# Class13Lab

Xinyu Wen (A17115443)

## Table of contents



## 2. Bioconductor setup

In console: install.packages("BiocManager") BiocManager::install() BiocManager::install("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, 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
```

Today we will analyze data from a published RNA-seq experiment where airway sommoth muscle cells were treated with dexamethasone, a synthetic glucocorticoid steroid with anti-inflammatory effects (Himes et al. 2014).

## 3. Import countData and colData

There are two datasets I need to import/read

- `countData` The transcript counts per gene (row) in the different experiments
- `colData` Information about the columns (ie. experiments) in `countData`

```
counts <- read.csv("airway_scaledcounts.csv", row.name = 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
```

```
head(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
```

Q1. How many genes are there in this dataset?

```
nrow(counts)
```

```
[1] 38694
```

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

```
metadata$dex
```

```
[1] "control" "treated" "control" "treated" "control" "treated" "control"
[8] "treated"
```

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

```
control treated
      4       4
```

or

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

```
[1] 4
```

## 4. Toy differential gene expression

We can find the average (aka. mean) count values per gene for all "control" experiments vs "treated". Exact all "control" columns from the `counts` data.

```
control.inds <- metadata$dex == "control" #controls are the columns in `counts` dataset
control.counts <- counts[ ,control.inds]
dim(control.counts)
```

```
[1] 38694     4
```

Find the mean value for each gene (row).

```
control.mean <- rowSums(control.counts)/ncol(control.counts) #divided by "ncol()" to make the
#remember to code ", row.name = 1" when importing the dataset
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?

Divided by "ncol(control.counts)" to make the code more robust

> 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]
dim(control.counts)
```

```
[1] 38694     4
```

Find the mean value for each gene (row).

```
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
```

### Plot "control.mean" vs "treated.mean"

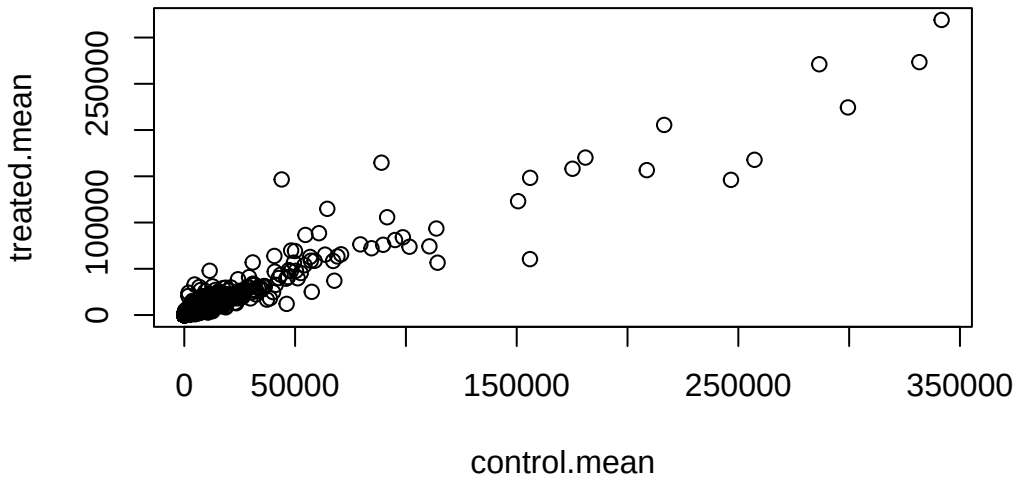
Put them together to make it easier for book-keeping

