Class13

Brooke Clements (PID: 17532793)

2026-02-19

This week we are looking at differential expression enalysis.

The data for this hands-on session comes from a published RNA-seq experiment where airway smooth muscle cells were treated with dexamethasone, a synthetic glucocorticoid steroid with anti-inflammatory effects (Himes et al. 2014).

Import countData and colData

```
download.file("https://bioboot.github.io/bimm143_W18/class-material/airway_scaledcounts.csv", destfile =
download.file("https://bioboot.github.io/bimm143_W18/class-material/airway_metadata.csv", destfile = "a
```

```
# Complete the missing code
counts <- read.csv("airway_scaledcount.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
```

```
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
```

Sanity check on correspondence of counts and metadata

```
all (metadata$id == colnames(counts))
```

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

Q1. How many genes are in this dataset?

There are 38694 genes in this dataset.

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

There are 4 control cell lines in this dataset.

Extract and summerize the control samples

To find out where the control samples are we need the metadata

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

Extract and summarize the treated (i.e. drug) samples

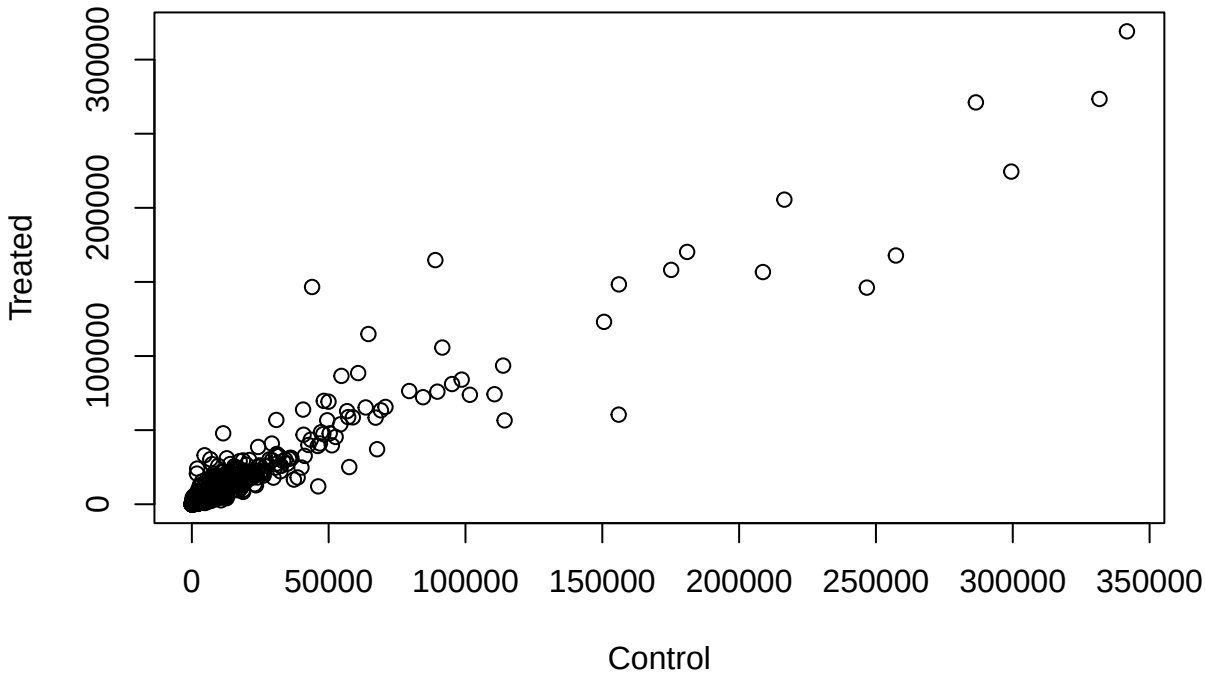
Q4. Follow the same procedure for the treated samples (i.e. calculate the mean per gene across drug treated samples and assign to a labeled vector called `treated.mean`)

```
treated <- metadata[metadata[,"dex"]=="treated",]
treated.counts <- counts [, treated$id]
treated.mean <- rowMeans(treated.counts)
```

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

Lets make a plot to explore thee results a little. > 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.

