# Lab 12

Adam Bisharat

```
library(BiocManager)
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, aperm, 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, saveRDS, setdiff,
    table, tapply, union, unique, unsplit, which.max, which.min
```

```

Attaching package: 'S4Vectors'
```

```
The following object is masked from 'package:utils':

    findMatches

The following objects are masked from 'package:base':

    expand.grid, I, unname

Loading required package: IRanges

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
```

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

nrow(counts)
```

```
[1] 38694
```

```
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
```

```
View(metadata)
```

Q1. How many genes are in this dataset?

There are 38694 genes

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

```
[1] 4
```

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

```
control treated
      4       4
```

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

There are 4 control cell lines

### Toy differential expression analysis

Calculate the mean per gene count values for all "control" samples (i.e columns in `counts`) and do the same for "treated" and then compare them.

1. Find all "control values/columns in `counts`

```
table(metadata$dex == "control")
```

```
FALSE  TRUE
    4     4
```

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]
```

2. Find the mean per gene across all control columns.

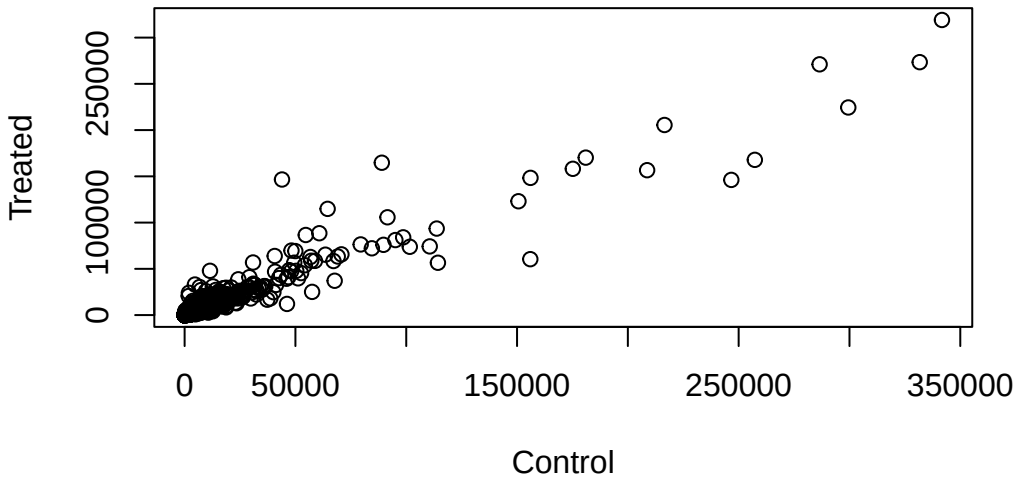
```
control.mean <- apply(control.counts, 1, mean)
```

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 <- apply(treated.counts, 1, mean)
```

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.

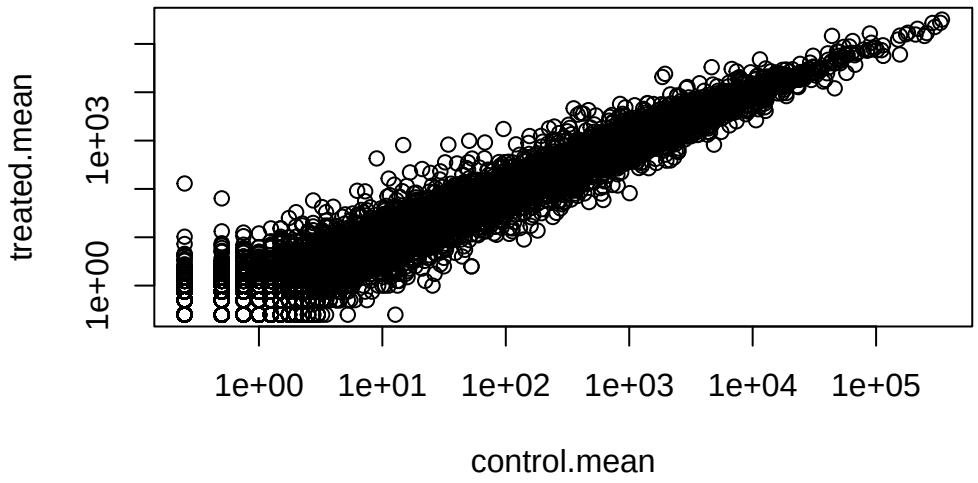
```
meancounts <- data.frame(control.mean, treated.mean)
plot(meancounts[,1],meancounts[,2], xlab="Control", ylab="Treated")
```




