class13 rna seq (pt.1)

Today we will analyze data from a published rna seq experiment where airway smooth muscle cells were treated with dexamethsone, a synthetic gluticosteroid steroid with anti inflmmatory effects (Himes et al).

Import countDara and colData

There are two databases I need to import / read

-'count data' the trascript counts per gene (rows) in the different experiments

-col'Data' information about the columns (ie experiments) in 'countData'. 'countData'.

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

We can see '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
```

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
```

```
metadata$dex == "control"
```

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

```
control <- metadata[metadata[,"dex"]=="control",]
control.counts <- counts[ ,control$id]
control.mean <- rowSums( control.counts )/4
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?

```
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
```

Q4. Follow the same procedure for the treated samples

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

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

```
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
```

We can find the average count values per gene for all control experiments and compare it to the mean values for treated.

-Extract all “control” columns from the ‘counts’ data -Find the mean value for each gene in these columns

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

```
dim(control.counts)
```

```
[1] 38694     4
```

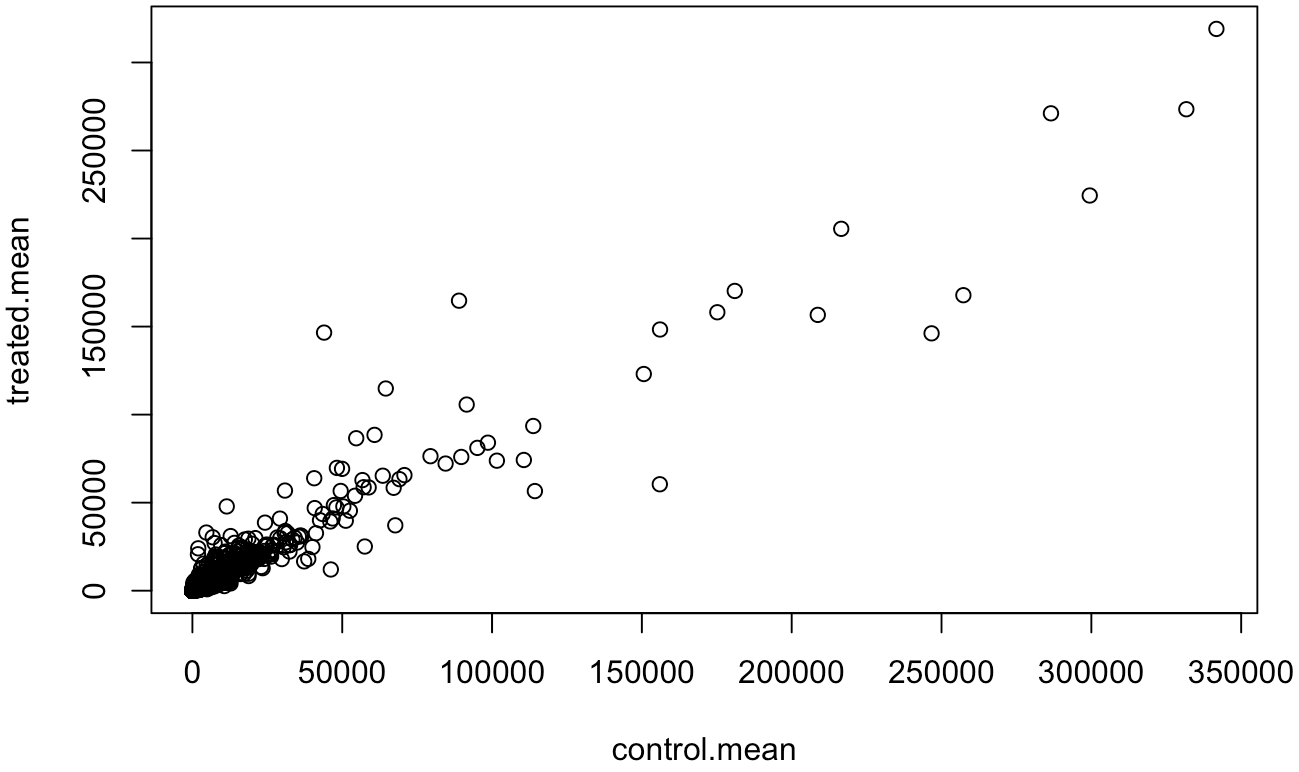
Lets put these two mean values together for easy book keeping