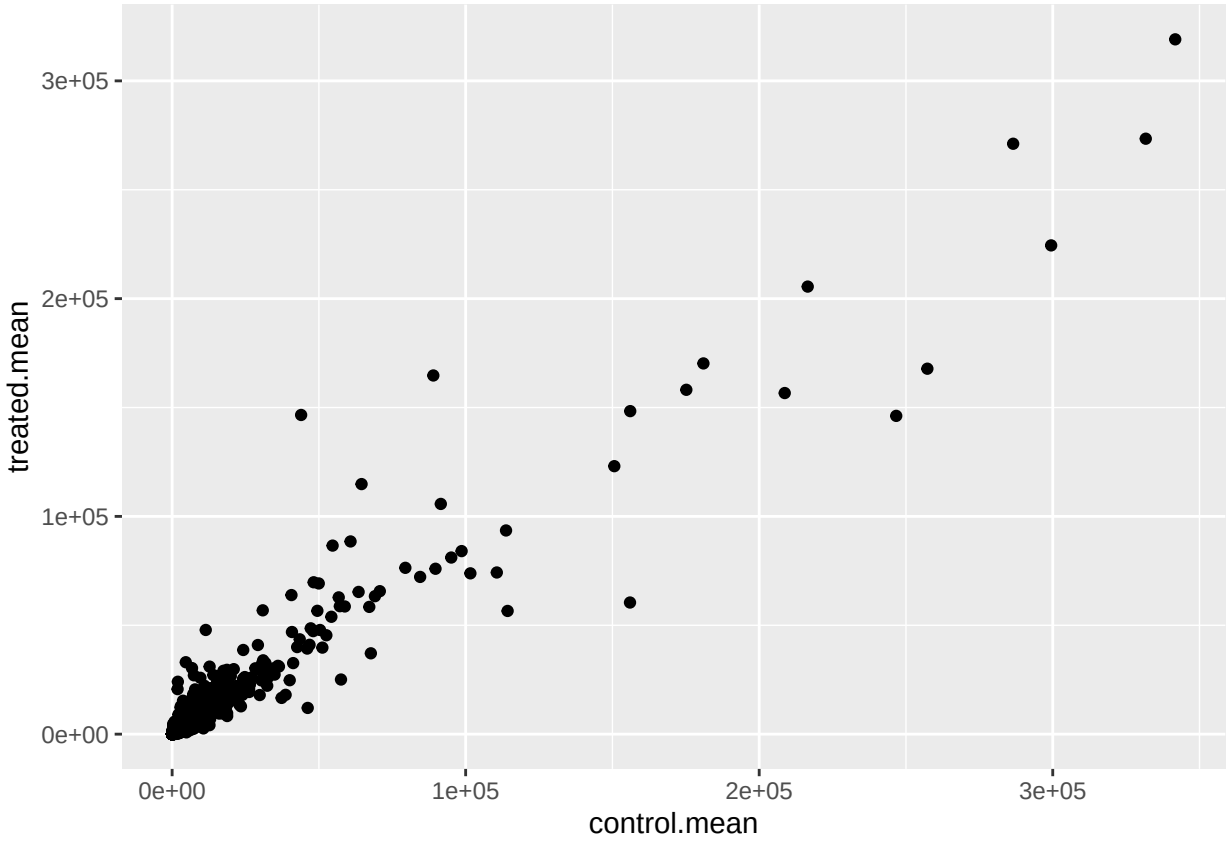
```
plot(meancounts[,1],meancounts[,2], xlab="Control", ylab="Treated")
```



> 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)

ggplot(meancounts) +
  aes(control.mean, treated.mean) +
  geom_point()
```

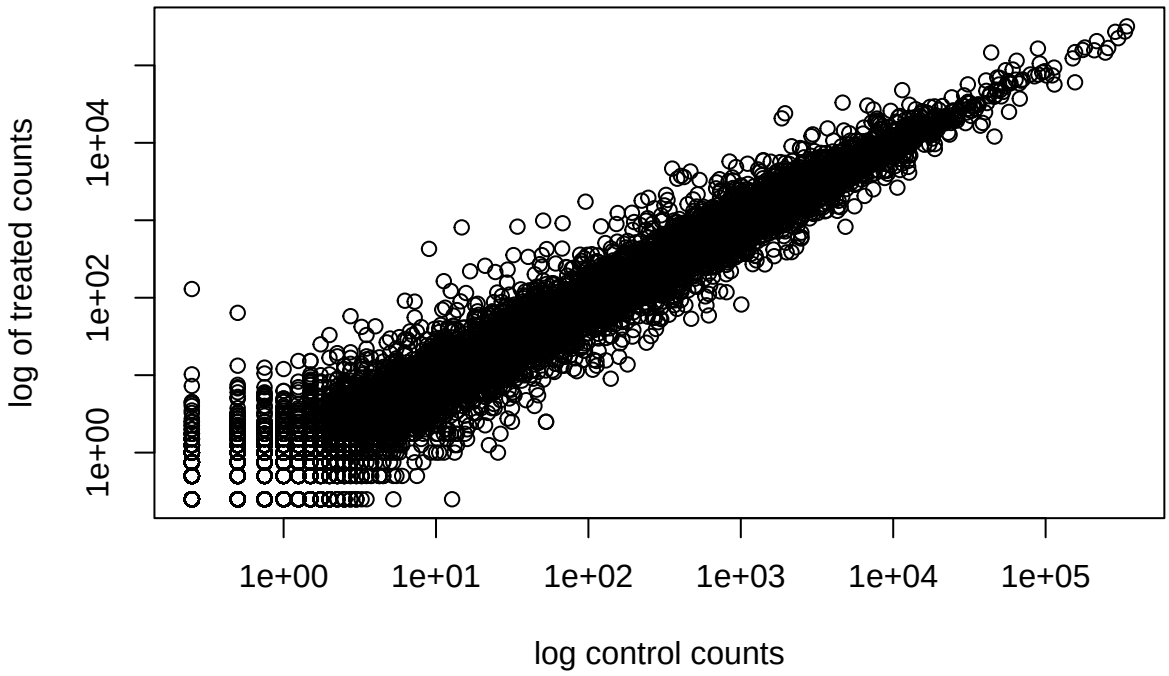


We will make a log-log plot to draw out this skewed data and see ehat is going on. > Q6. Try plotting both axes on a log scale. What is the argument to plot() that allows you to do this?

