# Class11: RNA-Seq continued

Sam Altshuler (PID: A59010373)

2/23/2022

## Transcriptomics and Analysis of RNA-Seq Data

Today we will run differential expression analysis of published data from Himes et al.

### Import the countData and colData

```
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
```

There are 38694 rows, i.e. "genes" in this dataset. There are 8 columns in this dataset, i.e. experiments in the dataset.

```
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
```

The rows in the metadata set corresponds to the experiments being run (the columns in the counts dataset). There are 4 controls and 4 treated experiments.

The next question is does the drug do anything?

First confirm that the metadata matches the counts data.

```
#column names of counts compared to ID column of metadata
all(colnames(counts) == metadata$id)
```

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

All of the data names match up!

Gather all of the control data (extract from metadata).

```
#Store the IDs of the control experiments
control <- metadata[metadata$dex == "control", "id"]
# Pull the columns corresponding to the controls from the counts dataset
ct_control <- counts[,control]
```

Gather all of the treated data. This is the same as above but for the treated columns.

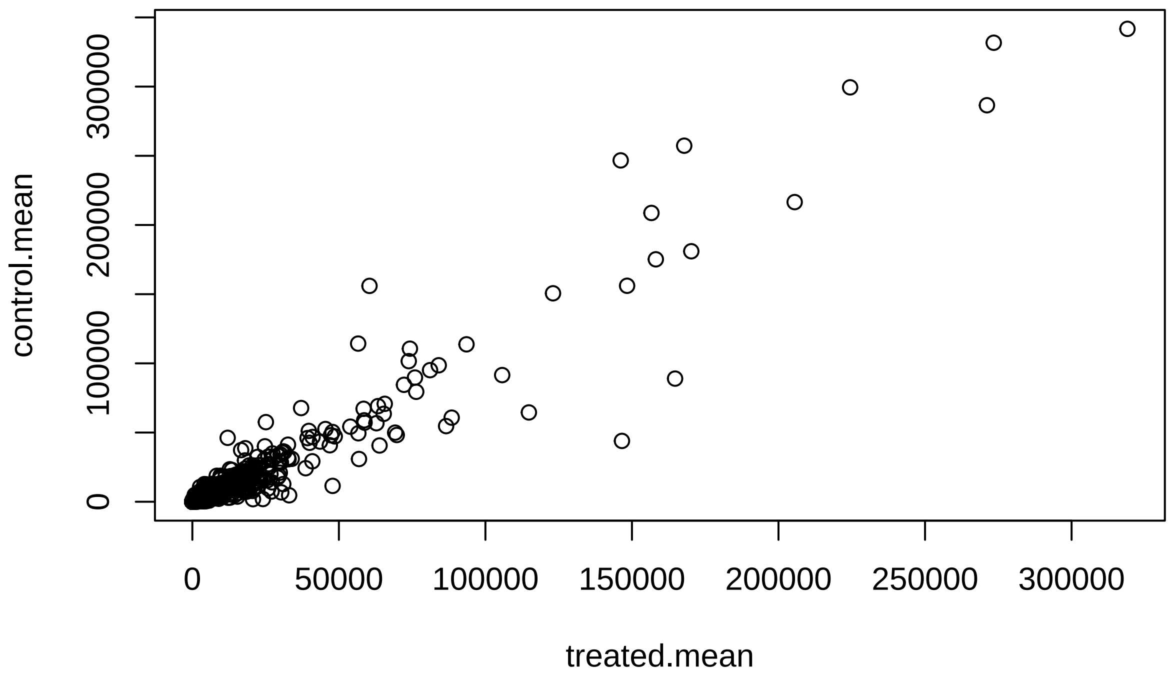
```
treat <- metadata[metadata$dex == "treated", "id"]
ct_treat <- counts[,treat]
```

Get a mean gene expression per gene for both the control and the treated. Use `apply()` or `rowMeans()`.

```
# using apply: apply(ct_control, 1, mean)
control.mean <- rowMeans(ct_control)
treated.mean <- rowMeans(ct_treat)
```

Compare the two experimental conditions in a plot.

```
plot(treated.mean, control.mean)
```