```
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
```

Lets have a look. Plot control.mean vs treated. mean

```
plot(meancounts)
```

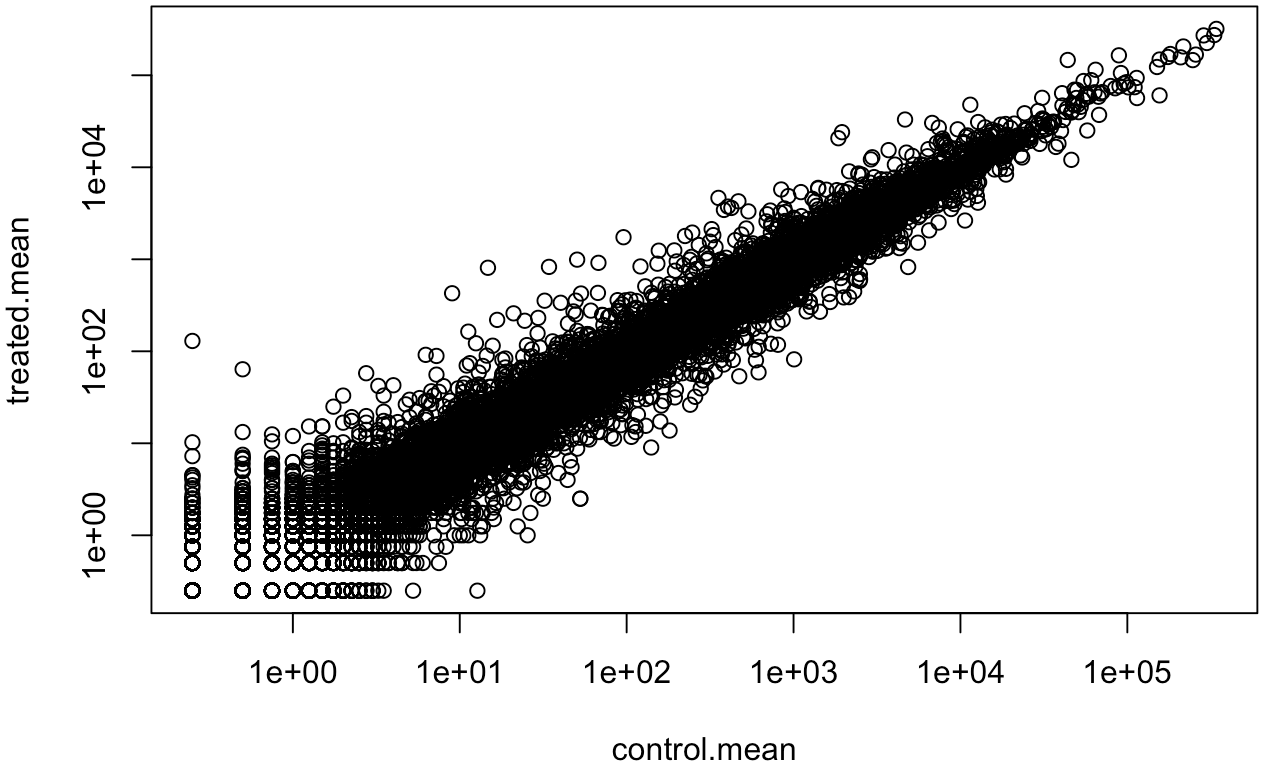


Whenver we see data that is so heavily skewed like this we often log transform it so we can see what is going on more easily.