```
plot(meancounts[,1],meancounts[,2], log="xy",
     xlab="log control counts",
     ylab="log of treated counts")
```

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

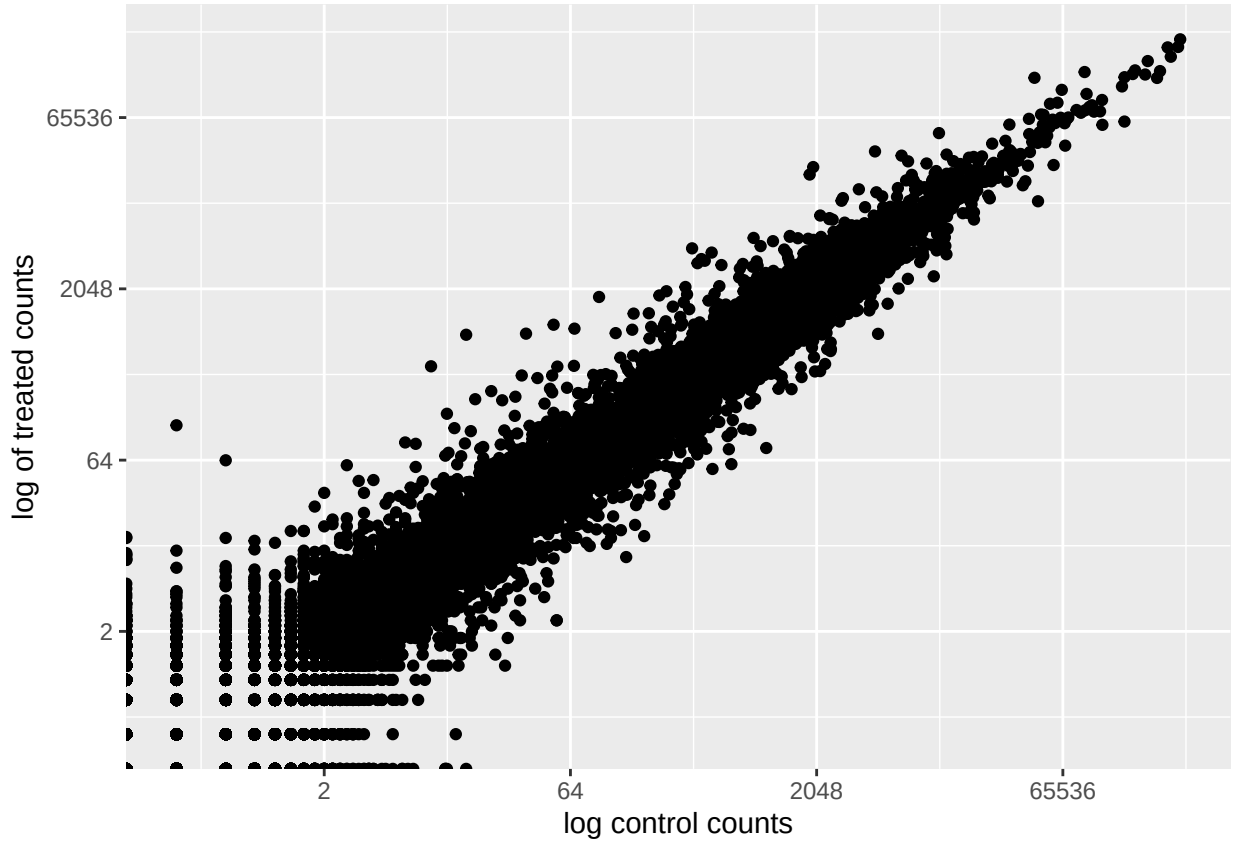
```
## Warning in xy.coords(x, y, xlabel, ylabel, log): 15281 y values <= 0 omitted
## from logarithmic plot
```




```
ggplot(meancounts) +
  aes(x=control.mean, y=treated.mean) +
  geom_point() +
  scale_x_continuous(trans="log2") +
  scale_y_continuous(trans="log2") +
  xlab("log control counts") +
  ylab("log of treated counts")
```

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

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



We often log2 transformations when dealing with the sort of data.

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

```
## [1] 0
```

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

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

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

```
## [1] -1
```

```
log2(80/20)
```

```
## [1] 2
```

This log2 transformation has this nice property where if there is no change the log2 value will be zero and if it double the log will be q and if halved it will be -1.

So lets add a log2 fold change column to our results so far

```
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 get rid of zero count genes that we can not say anything about

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

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

```
##                 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
```

How many genes are remaning?

```
nrow(mycounts)
```

```
## [1] 21817
```

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 tells us which rows and columns the true values are in. The unique () function is so we do not remove the same row multiple times.

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

Use fold change to see up and down regulated genes.

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)
```

```
## [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)
```

```
## [1] 367
```

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

Not fully because we don't yet know if these changes are significant.

Setting up for DESeq

Let's do this the right way. DESeq2 is an R package specifically for analyzing count-based NGS data like RNA-seq. It is available from Bioconductor. Bioconductor is a project to provide tools for analyzing high-throughput genomic data including RNA-seq, ChIP-seq and arrays.