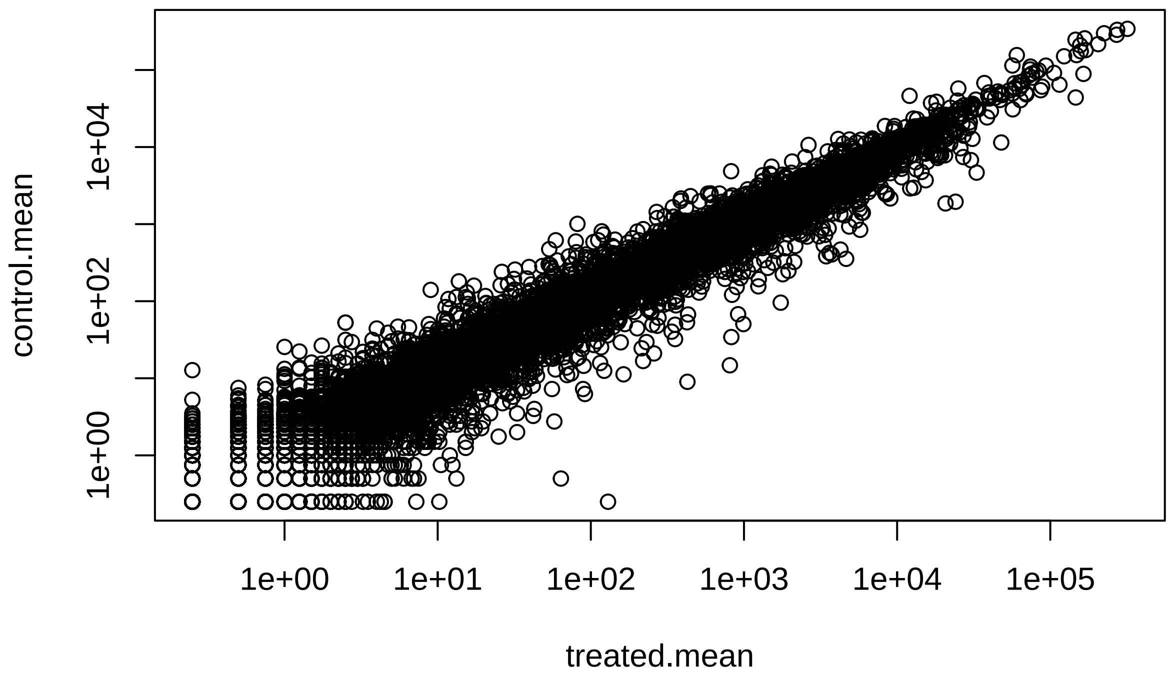


There are a bunch of genes with low values that overlap, making it hard to ID individual genes. The data is very skewed. The solution is to transform the data (like a log transformation) to make it more readable especially around overlapping values.

```
plot(treated.mean, control.mean, log = "yx")
```

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

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



We need to get rid of zeros because you can't take a log of 0.

We often use log 2 transformation because it has an easier to understand output. A log 2 value of zero means that there's been no change (lies on the straight line). A value of 1 means it's doubled from treatment compared to control and -1 means it's half. This is called the fold change (how much is it doubling)

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

Make a dataframe to store the results

```
ct_mean <- data.frame(control.mean, treated.mean, log2fc)
head(ct_mean)
```

```
##                 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
```

If either the control or treated have values of zero, there won't be a valuable log2fc value. If the denominator is a zero, the answer will be `NaN` (not a number), and if it's in the numerator, the answer will be infinity.

Try to find and filter out the zero values.

```
# Choose all count values for both control and treated with a value of zero
# make sure to return array indices where in the dataframe the zero values are
# Save the rows that correspond to the zero values
zip <- unique(which(ct_mean[,1:2] == 0, arr.ind = TRUE)[,"row"])
# Remove the rows that correspond to zero values from the dataframe
ct_mean_2 <- ct_mean[-zip,]
head(ct_mean_2)
```

```
##                 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
```

There are 21817 genes left after removing the zero values.

There are 250 genes that have a log2fc more than +2 (upregulated).

```
sum(ct_mean_2$log2fc > 2)
```

```
## [1] 250
```

These log2fc may not actually be statistically significant. Time to use the DESeq2 package!