```
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.

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

```
[1] 0
```

```
log2(20/40)
```

```
[1] -1
```

```
log2(20/40)
```

```
[1] -1
```

We can now add “log2 fold change” values to our ‘meancounts’ dataset.

```
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 genes - remove rows (genes) that have 0 value in either control or treated means.

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

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

```
[1] 1910
```

How many genes are “down” regulated at threshold -2?

```
down.inds <- meancounts$log2fc <= -2
```

DESeq2 analysis

Consider the significance of differences not just their magnitude

```
#/ message: false
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
```

To use this package it wants countData and colData in a specific format.

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

```
estimating size factors

estimating dispersions

gene-wise dispersion estimates

mean-dispersion relationship

final dispersion estimates

fitting model and testing
```

Extract my results

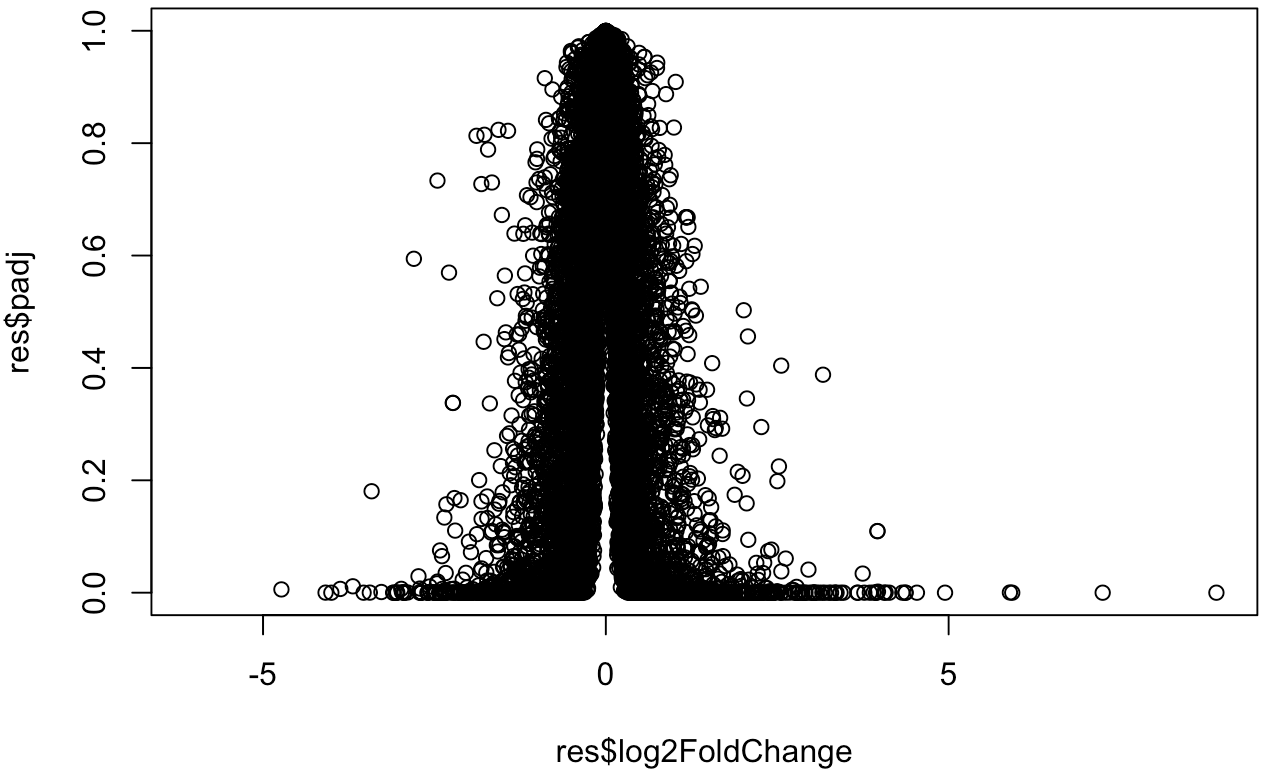
```
res <- results (dds)
head(res)
```