```
#load up DESeq2
library(DESeq2)
```

```
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
```

```
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      -0.350703  0.168242 -2.084514 0.0371134
## ENSG00000000005    0.0000             NA        NA        NA        NA
## ENSG00000000419  520.1342       0.206107  0.101042  2.039828 0.0413675
## ENSG00000000457  322.6648       0.024527  0.145134  0.168996 0.8658000
## ENSG00000000460   87.6826      -0.147143  0.256995 -0.572550 0.5669497
## ...                   ...            ...       ...       ...       ...
## ENSG00000283115  0.000000             NA        NA        NA        NA
## ENSG00000283116  0.000000             NA        NA        NA        NA
## ENSG00000283119  0.000000             NA        NA        NA        NA
## ENSG00000283120  0.974916       -0.66825   1.69441 -0.394385  0.693297
## ENSG00000283123  0.000000             NA        NA        NA        NA
##                      padj
##                 <numeric>
## ENSG00000000003  0.163017
## ENSG00000000005        NA
## ENSG00000000419  0.175937
## ENSG00000000457  0.961682
## ENSG00000000460  0.815805
## ...                   ...
## ENSG00000283115        NA
## ENSG00000283116        NA
## ENSG00000283119        NA
## ENSG00000283120        NA
## ENSG00000283123        NA
```

We can get some bssic summary tallies using the `summary()` function.

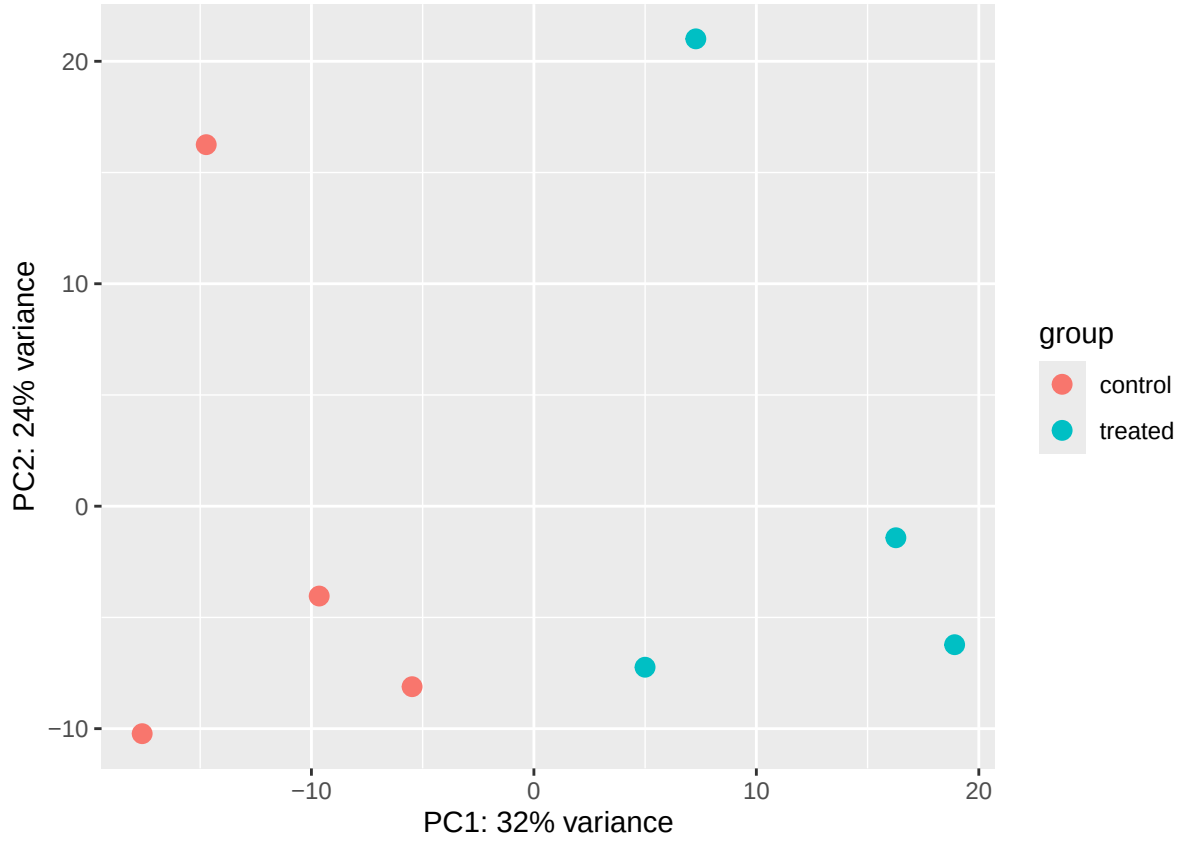
```
summary(res, alpha=0.05)
```

```
##
## out of 25258 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)       : 1242, 4.9%
## LFC < 0 (down)     : 939, 3.7%
## outliers [1]       : 142, 0.56%
## low counts [2]     : 9971, 39%
## (mean count < 10)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Principal Component Analysis (PCA)

```
vsd <- vst(dds, blind = FALSE)
plotPCA(vsd, intgroup = c("dex"))
```

```
## using ntop=500 top features by variance
```




```
pcaData <- plotPCA(vsd, intgroup=c("dex"), returnData=TRUE)
```

