Class 13: RNA-Seq Part 1

Dylan Mullaney (A16869793)

Today we will analyze from a published RNA-Seq experiment where airway smooth muscle tissue cells were treated with dexamenthasone, a synthetic glucocortocoid steroid with anti-inflammatory effects (Himes et. al, 2014).

Import countData and colData

There are two datasets I need to import/read

- `countData` the trenascript counts per gene (rows) in the different experiments
- `colData` information (ie. methods) about the columns (ie experiments) in countData

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

We can have a peek at these with `head()`

```
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
```

```
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
7 SRR1039520 control  N061011 GSM1275874
8 SRR1039521 treated  N061011 GSM1275875
```

> Q1. How many genes are in this dataset?

38694

```
nrow(counts)
```

```
[1] 38694
```

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

4 control cells

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

```

control treated 
      4       4 
```

```
#sum(metadata$dex == "control")
```

How do I start to analyze or compare all this data? Let's take the average of values under similar treatments. –> mean value per gene for all "control" and compare to those for all "treated".

- Extract all "control" columns from the **counts** data (this is tough because the part we can comprehend exists in the metadata)
- Find the mean value for each gene –> across the rows

1.

```
control.inds <- metadata$dex == "control"
control.counts <- counts[ ,control.inds]
```

2.

```
control.mean <- rowSums(control.counts)/ ncol(control.counts)
head(control.mean)
```

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

> Q3. How would you make the above code in either approach more robust? Is there a function that could help here?

Instead of div by 4 for column number, use ncol()

> Q4. Follow the same procedure for the treated samples (i.e. calculate the mean per gene across drug treated samples and assign to a labeled vector called treated.mean)

```
treated.inds <- metadata$dex == "treated"
treated.counts <- counts[ ,treated.inds]
treated.mean <- rowSums(treated.counts)/ ncol(treated.counts)
head(treated.mean)
```

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

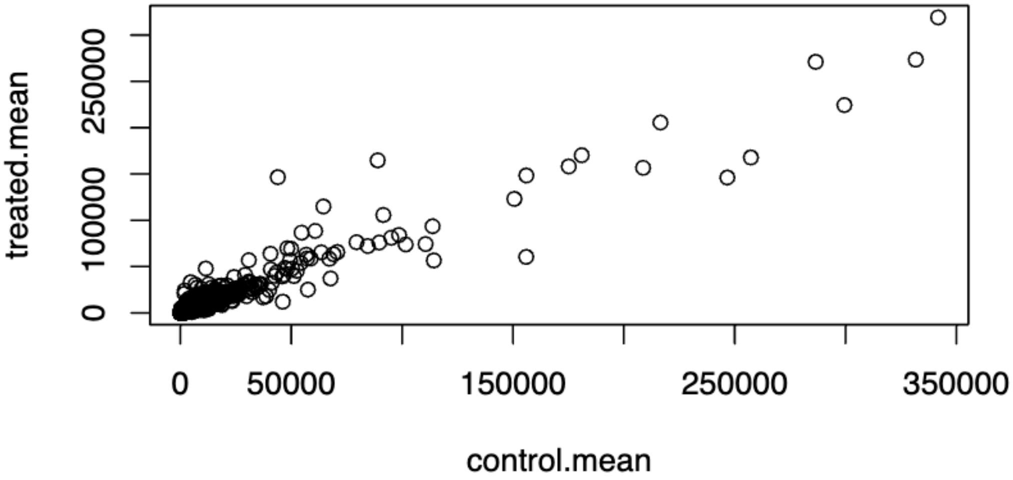
Let's plot the values

```
meancounts <- data.frame (control.mean, treated.mean)
head(meancounts)
```

```
                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
```

Q5 (a). Create a scatter plot showing the mean of the treated samples against the mean of the control samples. Your plot should look something like the following.

```r
plot(meancounts)
```

