Class 12: Transcriptomics and the analysis of RNA-Seq data

(Suraj Sidhu: A18512793)

Table of contents

Background

Today we will analyze some RNASeq data from Himes et al. on the effects of a common steroid (dexamethasone) on airwy smooth muscle cells (ASM cells).

Our starting point is the "counts" dat and "metadata" that contain the count values for each gene in their different experiments.

Data Import

```
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
```

Q1 How many genes are in this dataset ?

```
nrow(counts)
```

```
[1] 38694
```

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

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

```
control treated
      4       4
```

Toy differential gene expression

To start our analysis let's calculate the mean counts for all genes in the control experiments.

1. Extract all "control" columns from the `counts` object.
2. Calculate the mean for all rows (i.e. genes) of these "control" columns 3-4. Do the same for "treated"
3. Compare these `control.mean` and `treated.mean` values.

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

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

control.counts <- counts[ ,control.inds]

control.mean <- rowMeans( control.counts )

head(control.mean)
```

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

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 <- rowMeans( treated.counts )

head(treated.mean)
```

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

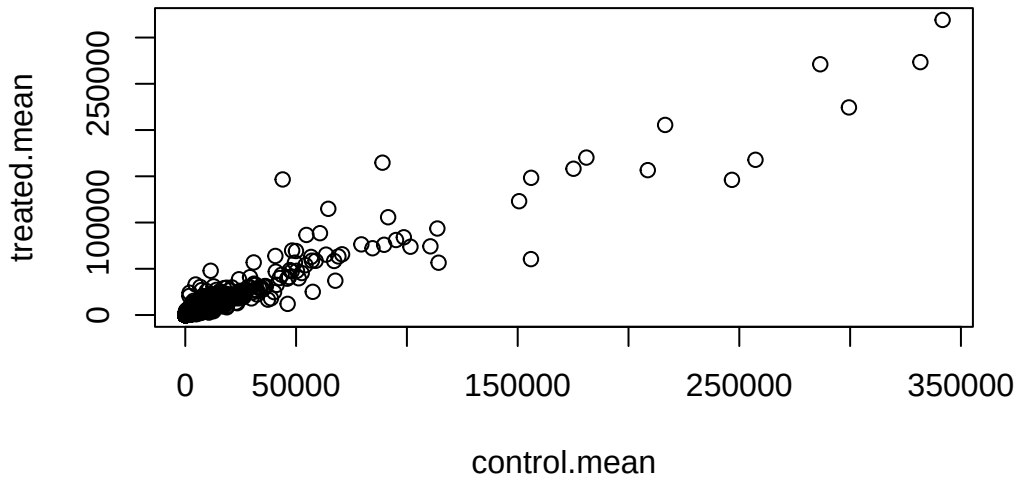
Store these together for easy bookkeeping as `meancounts`

```
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.

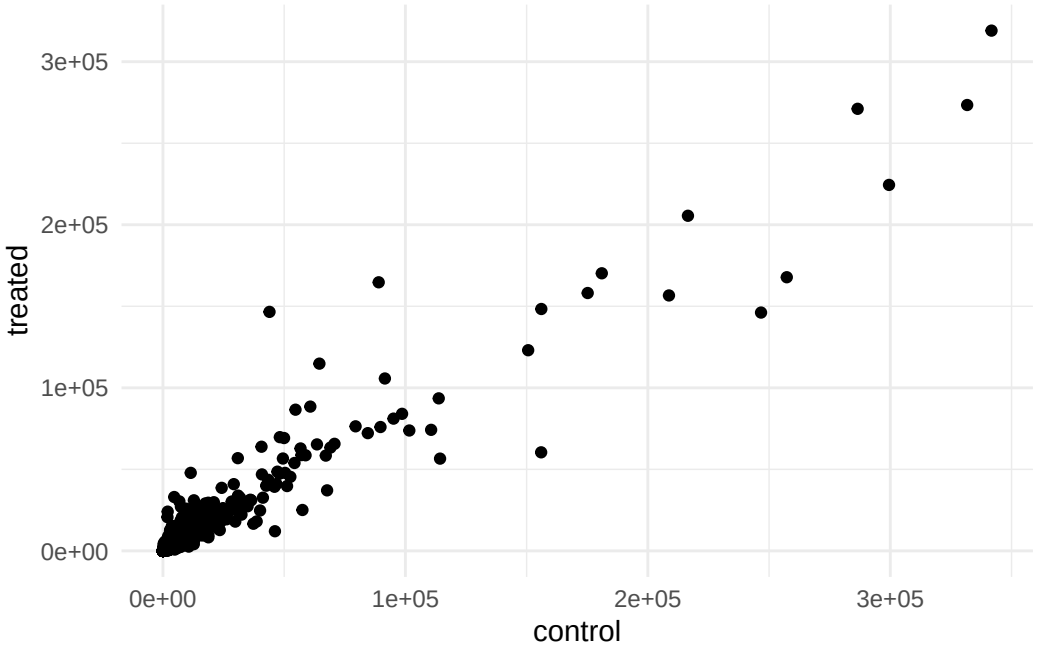
```
plot(meancounts)
```



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? We aften talk to metrics like "log2 fold-change"

```
library(ggplot2)
meancounts <- data.frame(
  control = control.mean,
  treated = treated.mean
)

ggplot(meancounts, aes(x = control, y = treated)) +
  geom_point() +
  labs() +
  theme_minimal()
```




