUE)
plotPCA(vstdata, intgroup="prepost")
```

```
## using ntop=500 top features by variance
```



###References

Count-Based Differential Expression Analysis of RNA-seq Data. (n.d.). Bioconnector.github.io. Retrieved April 3, 2024, from https://bioconnector.github.io/workshops/r-rnaseq-airway.html

Lashlock. (n.d.). DESEQ2 R Tutorial. Lashlock.github.io. https://lashlock.github.io/compbio/R_presentation.html

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