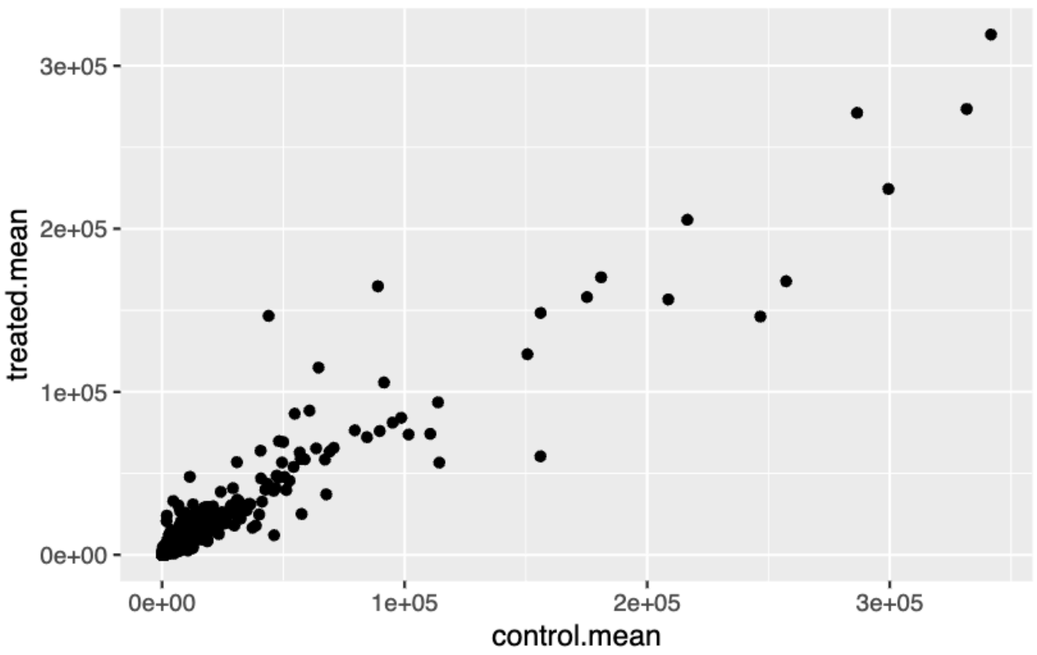


Q5 (b).You could also use the ggplot2 package to make this figure producing the plot below. What geom_?() function would you use for this plot?

geom_point()

```r
library(ggplot2)

ggplot(meancounts) +
  aes(control.mean, treated.mean) +
  geom_point()
```



If a point is on the diagonal, there is no difference in control vs. treated. Any departure from the diagonal indicates a change in expression. If a point is above the diagonal, there is an increase in gene expression.

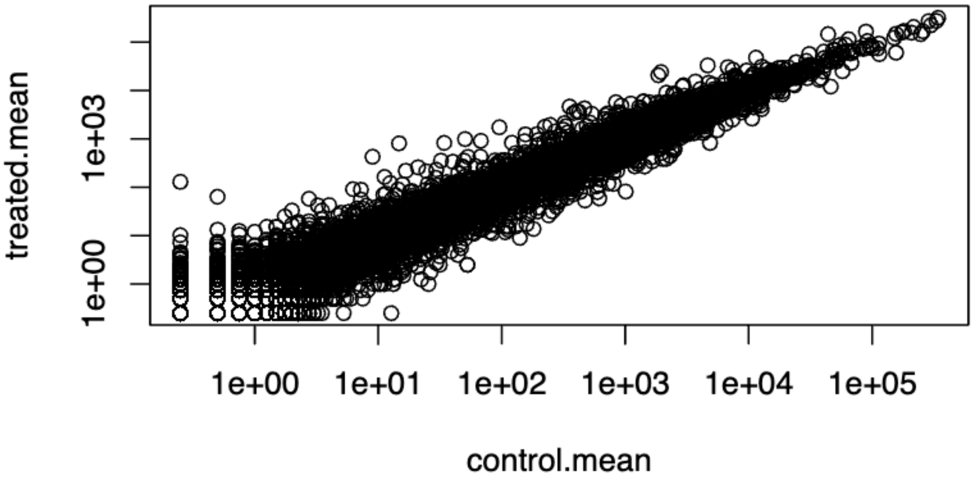
> Q6. Try plotting both axes on a log scale. What is the argument to plot() that allows you to do this?

Add log after your data, set equal to "x", "y", or "xy" for both.

```
plot(meancounts, 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
```



We most often work in log2 units as this makes the math easier. For example, this is a doubling, a logfold change of 1 when treated is 2x the amount of control. If treated is half as much as control, it will create a -1 output.

```
# treated/ control
log2(40/20)
```

```
[1] 1
```

QUESTION 7 QUESTION 7QUESTION 7QUESTION 7QUESTION 7QUESTION 7QUESTION 7QUESTION 7

We can add a "log2 fold change" value to our **meancounts** dataset and create it as a new column. NaN stands for not a number, ie log of 0/0. -Inf refers to the log of 0/any number .

```
meancounts$log2fc <- log2(meancounts$treated.mean/ meancounts$control.mean)
head (meancounts)
```

```
                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
```

We need to filter out zero count values because they're messing up the data.