```
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
```

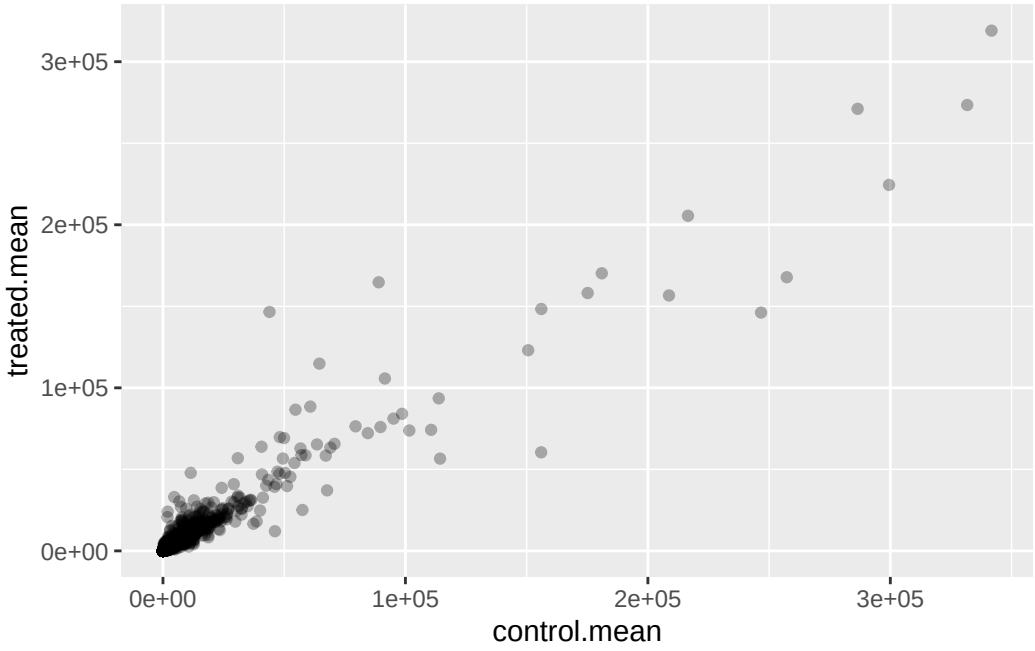
Q5 (a). Create a scatter plot showing the mean of the treated samples against the mean of the control samples. Your plot should look something like the following.

```
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?

```
library(ggplot2)
meancounts.ggplot <- ggplot(meancounts) +
  aes(control.mean, treated.mean)+
  geom_point(alpha=0.3)
meancounts.ggplot
```




If the dots are all in the central diagonal line, it means the gene expression for treated and controls are the same. i.e. The drug did not have an effect on the gene expression of the treated group. - If the dots are above diagonal line, it means the treated gene expression is higher.

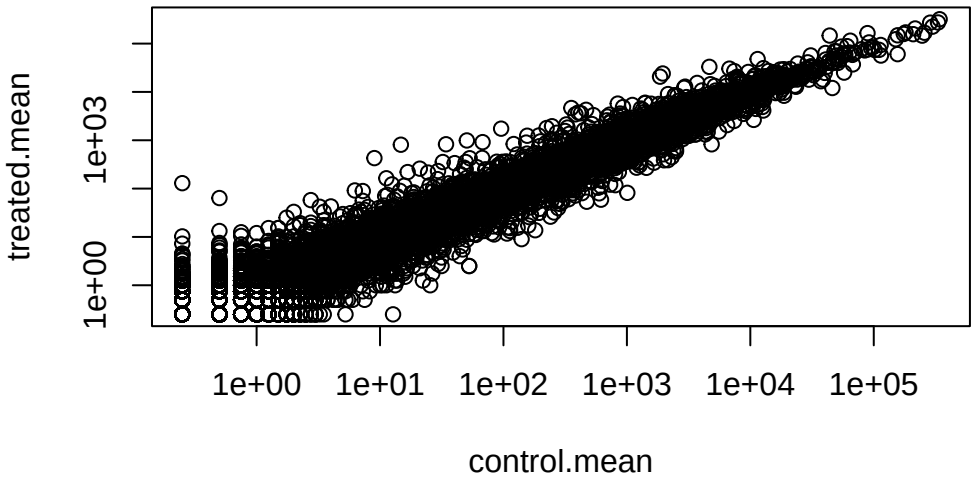
> Q6. Try plotting both axes on a log scale. What is the argument to plot() that allows you to do this?

A lot of the genes have low expression, so they are all overlapped in the lower left corner. Confirmed by setting ggplot geom to alpha = 0.3. It is highly skewed. Thus we want to use log to transform the plot.

