Week 6 DESeq analysis

Angela Abraham

2025-11-06

This week we are looking at differential expression analysis.

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

Import/Read the data frome Himes et al.

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

metadata<-read.csv("airway_metadata.csv")
```

Lets have a wee peak at this data

```
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
```

```
all( c(T,T,F,T))
```

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

Q1. How many genes are in this dataset?

```
nrow(counts)
```

```
## [1] 38694
```

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 summarize 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<-rowMeans(control.counts)
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)
```

Using 'rowSums' would help improve the code to make it more robust.

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

> Q4. Follow the same procedure for the treated samples (1.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)
```

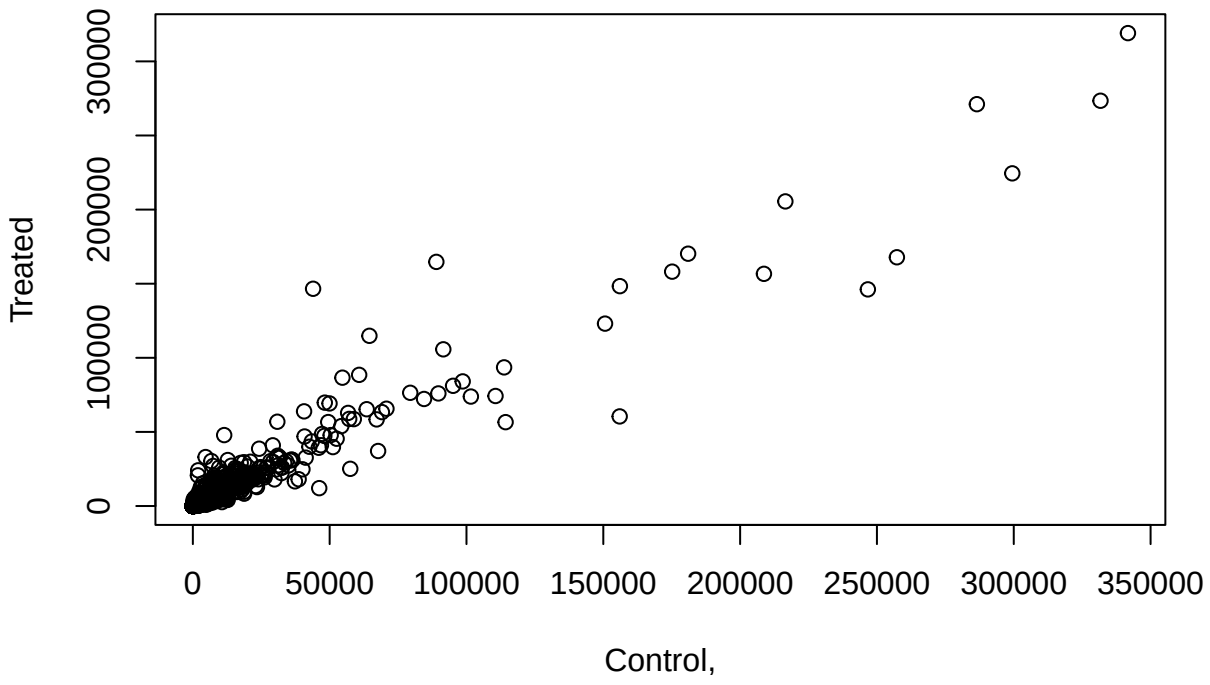
Store these results together in a new data frame called 'meancounts'

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

Lets make a plot to explore the results a little

> Q5(a). Create a scatter plot showing the mean of the treated samples against the mean of the control samples.

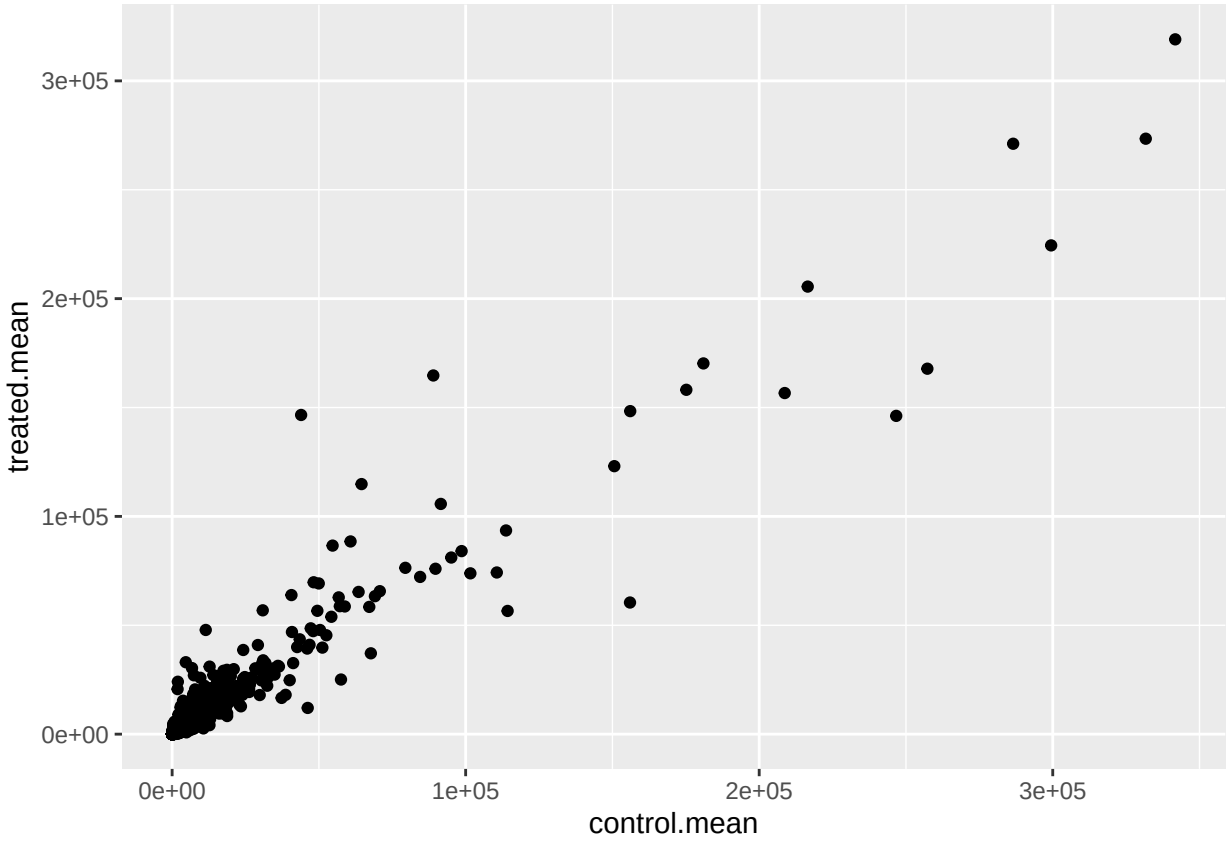
```
plot(meancounts[,1], meancounts[,2], xlab="Control,", ylab="Treated")
```



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