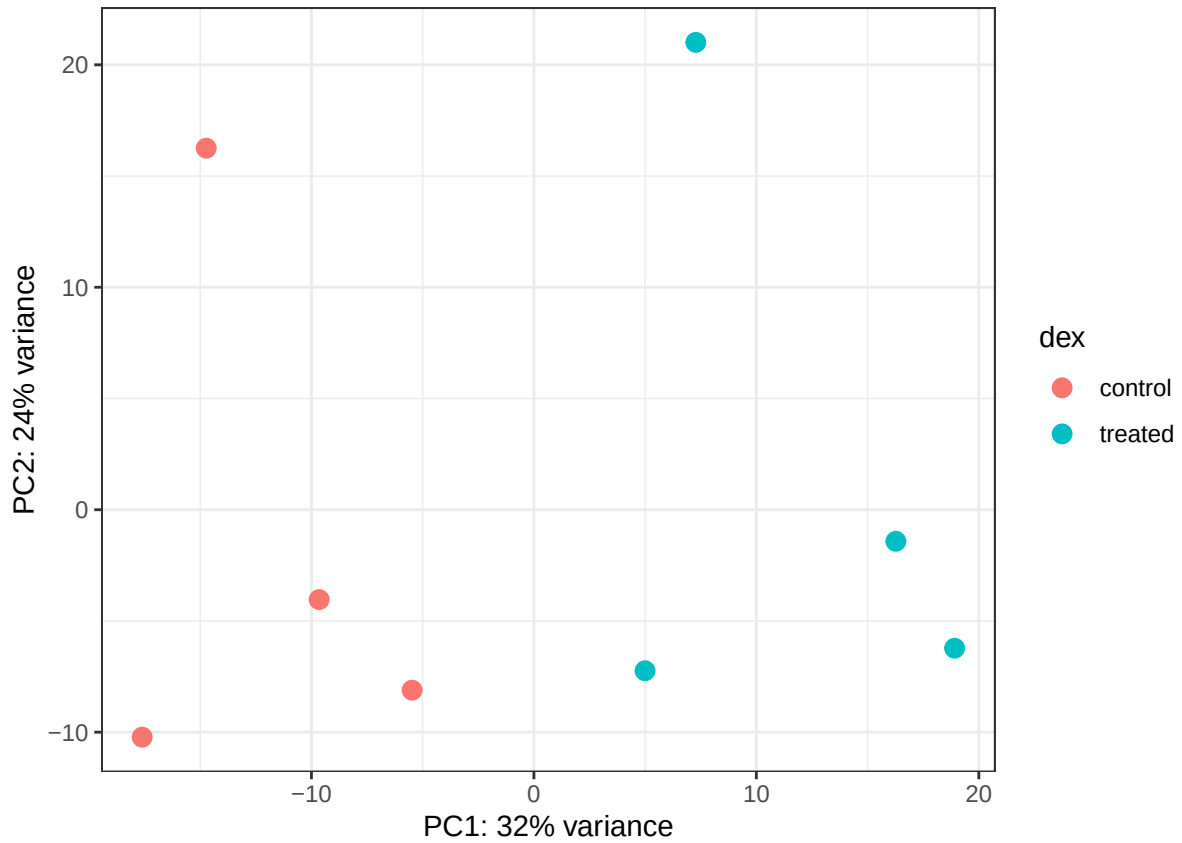
```
## using ntop=500 top features by variance
```

```
head(pcaData)
```

```
##                  PC1        PC2   group       name         id     dex celltype
## SRR1039508 -17.607922 -10.225252 control SRR1039508 SRR1039508 control   N61311
## SRR1039509   4.996738  -7.238117 treated SRR1039509 SRR1039509 treated   N61311
## SRR1039512  -5.474456  -8.113993 control SRR1039512 SRR1039512 control  N052611
## SRR1039513  18.912974  -6.226041 treated SRR1039513 SRR1039513 treated  N052611
## SRR1039516 -14.729173  16.252000 control SRR1039516 SRR1039516 control  N080611
## SRR1039517   7.279863  21.008034 treated SRR1039517 SRR1039517 treated  N080611
##                geo_id sizeFactor
## SRR1039508 GSM1275862  1.0193796
## SRR1039509 GSM1275863  0.9005653
## SRR1039512 GSM1275866  1.1784239
## SRR1039513 GSM1275867  0.6709854
## SRR1039516 GSM1275870  1.1731984
## SRR1039517 GSM1275871  1.3929361
```

```
# Calculate percent variance per PC for the plot axis labels
percentVar <- round(100 * attr(pcaData, "percentVar"))
```

```
ggplot(pcaData) +
  aes(x = PC1, y = PC2, color = dex) +
  geom_point(size =3) +
  xlab(paste0("PC1: ", percentVar[1], "% variance")) +
  ylab(paste0("PC2: ", percentVar[2], "% variance")) +
  coord_fixed() +
  theme_bw()
```



Adding annotation data

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

```
##
```

```
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"
```

```
AnnotationDbi::select
```

```
## standardGeneric for "select" defined from package "AnnotationDbi"
##
## function (x, keys, columns, keytype, ...)
## standardGeneric("select")
## <bytecode: 0x307ca7e78>
## <environment: 0x307cb9350>
## Methods may be defined for arguments: x
## Use  showMethods(select)  for currently available ones.
```

```
res$symbol <- mapIds(org.Hs.eg.db,
                     keys=row.names(res), # Our genenames
                     keytype="ENSEMBL",        # The format of our genenames
                     column="SYMBOL",          # The new format we want to add
                     multiVals="first")
```

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

We need to add missing annotation data to our main "res" result object. This includes the common gene symbol

```
head(res)
```

```
## log2 fold change (MLE): dex treated vs control
## Wald test p-value: dex treated vs control
## DataFrame with 6 rows and 7 columns
##                   baseMean log2FoldChange     lfcSE      stat    pvalue
##                  <numeric>      <numeric> <numeric> <numeric> <numeric>
## ENSG00000000003 747.194195      -0.350703  0.168242 -2.084514 0.0371134
## ENSG00000000005   0.000000             NA        NA        NA        NA
## ENSG00000000419 520.134160       0.206107  0.101042  2.039828 0.0413675
## ENSG00000000457 322.664844       0.024527  0.145134  0.168996 0.8658000
## ENSG00000000460  87.682625      -0.147143  0.256995 -0.572550 0.5669497
## ENSG00000000938   0.319167      -1.732289  3.493601 -0.495846 0.6200029
##                      padj      symbol
##                 <numeric> <character>
## ENSG00000000003  0.163017      TSPAN6
## ENSG00000000005        NA        TNMD
## ENSG00000000419  0.175937        DPM1
## ENSG00000000457  0.961682       SCYL3
## ENSG00000000460  0.815805       FIRRM
```

```
## ENSG00000000938        NA         FGR
```

Q11. Run the mapIds() function two more times to add the Entrez ID and UniProt accession and GENENAME as new columns called res$entrez, res$uniprot and res$genename.

We can use the `mapIds()` function now to "translate" between any of these databases

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

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

```
res$uniprot <- mapIds(org.Hs.eg.db,
                     keys=row.names(res),
                     column="UNIPROT",
                     keytype="ENSEMBL",
                     multiVals="first")
```