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

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

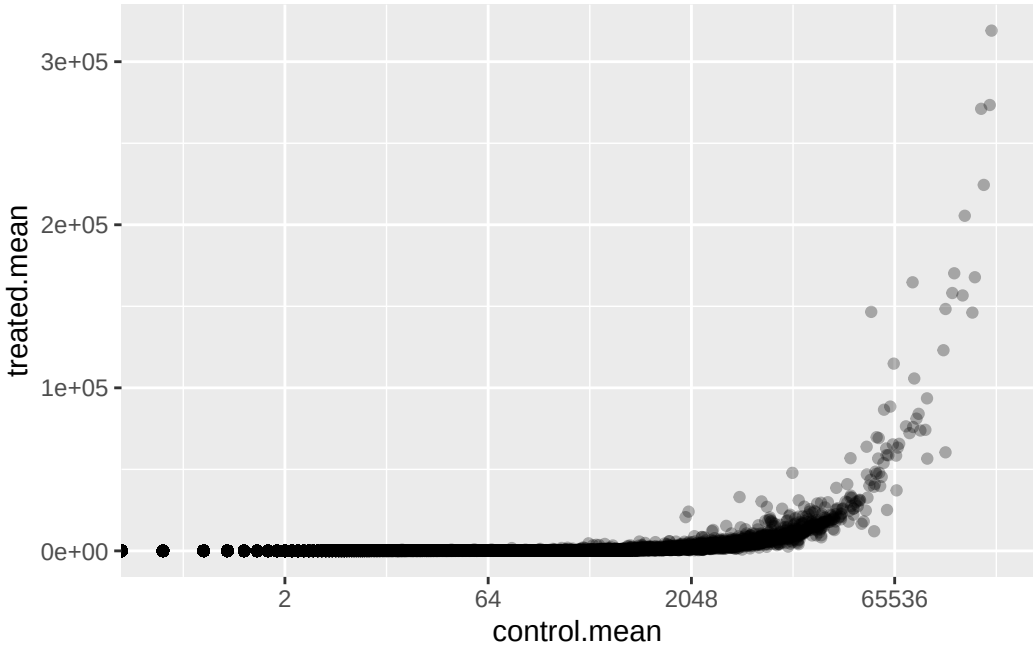
```
Warning in xy.coords(x, y, xlabel, ylabel, log): 15281 y values <= 0 omitted
from logarithmic plot
```




or

```
meancounts.ggplot + scale_x_continuous(trans="log2")
```

```
Warning in scale_x_continuous(trans = "log2"): log-2 transformation introduced
infinite values.
```




Fold change:

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

```
[1] 0
```

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

```
[1] 1
```

```
# treated / control
log2(80/20)
```

```
[1] 2
```

Positive value means we have more in the treated.

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

```
[1] -1
```

Negative value means it went to the control direction/ below the diagonal line

Let's 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 all the 0.00 count genes (i.e. remove the rows/genes that have a 0.00 value in either control or treated means) to eliminate NaN(not a number) and -Inf from log2fc.

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

The arr.ind argument calls the number of rows that have TRUE (non-zero) in both column 1 and 2. unique() extracts the zero values in column 1 without extacting the duplicated ID twice.

> Q8. How many genes are "up" regulated at the common log2 fold-change threshold of +2?

```
zero.vals <- which(meancounts[,1:2]==0, arr.ind = T)
to.rm <- unique(zero.vals[,1])
mycounts <- meancounts[-to.rm,]

up.inds <- mycounts$log2fc > 2
sum(up.inds)
```

```
[1] 250
```

Q9. How many genes are "down" regulated at the common log2 fold-change threshold of -2?

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

down.inds <- mycounts$log2fc < -2
sum(down.inds)
```

```
[1] 367
```

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

We can use p-value to determine whether the difference is significant. (#5)

## 5. DESeq2 analysis

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

```
#/ message: false
library(DESeq2)
```

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
```