## DESeq2

```
library(DESeq2)
```

```
## Loading required package: S4Vectors

## Loading required package: stats4

## Loading required package: BiocGenerics

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min
```

```
##
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: GenomicRanges

## Loading required package: GenomeInfoDb

## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

##
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## Loading required package: Biobase
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

First we need to set up the DESeq data object.

```
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
```

```
dds
```

```
## class: DESeqDataSet
## dim: 38694 8
## metadata(1): version
## assays(1): counts
## rownames(38694): ENSG00000000003 ENSG00000000005 ... ENSG00000283120
##   ENSG00000283123
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(4): id dex celltype geo_id
```

```
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
```

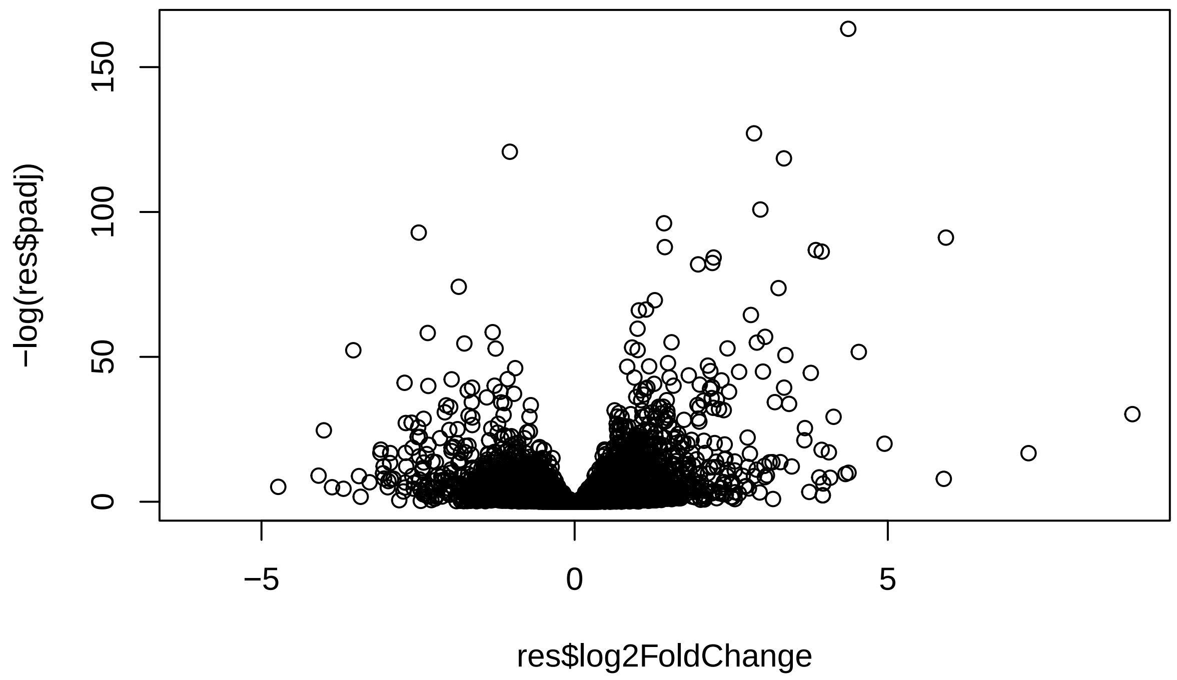
```
## log2 fold change (MLE): dex treated vs control
## Wald test p-value: dex treated vs control
## DataFrame with 38694 rows and 6 columns
##                  baseMean log2FoldChange     lfcSE      stat    pvalue
##                 <numeric>      <numeric> <numeric> <numeric> <numeric>
## ENSG00000000003  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
```