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

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

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

```
head(res)
```

```
## log2 fold change (MLE): dex treated vs control
## Wald test p-value: dex treated vs control
## DataFrame with 6 rows and 10 columns
##                   baseMean log2FoldChange     lfcSE      stat    pvalue
##                  <numeric>      <numeric> <numeric> <numeric> <numeric>
## ENSG00000000003 747.194195      -0.350703  0.168242 -2.084514 0.0371134
## ENSG00000000005   0.000000             NA        NA        NA        NA
## ENSG00000000419 520.134160       0.206107  0.101042  2.039828 0.0413675
## ENSG00000000457 322.664844       0.024527  0.145134  0.168996 0.8658000
## ENSG00000000460  87.682625      -0.147143  0.256995 -0.572550 0.5669497
## ENSG00000000938   0.319167      -1.732289  3.493601 -0.495846 0.6200029
##                      padj      symbol      entrez     uniprot
##                 <numeric> <character> <character> <character>
## ENSG00000000003  0.163017      TSPAN6        7105  A0A087WYV6
## ENSG00000000005        NA        TNMD       64102      Q9H2S6
## ENSG00000000419  0.175937        DPM1        8813      H0Y368
## ENSG00000000457  0.961682       SCYL3       57147      X6RHX1
## ENSG00000000460  0.815805       FIRRM       55732      A6NFP1
## ENSG00000000938        NA         FGR        2268      B7Z6W7
##                            genename
##                         <character>
## ENSG00000000003          tetraspanin 6
## ENSG00000000005            tenomodulin
## ENSG00000000419 dolichyl-phosphate m..
## ENSG00000000457 SCY1 like pseudokina..
```

```
## ENSG00000000460 FIGNL1 interacting r..
## ENSG00000000938 FGR proto-oncogene, ..
```

```
ord <- order( res$padj )
#View(res[ord,])
head(res[ord,])
```

```
## log2 fold change (MLE): dex treated vs control
## Wald test p-value: dex treated vs control
## DataFrame with 6 rows and 10 columns
##                   baseMean log2FoldChange     lfcSE      stat      pvalue
##                  <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000152583   954.771        4.36836 0.2371306   18.4217 8.79214e-76
## ENSG00000179094   743.253        2.86389 0.1755659   16.3123 8.06568e-60
## ENSG00000116584  2277.913       -1.03470 0.0650826  -15.8983 6.51317e-57
## ENSG00000189221  2383.754        3.34154 0.2124091   15.7316 9.17960e-56
## ENSG00000120129  3440.704        2.96521 0.2036978   14.5569 5.27883e-48
## ENSG00000148175 13493.920        1.42717 0.1003811   14.2175 7.13625e-46
##                        padj      symbol      entrez     uniprot
##                   <numeric> <character> <character> <character>
## ENSG00000152583 1.33157e-71     SPARCL1        8404      B4E2Z0
## ENSG00000179094 6.10774e-56        PER1        5187      A2I2P6
## ENSG00000116584 3.28806e-53     ARHGEF2        9181  A0A8Q3SIN5
## ENSG00000189221 3.47563e-52        MAOA        4128      B4DF46
## ENSG00000120129 1.59896e-44       DUSP1        1843      B4DRR4
## ENSG00000148175 1.80131e-42        STOM        2040      F8VSL7
##                              genename
##                           <character>
## ENSG00000152583           SPARC like 1
## ENSG00000179094 period circadian reg..
## ENSG00000116584 Rho/Rac guanine nucl..
## ENSG00000189221    monoamine oxidase A
## ENSG00000120129 dual specificity pho..
## ENSG00000148175               stomatin
```

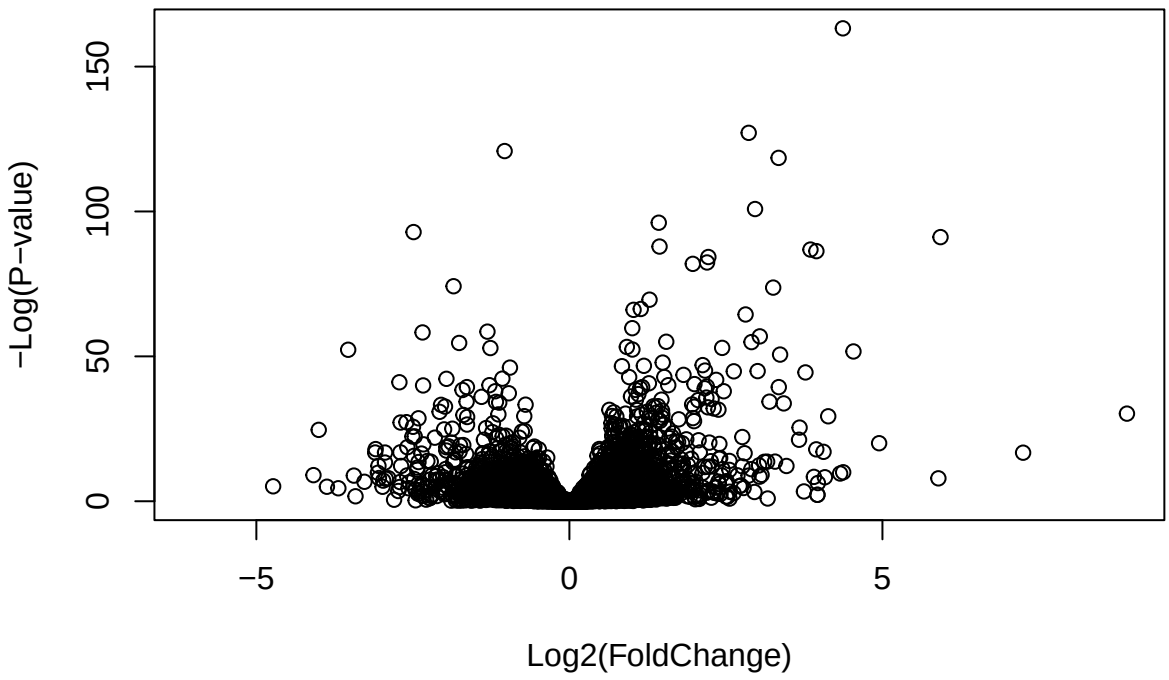
Finish for today by saving our results

```
write.csv(res[ord,], "deseq_results.csv")
```