Skipped 6

## 7. DESeq analysis

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
```

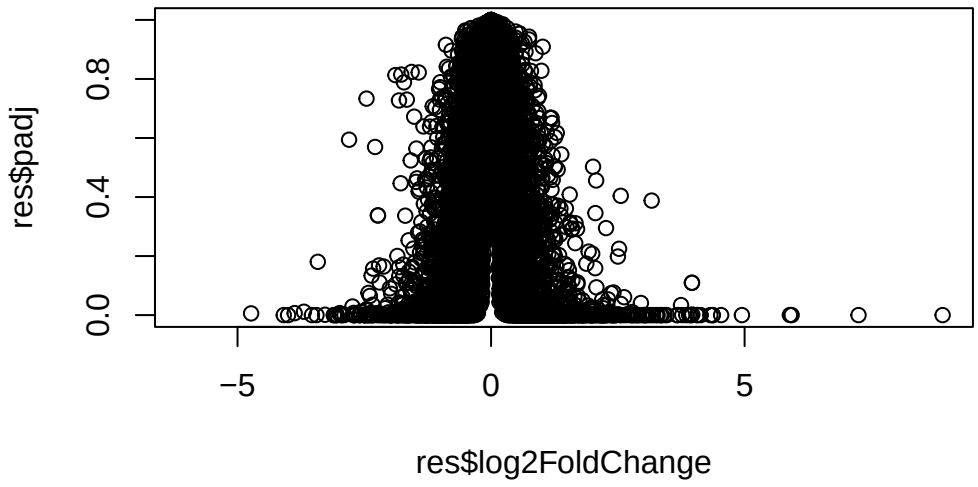
padj means adjusted p value. More strict.

Skipped 8

## 9. Data Visualization

Plot a fold-change vs p-value (adjusted for multiple testing)

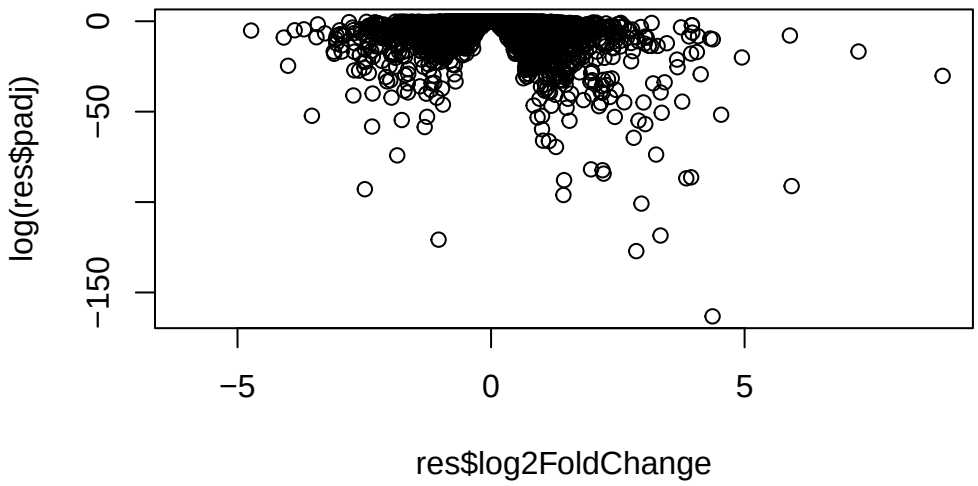
```
plot(res$log2FoldChange, res$padj)
```




Very skewed. We only care about the dots with low p-value

Take log of the p-values:

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




```
log(0.01)
```

```
[1] -4.60517
```

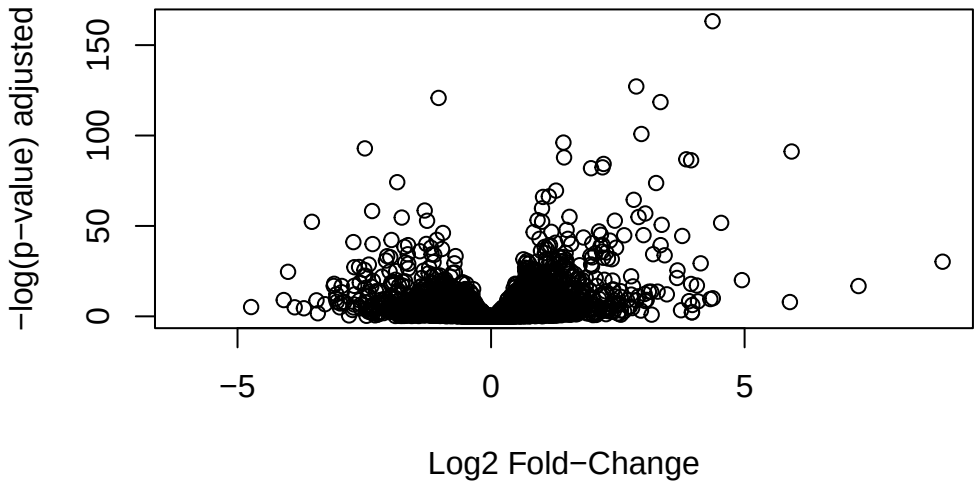
```
log(0.0000000001)
```