```
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
```

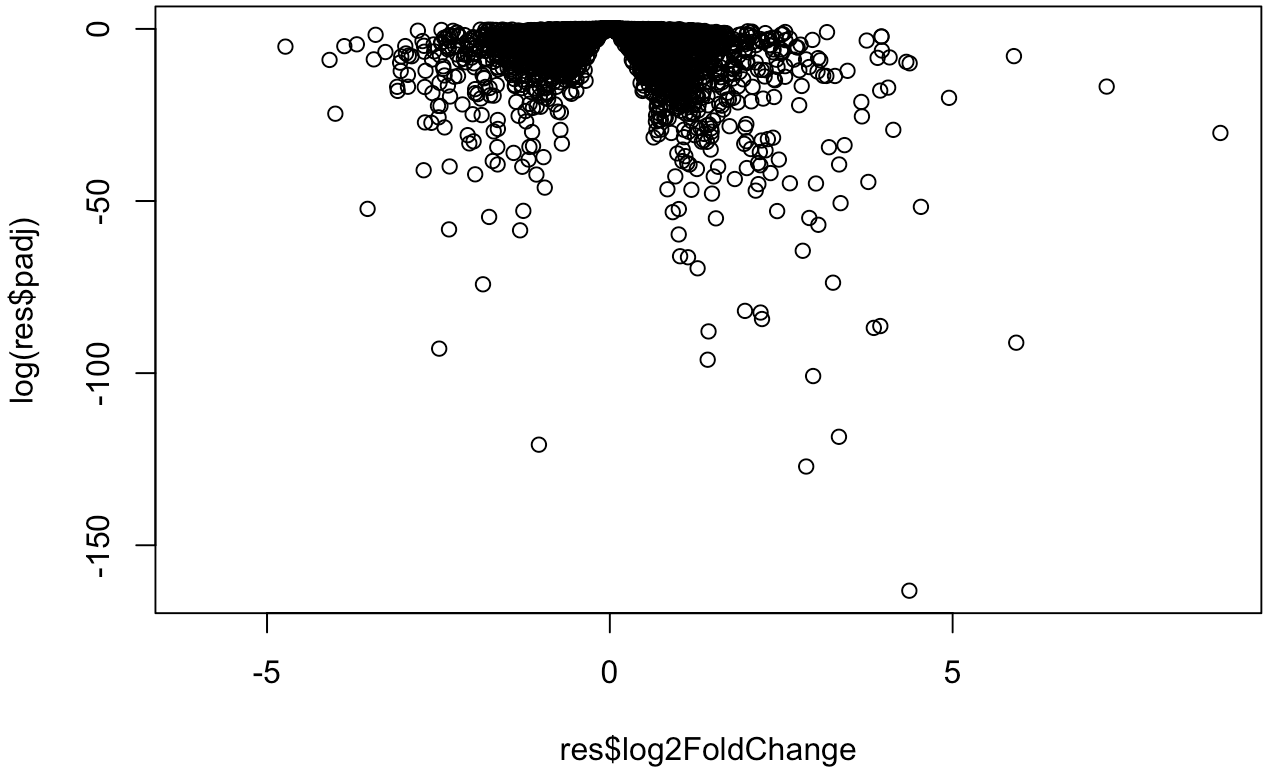
Plot of fold change vs p value

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