```
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
```

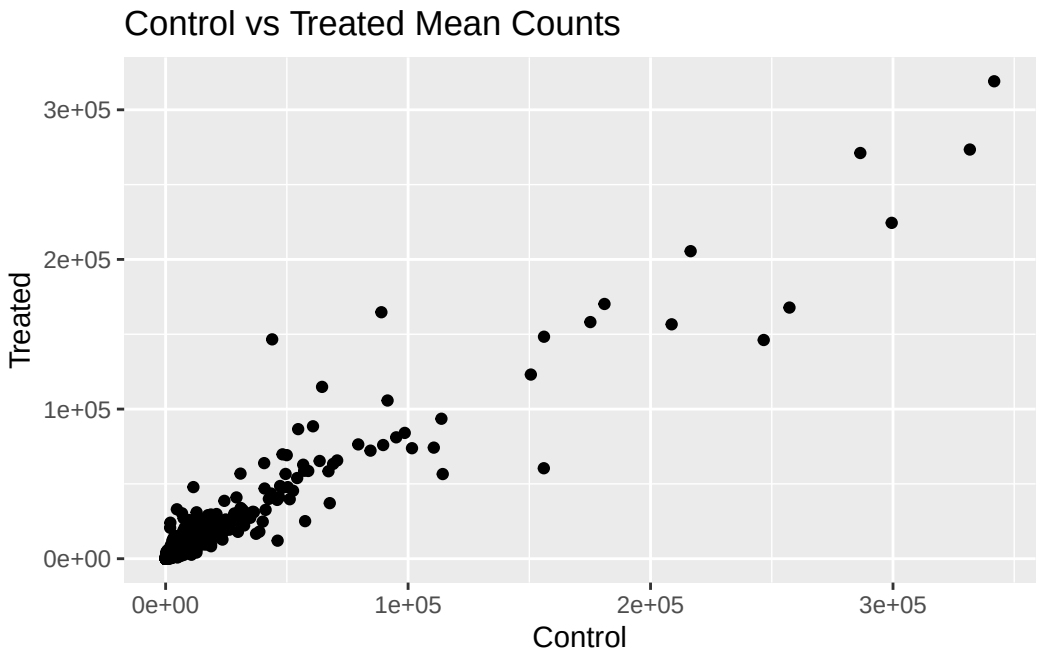



Q5 (b).You could also use the ggplot2 package to make this figure producing the plot below. What geom_?() function would you use for this plot?

point

```
library(ggplot2)

ggplot(meancounts, aes(x = control.mean, y = treated.mean)) +
  geom_point() +
  xlab("Control") +
  ylab("Treated") +
  ggtitle("Control vs Treated Mean Counts")
```




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

plot(meancounts, `log='xy'`)

We most frequently use log2 transformations for this type of data

```
log2(10/10)
```

```
[1] 0
```

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

```
[1] 1
```

```
log2(30/10)
```

```
[1] 1.584963
```

```
log2(40/10)
```

```
[1] 2
```

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

```
[1] -1
```

These log2 values make the interpretation of "fold-change" a little easier and a rule-of-thumb in the field is a log2 fold-change of +2 or -2 is where we start to pay attention

```
log2(40/10)
```

```
[1] 2
```

Lets calculate the log2 (fold-change) and add it to our `meancounts` data.frame

```
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
```

> 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?

```
to.rm <- rowSums((meancounts[,1:2]==0) > 0)
mycounts <- meancounts[!to.rm,]
```

> Q. how many genes do I have left after this zero count filtering

```
nrow(mycounts)
```

```
[1] 21817
```

> Q. How many genes are "up" regulated upon drug treatment with a threshold of +2 log2-fold-change?

```
up.ind <- mycounts$log2fc > 2
count(up.ind)
```

```
[1] 250
```

250 up-regulated genes

> Q. How many genes are "down" regulated upon drug treatment with a threshold of -2 log2-fold-change?

```
down.ind <- mycounts$log2fc < (-2)
count(down.ind)
```

```
[1] 367
```

367 down regulated genes