`padj` is the adjusted p-value for multiple testing.

## A main results figure

A common main results figur is a volcano plot. This is a plot of the log2 fold change on the x axis v the p-value (or padj) on the y-axis.

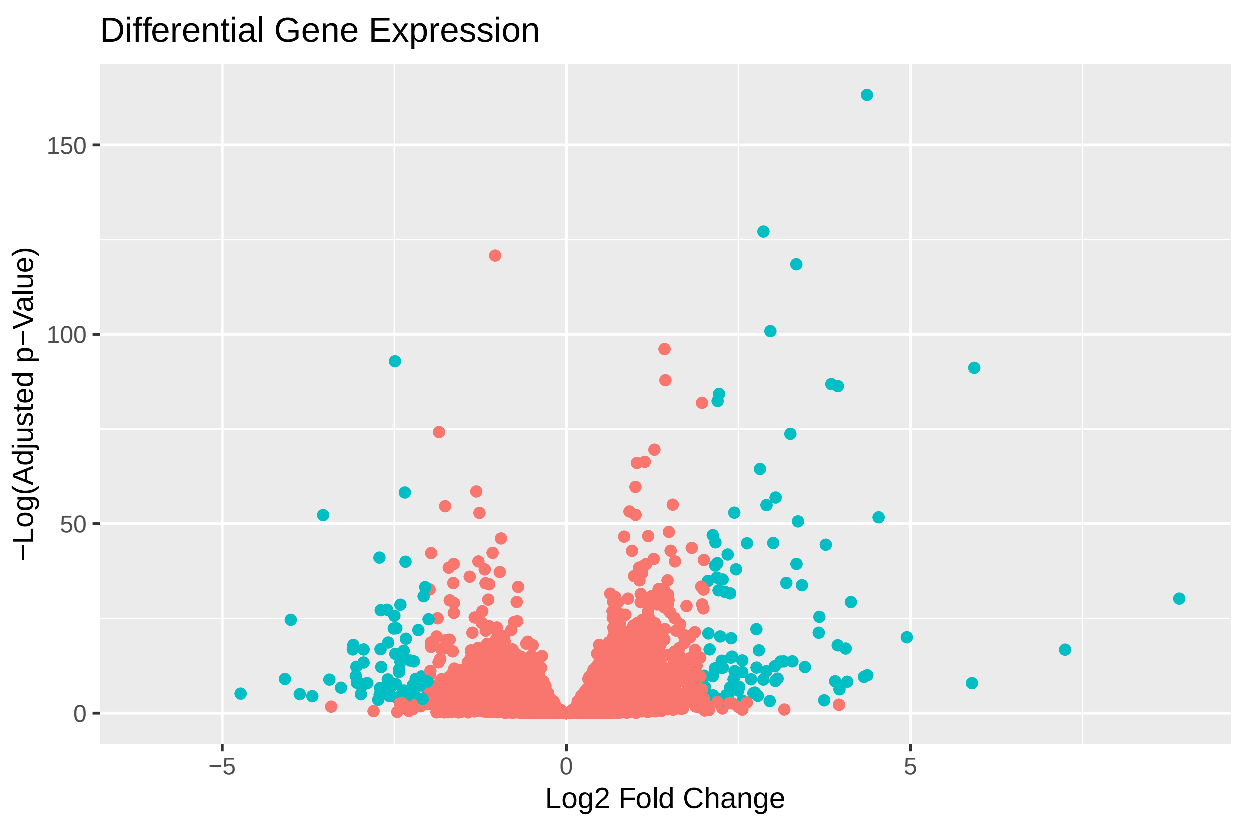
```r
plot(res$log2FoldChange, -log(res$padj))
```



As it goes up the y-axis, the smaller the p-value, the less likely the fold change is due to random chance (i.e. false positivess). The plot function should be `plot(foldchange, -log(p-value))`.

```r
library(ggplot2)
# make the same volcano plot as above but color by if the p value is less than 0.05 AND
# the log2 fold change is greater than 2 or less than -2 (absolute value is greater than 2)
ggplot(as.data.frame(res))+
  aes(x = log2FoldChange, y = -log(padj), color = padj < 0.05 & abs(log2FoldChange) > 2)+
  geom_point()+
  xlab("Log2 Fold Change") +
  ylab("-Log(Adjusted p-Value)")+
  labs(title = "Differential Gene Expression", caption = "Data from Himes et al.")+
  theme(legend.position = "none")
```

```
## Warning: Removed 23549 rows containing missing values (geom_point).
```



Data from Himes et al.