Take the log of p value

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

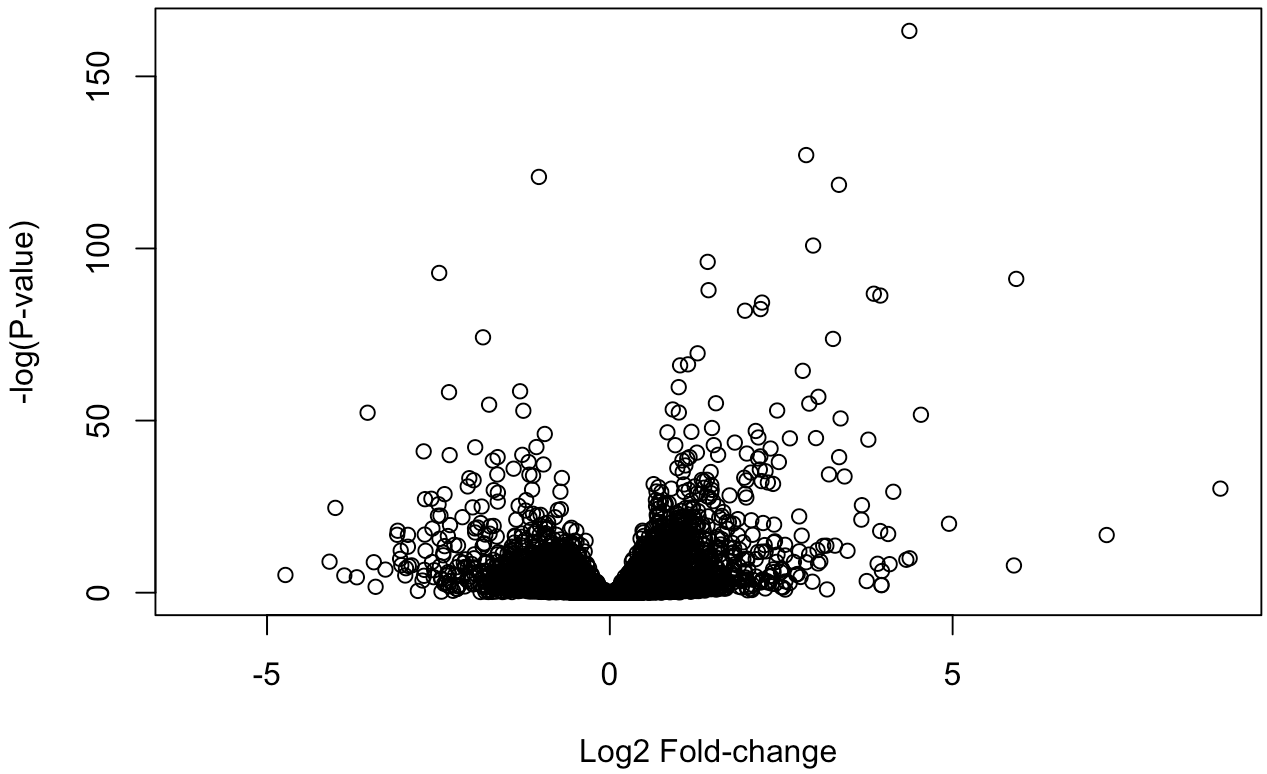


Take the log of p value

