Class 13: RNASeq Analysis

Kevin Tan (PID: A16774162)

The data for today's lab comes from a published RNA-seq experiment where airway smooth muscle cells were treated with dexamethasone, a synthetic glucocorticoid steroid with anti-inflammatory effects

Import Data

We need tow things for this analysis: counts and metadata. These are called "counutdata" and "colData" in the DESeq2 world.

```
counts <- read.csv("airway_scaledcounts.csv", row.names=1)
metadata <-  read.csv("airway_metadata.csv")
```

```
head(counts)
```

```
                SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
ENSG00000000003        723        486        904        445       1170
ENSG00000000005          0          0          0          0          0
ENSG00000000419        467        523        616        371        582
ENSG00000000457        347        258        364        237        318
ENSG00000000460         96         81         73         66        118
ENSG00000000938          0          0          1          0          2
                SRR1039517 SRR1039520 SRR1039521
ENSG00000000003       1097        806        604
ENSG00000000005          0          0          0
ENSG00000000419        781        417        509
ENSG00000000457        447        330        324
ENSG00000000460         94        102         74
ENSG00000000938          0          0          0
```

The counts are organized with a gene per row and experiment per column

```r
head(metadata)
```

```
          id     dex celltype     geo_id
1 SRR1039508 control   N61311 GSM1275862
2 SRR1039509 treated   N61311 GSM1275863
3 SRR1039512 control  N052611 GSM1275866
4 SRR1039513 treated  N052611 GSM1275867
5 SRR1039516 control  N080611 GSM1275870
6 SRR1039517 treated  N080611 GSM1275871
```

Examine the Data

> Q1. How many genes are in this dataset?

```r
nrow(counts)
```

```
[1] 38694
```

> Q2. How many 'control' cell lines do we have?

```r
sum(metadata$dex == 'control')
```

```
[1] 4
```

```r
table(metadata$dex)
```

```

control treated 
      4       4 
```

Check on match of metadata and coldata

```r
colnames(counts)
```

```
[1] "SRR1039508" "SRR1039509" "SRR1039512" "SRR1039513" "SRR1039516"
[6] "SRR1039517" "SRR1039520" "SRR1039521"
```

```r
metadata$id
```

```
[1] "SRR1039508" "SRR1039509" "SRR1039512" "SRR1039513" "SRR1039516"
[6] "SRR1039517" "SRR1039520" "SRR1039521"
```

```r
colnames(counts) == metadata$id
```

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

If you want to know that all the elements of a vector are TRUE, we can use the `all()` function

```r
all(c(T,T,T,F))
```

```
[1] FALSE
```

```r
all(colnames(counts) == metadata$id)
```

```
[1] TRUE
```

Analysis

I want to start by comparing "control" and "treated" columns. To do this, I will find the average for each gene (row) in all "control" columns. Then I will find the average in the "treated" columns. Then I will compare them.

```r
control.inds <- metadata$dex == "control"
```

```r
control_counts <- counts[,control.inds]
```

Now find the mean count value per gene using the `apply()` function.

```r
control_mean <- apply(control_counts,1,mean)
head(control_mean)
```

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
         900.75            0.00          520.50          339.75           97.25
ENSG00000000938
           0.75
```

Now do the same for the “treated” columns.

```r
treated.inds <- metadata$dex == "treated"
treated_counts <- counts[,treated.inds]
treated_mean <- apply(treated_counts,1,mean)
head(treated_mean)
```

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
         658.00            0.00          546.00          316.50           78.75
ENSG00000000938
           0.00
```

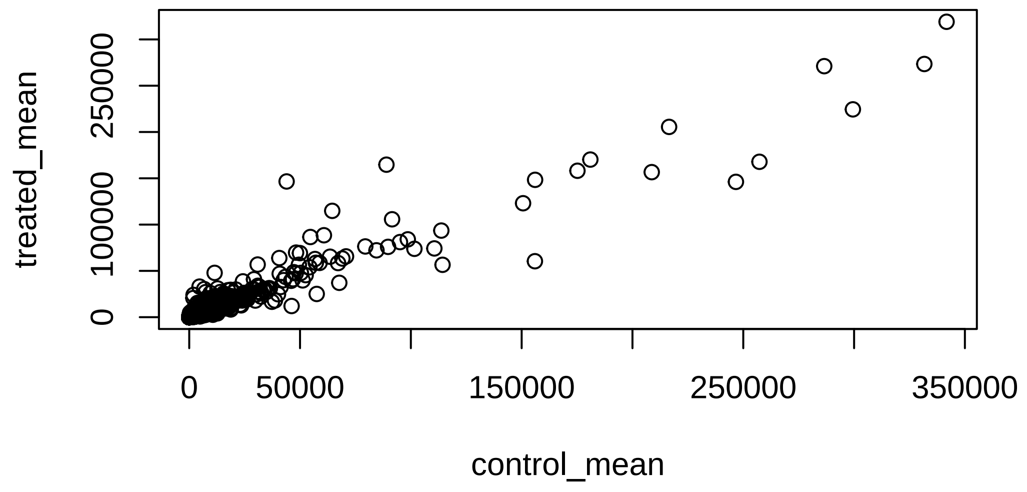
Put these two mean vectors together for ease of book-keeping

```r
meancount <- data.frame(control_mean, treated_mean)
head(meancount)
```

```
                control_mean treated_mean
ENSG00000000003       900.75       658.00
ENSG00000000005         0.00         0.00
ENSG00000000419       520.50       546.00
ENSG00000000457       339.75       316.50
ENSG00000000460        97.25        78.75
ENSG00000000938         0.75         0.00
```