How many genes are "up" regulated at the common log2 fold-change threshold of +2.

> Q8. Using the up.ind vector above can you determine how many up regulated genes we have at the greater than 2 fc level?

1846

```r
up.inds <- meancounts$log2fc > 2
sum(up.inds, na.rm=T)
```

```
[1] 1846
```

> Q9. Using the down.ind vector above can you determine how many down regulated genes we have at the greater than 2 fc level?

23348

```r
down.inds <- meancounts$log2fc < 2
sum(down.inds, na.rm=T)
```

```
[1] 23348
```

> Q10. Do you trust these results? Why or why not?

No because we're using averages and not taking into account outliers.

DESeq2 Analysis

To do this the right way we need to consider the significance of the differences not just their magnitude.

```r
library(DESeq2)
```

To use this package, it wants countData and colData in a specific 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
```

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

```
estimating size factors
```

```
estimating dispersions
```

```
gene-wise dispersion estimates
```

```
mean-dispersion relationship
```

```
final dispersion estimates
```

```
fitting model and testing
```

Extract my results

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

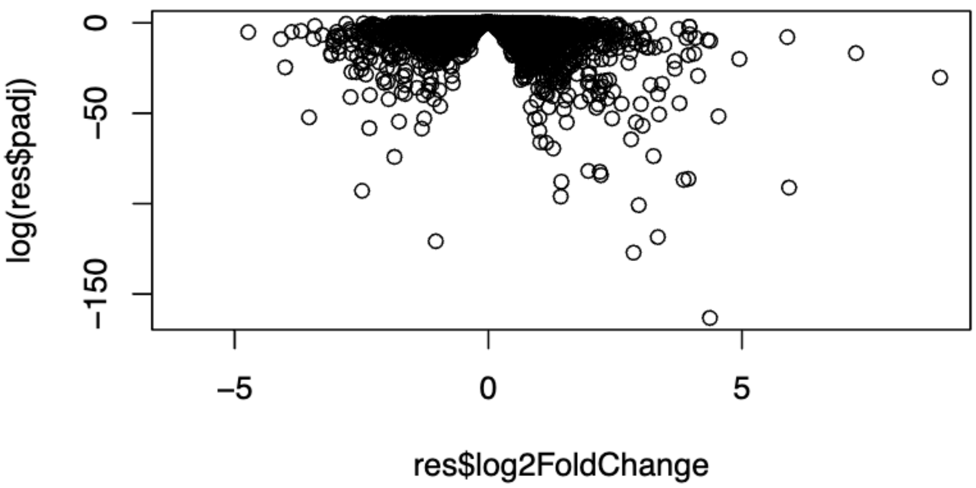
```
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 6 rows and 6 columns
                  baseMean log2FoldChange     lfcSE      stat    pvalue
                 <numeric>      <numeric> <numeric> <numeric> <numeric>
ENSG00000000003 747.194195     -0.3507030  0.168246 -2.084470 0.0371175
ENSG00000000005   0.000000             NA        NA        NA        NA
ENSG00000000419 520.134160      0.2061078  0.101059  2.039475 0.0414026
ENSG00000000457 322.664844      0.0245269  0.145145  0.168982 0.8658106
ENSG00000000460  87.682625     -0.1471420  0.257007 -0.572521 0.5669691
```

```
ENSG00000000938   0.319167      -1.7322890  3.493601 -0.495846 0.6200029
                      padj
                 <numeric>
ENSG00000000003  0.163035
ENSG00000000005        NA
ENSG00000000419  0.176032
ENSG00000000457  0.961694
ENSG00000000460  0.815849
ENSG00000000938        NA
```

Check out the pvalues, looks like there's definitely some false positives.

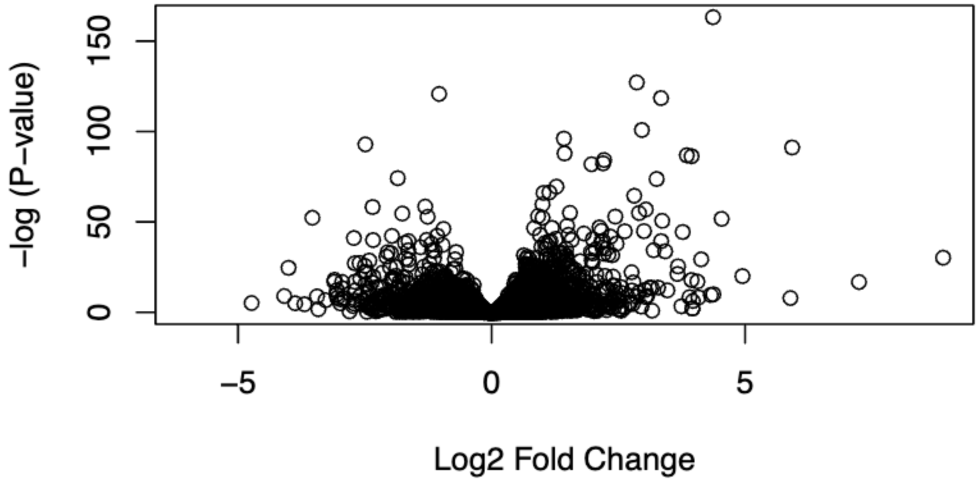
Plot of fold-change vs P-value(adjusted for multiple testing)