Volcano Plot

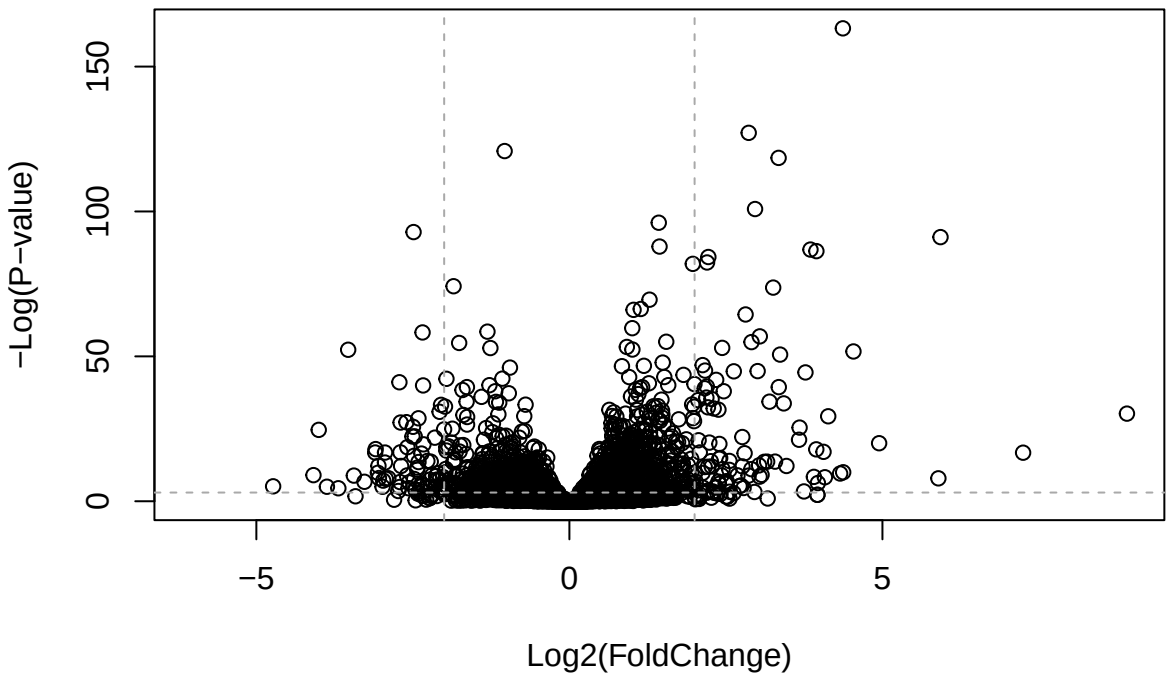
Make a summary plot of our results

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




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

# Add some cut-off lines
abline(v=c(-2,2), col="darkgray", lty=2)
abline(h=-log(0.05), col="darkgray", lty=2)
```

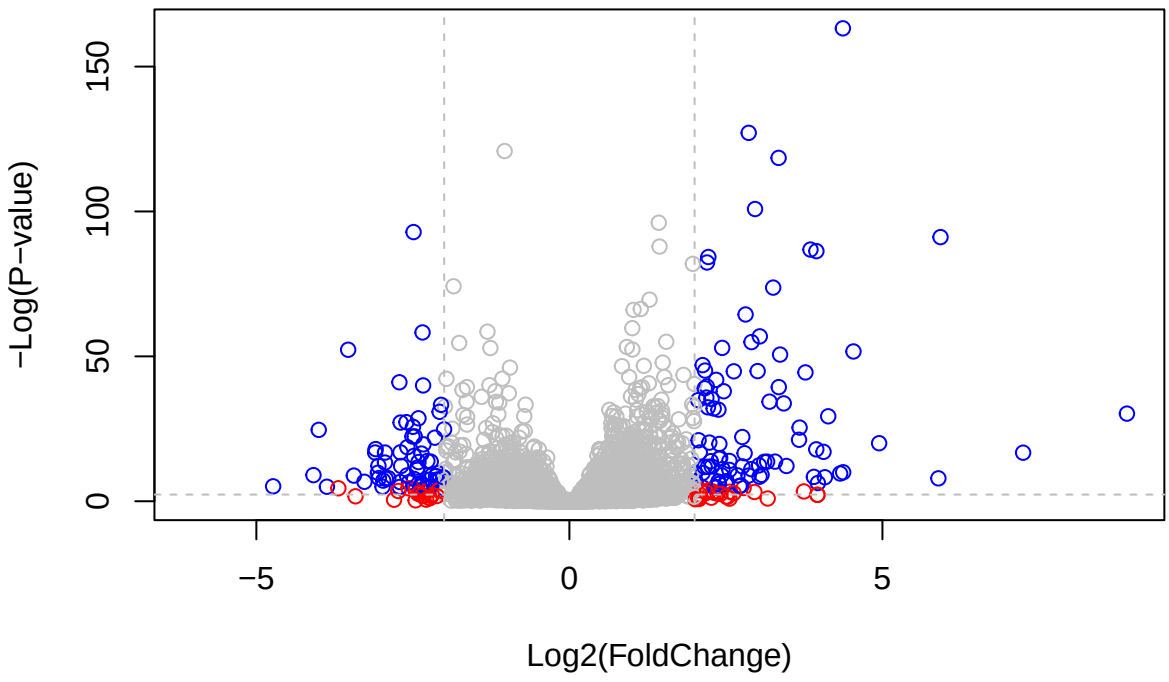



```
# Setup our custom point color vector
mycols <- rep("gray", nrow(res))
mycols[ abs(res$log2FoldChange) > 2 ]  <- "red"

inds <- (res$padj < 0.01) & (abs(res$log2FoldChange) > 2 )
mycols[ inds ] <- "blue"

# Volcano plot with custom colors
plot( res$log2FoldChange,  -log(res$padj),
 col=mycols, ylab="-Log(P-value)", xlab="Log2(FoldChange)" )