Lets have a look with a plot

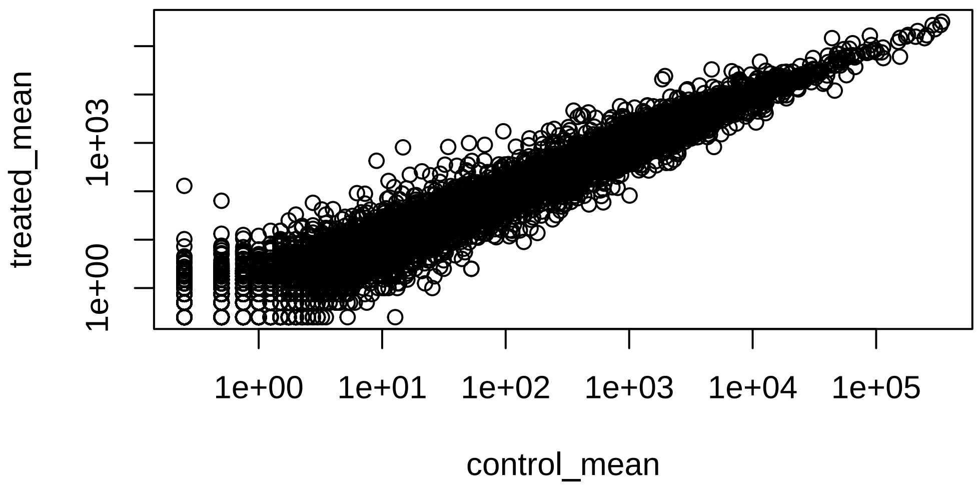
```r
plot(meancount)
```



```r
plot(meancount, log="xy")
```

```
Warning in xy.coords(x, y, xlabel, ylabel, log): 15032 x values <= 0 omitted
from logarithmic plot

Warning in xy.coords(x, y, xlabel, ylabel, log): 15281 y values <= 0 omitted
from logarithmic plot
```



```r
log2(20/10)
```

```
[1] 1
```

```r
log2(5/10)
```

```
[1] -1
```

Log2 is a useful scale to use allowing us to better determine the magnitude of change.

We most often work in log2 units because they have a more intuitive interpretation.

Here we calculate the log2 Fold-change of treated/control values and add it to our dataframe of results.

```r
meancount$log2fc <- log2(meancount$treated_mean / meancount$control_mean)
head(meancount)
```

```
                control_mean treated_mean      log2fc
ENSG00000000003       900.75       658.00 -0.45303916
```

```
ENSG00000000005         0.00         0.00         NaN
ENSG00000000419       520.50       546.00  0.06900279
ENSG00000000457       339.75       316.50 -0.10226805
ENSG00000000460        97.25        78.75 -0.30441833
ENSG00000000938         0.75         0.00        -Inf
```

There are some funky answers in here like NaN and -inf that are a result of zero count genes in the dataset

It is common practice to filter these zero count out.

```
to.keep.inds <- (rowSums(meancount[,1:2] == 0) == 0)

mycounts <- meancount[to.keep.inds,]
head(mycounts)
```

```
                control_mean treated_mean      log2fc
ENSG00000000003       900.75       658.00 -0.45303916
ENSG00000000419       520.50       546.00  0.06900279
ENSG00000000457       339.75       316.50 -0.10226805
ENSG00000000460        97.25        78.75 -0.30441833
ENSG00000000971      5219.00      6687.50  0.35769358
ENSG00000001036      2327.00      1785.75 -0.38194109
```

> Q. How many genes do we have left after zero-count filtering

```
nrow(mycounts)
```

```
[1] 21817
```

A common threshold for calling a gene "up" or down" is a log2 fold change of +2 or -2.