```
[1] -23.02585
```

Smaller number in log gives more negative values

Flip the axis by adding a - in front of y value, making it easier to read (we are used to looking up a plot):

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




The "Volcano Plot"

Let's save our work to date

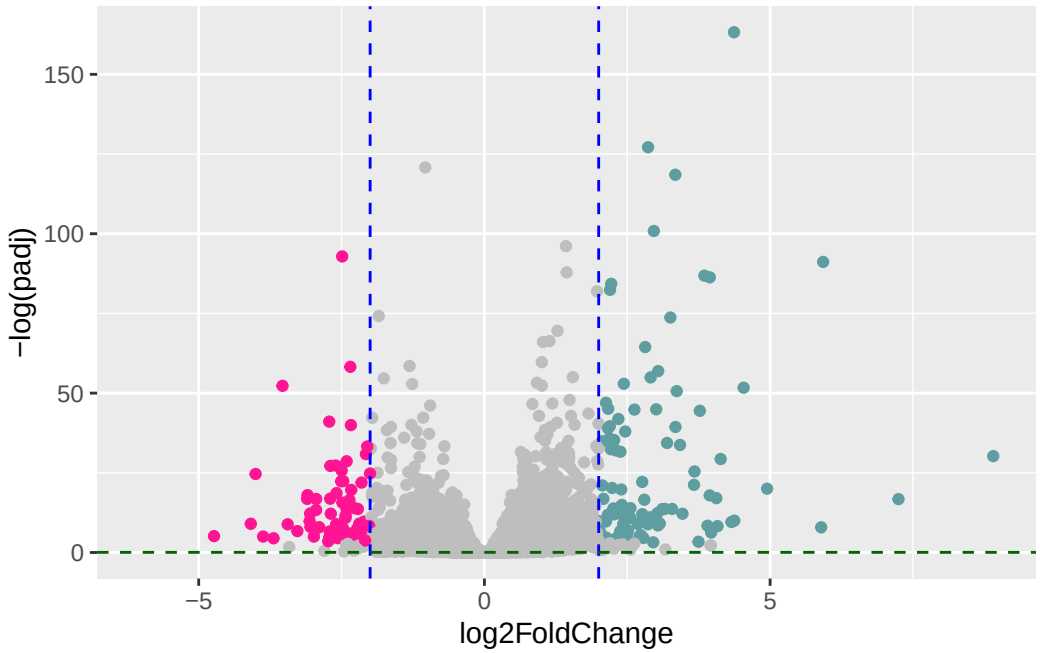
```
write.csv(res, file="MyResultsClass13.1Lab.csv")

res <- read.csv("MyResultsClass13.1Lab.csv", stringsAsFactors = F, header = T, row.names = 1)
```

To finish off, let's make a nicer volcano plot. - Add the log2 threshold lines at +2/-2 - Add p-value threshold lines at 0.05 - Add the color to highlight the subset

```
mycol <- rep("grey", nrow(res))
mycol[res$log2FoldChange >= 2] <- "cadetblue"
mycol[res$log2FoldChange <= -2] <- "deeppink"
mycol[res$padj > 0.05] <- "grey"
```

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

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




2/20/25

## 8. Adding annotation data

Now the question is what are the colored points in the above volcano plot. i.e. What are the genes most influenced by drugs here?

We will use some BioConductor packages to "map" the ENSEMBLE ids to more useful gene SYMBOL names/ids.

We can install these packages with: `BiocManager::install("AnnotationDbi")`

```
library(AnnotationDbi)
library(org.Hs.eg.db)
```

What databased identifiers can I translate between here:

```
columns(org.Hs.eg.db)
```

```
 [1] "ACCNUM"       "ALIAS"        "ENSEMBL"      "ENSEMBLPROT"  "ENSEMBLTRANS"
 [6] "ENTREZID"     "ENZYME"       "EVIDENCE"     "EVIDENCEALL"  "GENENAME"
[11] "GENETYPE"     "GO"           "GOALL"        "IPI"          "MAP"
[16] "OMIM"         "ONTOLOGY"     "ONTOLOGYALL"  "PATH"         "PFAM"
[21] "PMID"         "PROSITE"      "REFSEQ"       "SYMBOL"       "UCSCKG"
[26] "UNIPROT"
```

We can now use the `mapIDs()` function to translate/map between these different identifier formats.

Let's add SYMBOL, GNENNAME, and ENTREZID.

```
res$symbol <- mapIds(org.Hs.eg.db,
                     keys=rownames(res),
                      keytype = "ENSEMBL",
                     column = "SYMBOL")
```

```
'select()' returned 1:many mapping between keys and columns
```

```
res$genename <- mapIds(org.Hs.eg.db,
                     keys=rownames(res),
                      keytype = "ENSEMBL",
                     column = "GENENAME")
```

```
'select()' returned 1:many mapping between keys and columns
```

```
res$entrez <- mapIds(org.Hs.eg.db,
                     keys=rownames(res),
                      keytype = "ENSEMBL",
                     column = "ENTREZID")
```