# Cut-off lines
abline(v=c(-2,2), col="gray", lty=2)
abline(h=-log(0.1), col="gray", lty=2)
```




```
library(EnhancedVolcano)
```

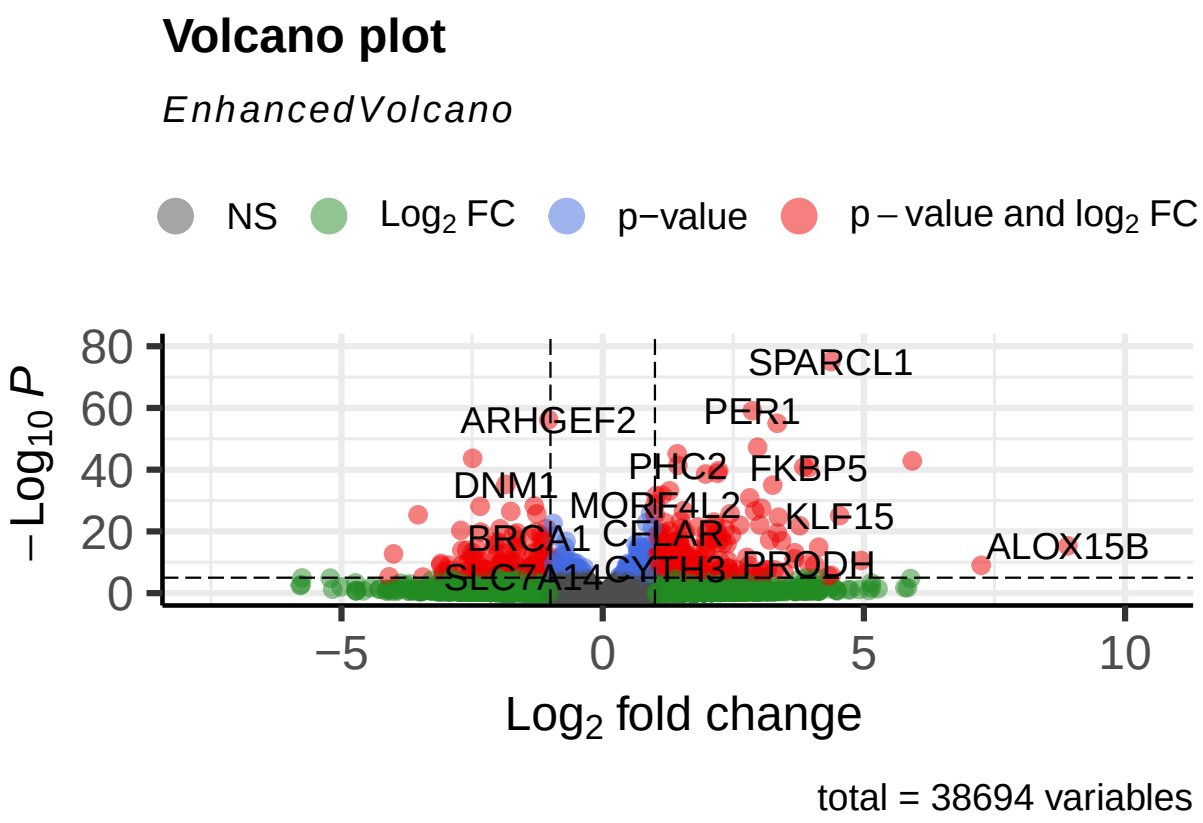
```
## Loading required package: ggrepel
```

```
x <- as.data.frame(res)

EnhancedVolcano(x,
    lab = x$symbol,
    x = 'log2FoldChange',
    y = 'pvalue')
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## i The deprecated feature was likely used in the EnhancedVolcano package.
##   Please report the issue to the authors.
## This warning is displayed once per session.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
## i The deprecated feature was likely used in the EnhancedVolcano package.
##   Please report the issue to the authors.
## This warning is displayed once per session.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Pathway Analysis

What known biological pathways do our differentially expressed genes overlap with (i.e play a role in)? There lot's of bioconductor packages to do this type of analysis. We will use one of the oldest called **gage** along with **pathview** to render nice pics of the pathways we find.

We can instal these with the command : `BioManager::instal(c("pathview", "gage", "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).
## ##############################################################################
```

```
library(gage)
```

```
##
```

```
library(gageData)

data(kegg.sets.hs)
```

```
# Examine the first 2 pathways in this kegg set for humans
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"
```

```
foldchanges = res$log2FoldChange
names(foldchanges) = res$entrez
head(foldchanges)
```

```
##        7105       64102        8813       57147       55732        2268
## -0.35070296          NA  0.20610728  0.02452701 -0.14714263 -1.73228897
```

Have a wee peak what is in gage

```
keggres = gage(foldchanges, gsets=kegg.sets.hs)
```

```
attributes(keggres)
```

```
## $names
## [1] "greater" "less"    "stats"
```

```
# Look at the first three down (less) pathways
head(keggres$less, 3)
```

```
##                                      p.geomean stat.mean        p.val
## hsa05332 Graft-versus-host disease 0.0004250607 -3.473335 0.0004250607
## hsa04940 Type I diabetes mellitus  0.0017820379 -3.002350 0.0017820379
## hsa05310 Asthma                    0.0020046180 -3.009045 0.0020046180
##                                         q.val set.size         exp1
## hsa05332 Graft-versus-host disease 0.09053792       40 0.0004250607
## hsa04940 Type I diabetes mellitus  0.14232788       42 0.0017820379
## hsa05310 Asthma                    0.14232788       29 0.0020046180
```

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

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

```
## Info: Working in directory /Users/brookeclements/BIMM 143/Class13
```

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

```
pathview(gene.data=foldchanges, pathway.id="hsa05310", kegg.native=FALSE)
```

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

```
## Info: Working in directory /Users/brookeclements/BIMM 143/Class13
```

```
## Info: Writing image file hsa05310.pathview.pdf
```