```
log(0.00000000001)
```

```
[1] -25.32844
```

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



Lets save our work to date

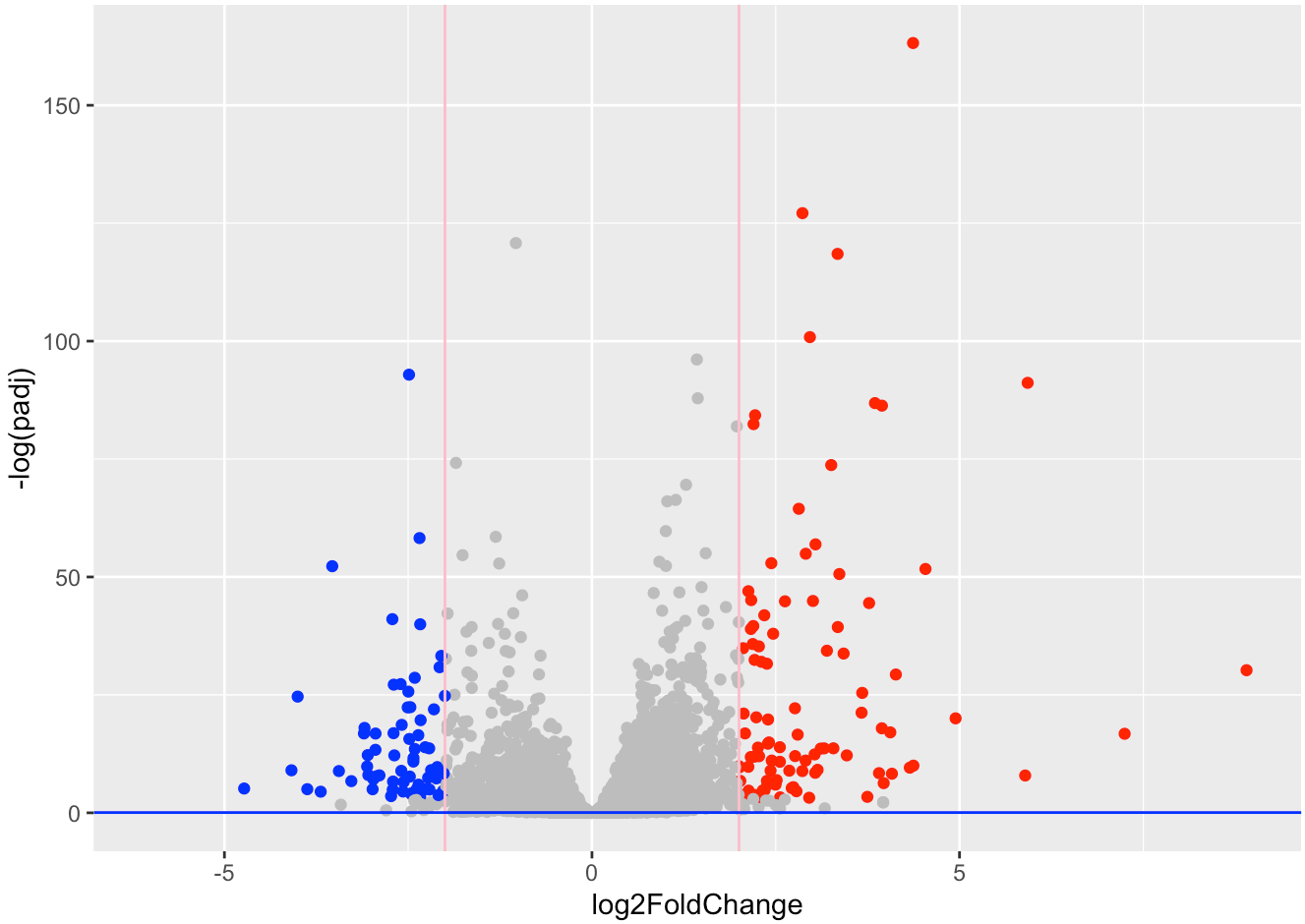
```
write.csv(res, file="myresults.csv")
```

To finish off lets make a nicer volcano plot. Add the log2 threshold of +2/-2. -Add P-value threshold lines at 0.05. -Add color to highlight subset of genes that meet both if the above thresholds. With ggplot.

```
mycols <-rep("grey", nrow(res))
mycols[res$log2FoldChange >= 2] <- "red"
mycols[res$log2FoldChange <= -2] <- "blue"
mycols[res$padj > 0.05] <- "grey"
```

```
library(ggplot2)
ggplot(res)+
  aes(log2FoldChange, -log(padj)) +
  geom_point(col=mycols)+
  geom_vline(xintercept =c(-2,2), col="pink" )+
  geom_hline(yintercept =0.05, col="blue" )
```

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


150
100
50
0
-log(padj)
-5
0
5
log2FoldChange