```
plot(res$log2FoldChange, log(res$padj))
```



Flip the axis so it's easier to look at

```
plot(res$log2FoldChange, -log(res$padj),
     xlab= "Log2 Fold Change",
     ylab= "-log (P-value)")
```



Let's save our work to date

```
write.csv(res, file="myresults.csv")
```

Let's make a nicer version of our volcano plot above - Add the log2 threshold lines at +2/-2 - Add P-value threshold lines at 0.05 - Add color to highlight the subset of genes that meet both the above thresholds

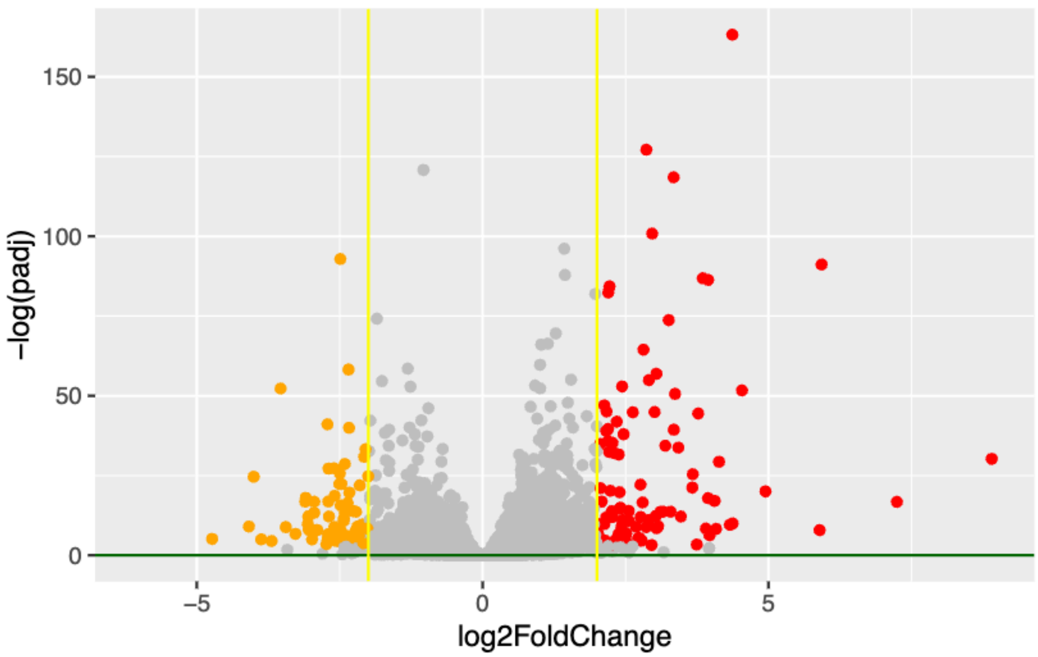
```
mycols <- rep("grey", nrow(res))

mycols [res$log2FoldChange>=2] <- "red"
mycols [res$log2FoldChange<=-2] <- "orange"

mycols[res$padj > 0.05] <- "grey"
```

```
ggplot(res) +
  aes(log2FoldChange, -log(padj))+
  geom_point(col=mycols) +
  geom_hline(yintercept=0.05, col="darkgreen") +
  geom_vline(xintercept=c(2,-2), col="yellow")
```

```
Warning: Removed 23549 rows containing missing values or values outside the scale range
(`geom_point()`).
```



The orange and red points are the ones that are interesting, where change is occurring.