```
'select()' returned 1:many mapping between keys and columns
```

## 10. Pathway analysis

Now I know the gene names and their IDs in different dtabases, I want to know what types of biology they are involved in ...

This is the job of "pathway analysis" (aka. "gene set enrichment")

There are tones of different BioConductor packages for pathway analysis. Here we use just one of them called **gage**, and **pathview** (for figures). Install them with `BiocManager::install(c("pathview", "gage", "gageData"))`

```
library(gage)
```

```
library(gageData)
library(pathview)
```

```
##############################################################################
Pathview is an open source software package distributed under GNU General
Public License version 3 (GPLv3). Details of GPLv3 is available at
http://www.gnu.org/licenses/gpl-3.0.html. Particullary, users are required to
formally cite the original Pathview paper (not just mention it) in publications
or products. For details, do citation("pathview") within R.

The pathview downloads and uses KEGG data. Non-academic uses may require a KEGG
license agreement (details at http://www.kegg.jp/kegg/legal.html).
##############################################################################
```

Load up the KEGG genesets

```
data(kegg.sets.hs)
```

```
head(kegg.sets.hs, 2)
```

```
$`hsa00232 Caffeine metabolism`
[1] "10"   "1544" "1548" "1549" "1553" "7498" "9"

$`hsa00983 Drug metabolism - other enzymes`
 [1] "10"     "1066"   "10720"  "10941"  "151531" "1548"   "1549"   "1551"
```

```
 [9] "1553"   "1576"   "1577"   "1806"   "1807"   "1890"   "221223" "2990"
[17] "3251"   "3614"   "3615"   "3704"   "51733"  "54490"  "54575"  "54576"
[25] "54577"  "54578"  "54579"  "54600"  "54657"  "54658"  "54659"  "54963"
[33] "574537" "64816"  "7083"   "7084"   "7172"   "7363"   "7364"   "7365"
[41] "7366"   "7367"   "7371"   "7372"   "7378"   "7498"   "79799"  "83549"
[49] "8824"   "8833"   "9"      "978"
```

We will use these KEGG genesets (aka. pathways) and our `res` results to see what overlaps. To do this, we will use the `gage()` function.

For input `gage()` wants just a vector of importance - in our case FoldChange values.

```
foldchanges <- res$log2FoldChange
```

Vectors in R can have "names" that are useful for bookkeeping, so we know what a given value corresponds to. Let's put names on the `foldchanges` vector - here we will use `res$entrez`.

```
names(foldchanges) <- res$entrez
```

Now we can run "pathway analysis"

```
# Get the results
keggres = gage(foldchanges, gsets=kegg.sets.hs)
```

```
head(keggres$less)
```

```
                                                      p.geomean stat.mean
hsa05332 Graft-versus-host disease                   0.0004250461 -3.473346
hsa04940 Type I diabetes mellitus                    0.0017820293 -3.002352
hsa05310 Asthma                                      0.0020045888 -3.009050
hsa04672 Intestinal immune network for IgA production 0.0060434515 -2.560547
hsa05330 Allograft rejection                         0.0073678825 -2.501419
hsa04340 Hedgehog signaling pathway                  0.0133239547 -2.248547
                                                             p.val      q.val
hsa05332 Graft-versus-host disease                   0.0004250461 0.09053483
hsa04940 Type I diabetes mellitus                    0.0017820293 0.14232581
hsa05310 Asthma                                      0.0020045888 0.14232581
hsa04672 Intestinal immune network for IgA production 0.0060434515 0.31387180
hsa05330 Allograft rejection                         0.0073678825 0.31387180
hsa04340 Hedgehog signaling pathway                  0.0133239547 0.47300039
                                                     set.size         exp1
```

```
hsa05332 Graft-versus-host disease                                40 0.0004250461
hsa04940 Type I diabetes mellitus                                42 0.0017820293
hsa05310 Asthma                                                  29 0.0020045888
hsa04672 Intestinal immune network for IgA production            47 0.0060434515
hsa05330 Allograft rejection                                     36 0.0073678825
hsa04340 Hedgehog signaling pathway                              56 0.0133239547
```

hsa##### are the kegg identifiers

We can get a pathway image file with our genesets highlighted via the `pathview()` function.

```
pathview(foldchanges, pathway.id = "hsa05310")
```

```
'select()' returned 1:1 mapping between keys and columns
```

```
Info: Working in directory /Users/xinyuwen/ Biology/BIMM 143 R/Class13Work
```

```
Info: Writing image file hsa05310.pathview.png
```

Insert this figure into my report