> Q. how many "up" regulated genes do we have?

```
sum(mycounts$log2fc >= 2)
```

```
[1] 314
```

DESeq Analysis

We need to do this analysis properly through testing for significance

```r
library("DESeq2")
```

To use DESeq we need to get our input data in a very particular format

```r
dds <- DESeqDataSetFromMatrix(countData = counts,
                       colData = metadata,
                       design = ~dex)
```

```
converting counts to integer mode

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

Run DEseq Analysis

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

```
estimating size factors

estimating dispersions

gene-wise dispersion estimates

mean-dispersion relationship

final dispersion estimates

fitting model and testing
```

```r
res <- results(dds)
res
```

```
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 38694 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat    pvalue
                <numeric>      <numeric> <numeric> <numeric> <numeric>
ENSG00000000003  747.1942     -0.3507030  0.168246 -2.084470 0.0371175
ENSG00000000005    0.0000             NA        NA        NA        NA
ENSG00000000419  520.1342      0.2061078  0.101059  2.039475 0.0414026
ENSG00000000457  322.6648      0.0245269  0.145145  0.168982 0.8658106
ENSG00000000460   87.6826     -0.1471420  0.257007 -0.572521 0.5669691
...                   ...            ...       ...       ...       ...
ENSG00000283115  0.000000             NA        NA        NA        NA
ENSG00000283116  0.000000             NA        NA        NA        NA
ENSG00000283119  0.000000             NA        NA        NA        NA
ENSG00000283120  0.974916      -0.668258   1.69456 -0.394354  0.693319
ENSG00000283123  0.000000             NA        NA        NA        NA
                     padj
                <numeric>
ENSG00000000003  0.163035
ENSG00000000005        NA
ENSG00000000419  0.176032
ENSG00000000457  0.961694
ENSG00000000460  0.815849
...                   ...
ENSG00000283115        NA
ENSG00000283116        NA
ENSG00000283119        NA
ENSG00000283120        NA
ENSG00000283123        NA
```

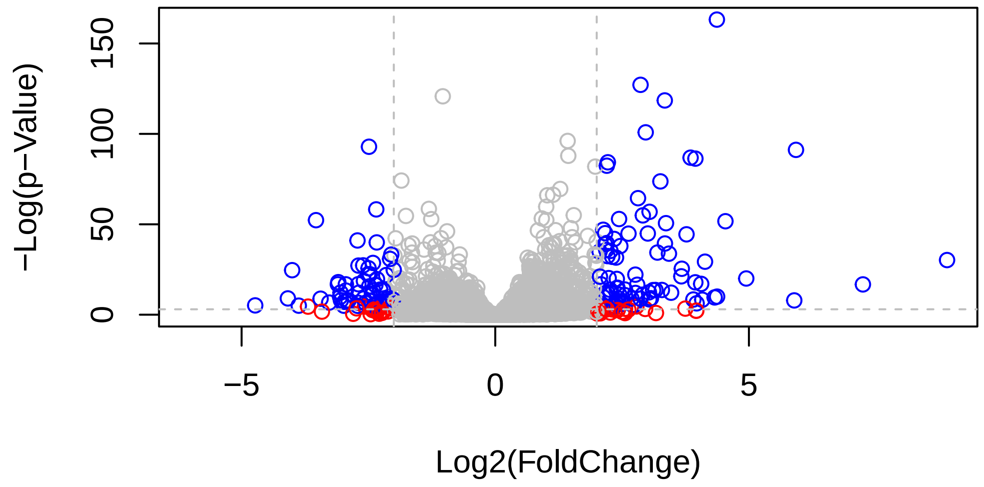
we get an padj because regular p-value is susceptible to giving a false positive if too many groups are being compared.

I want to make a figure showing an overiew of all my results to date. A plot of **log2 fold change** vs the **p-value** (adjusted p-value)

```r
# Color Vector
mycols <- rep("gray", nrow(res))
mycols[abs(res$log2FoldChange) >2 ] <- "red"

inds <- (res$padj <0.01) & (abs(res$log2FoldChange) > 2 )
mycols[inds] <- "blue"
```

```
#Plot
plot(res$log2FoldChange, -log(res$padj), col=mycols, ylab = "-Log(p-Value)", xlab="Log2(Fo
abline(v=-2, col="gray", lty=2)
abline(v=2, col="gray", lty=2)
abline(h=-log(0.05), col="gray", lty=2)
```


−Log(p−Value)
Log2(FoldChange)