```
## treated/control
log2(10/20)
```

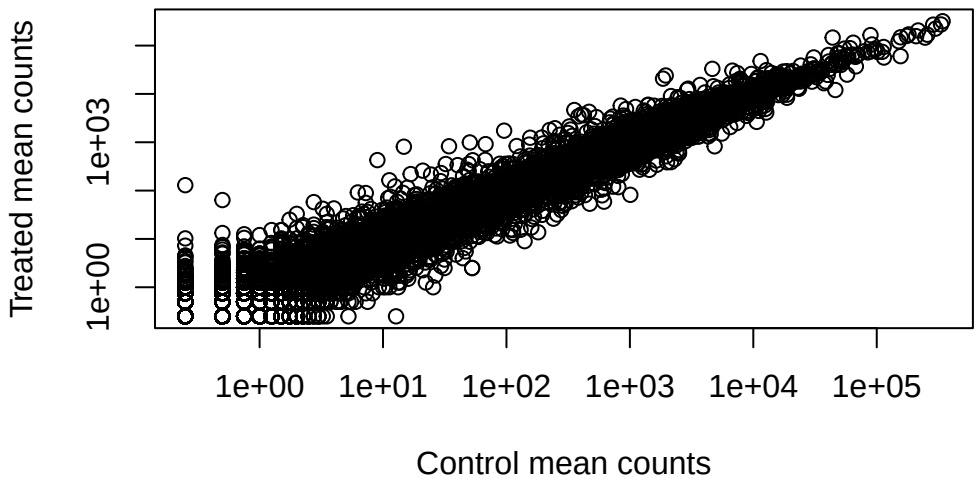
```
[1] -1
```

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

```
# log="xy" applies log scale to both x and y axes
plot(meancounts$control, meancounts$treated,
     xlab = "Control mean counts",
     ylab = "Treated mean counts",
     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
```



Let's calculate the log2 fold change for our treated over our control mean counts.

```
meancounts$log2fc <- log2(meancounts[,"treated"]/meancounts[,"control"])
```

```
head(meancounts)
```

```
                 control treated      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
```

A common "rule of thumb" is a log2 fold change cutoff of +2 and -2 to call genes "up regulated" or "down regulated"

Number of "up" genes

```
sum(meancounts$log2fc > +2, na.rm =T)
```

```
[1] 1846
```

Numbers of “down” genes

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

```
[1] 22928
```

```
zero.vals <- which(meancounts[,1:2]==0, arr.ind=TRUE)

to.rm <- unique(zero.vals[,1])
mycounts <- meancounts[-to.rm,]
head(mycounts)
```

```
                control treated      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
```

> Q7. What is the purpose of the arr.ind argument in the which() function call above? Why would we then take the first column of the output and need to call the unique() function?

arr.ind=TRUE makes which() return row and column indices of zeros. We take the first column to get the gene rows, and unique() ensures each gene is only counted once before removing them.

```
up.ind <- mycounts$log2fc > 2
down.ind <- mycounts$log2fc < (-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?

```
sum(up.ind, na.rm = TRUE)
```

```
[1] 250
```

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

```
sum(down.ind, na.rm = TRUE)
```

```
[1] 367
```

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

No not really because these numbers are just a rough exploratory estimate.

DESeq Analysis

Let's do this analysis properly and keep your inner stats nerd happy.

```
library(DESeq2)
```

```
Warning: package 'matrixStats' was built under R version 4.5.2
```

For DESeq analysis we need 3 things: 1. count values 2. metadata telling us about the columns in `countdata` (`colData`) 3. design of experiment

Our first function form DESeq2 will setup the input require for analysis by storing al these 3 things together.

```
dds <- 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
```

The main function of DESeq2 that runs the analysis is called `DESeq()`

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

```
estimating size factors
```

```
estimating dispersions
```

```
gene-wise dispersion estimates

mean-dispersion relationship

final dispersion estimates

fitting model and testing
```

```
results(dds)
```

```
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
```

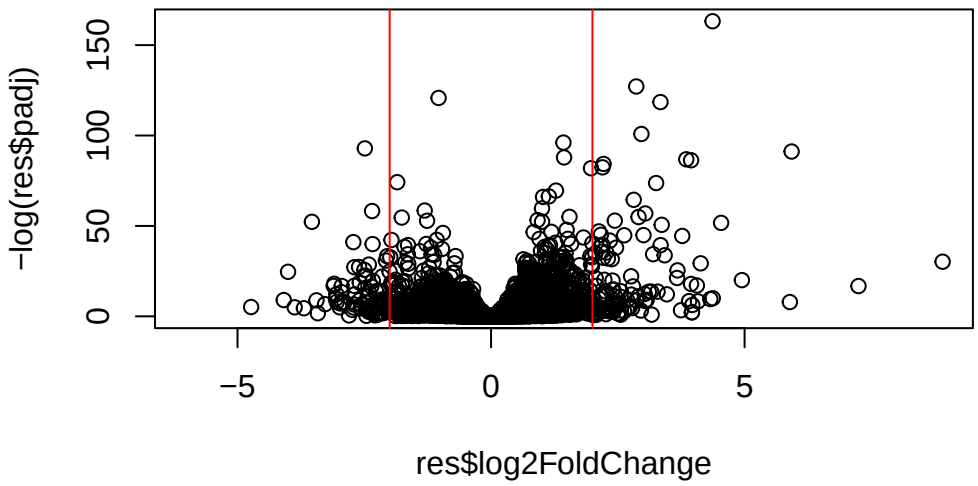
```
res <- results(dds)
head(res)
```

```
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
```

Volcano Plot

This is common summary result figure from these types of experiments and plot the log2 fold-change vs. the adjusted p-value.

```
plot(res$log2FoldChange, -log(res$padj))
abline(v=c(-2,2),col="red")
```



Save our Results

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