```
library(ggplot2)

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



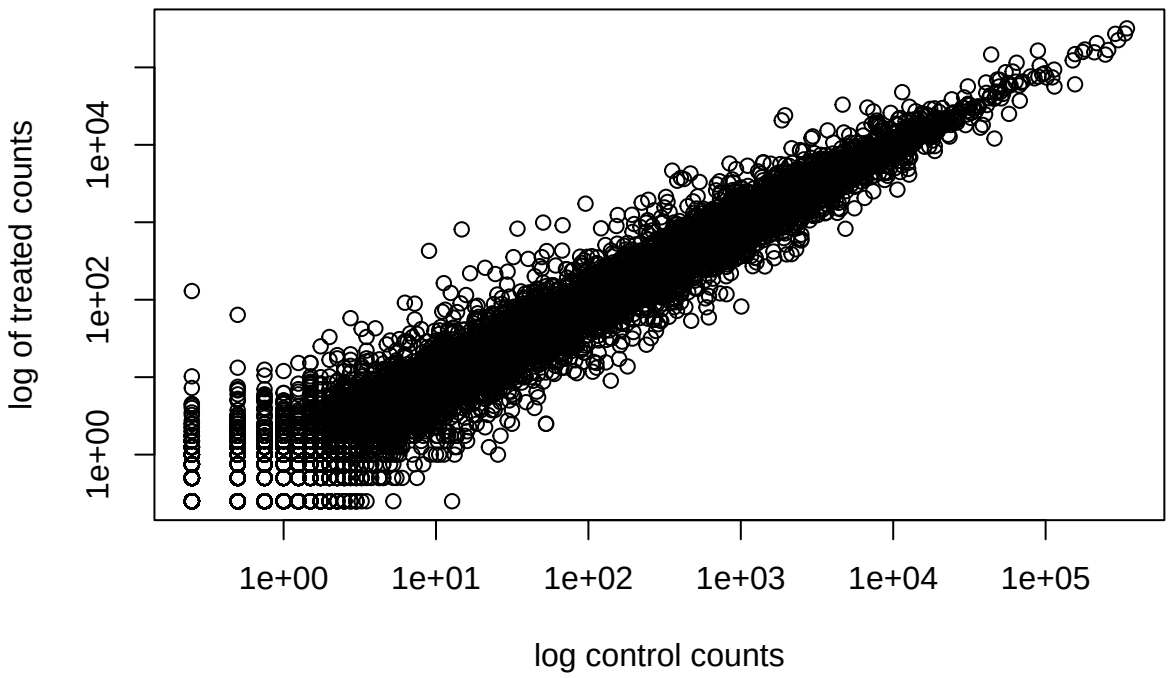
Q6. Try plotting both axes on a log scale. What is the argument to plot() that allows you to do this?

We will make a log-log plot to draw out this skewed data and see what is going on.

```
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
```



We often use log2 transformations when dealing with this 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 mice property where if there is no change, the log2 value will be zero and if it double the log2 value, it will be 1 and if it is 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 >Q7. What is the purpose of the arr.ind arhument in the which() function call above? Why would we then take the first column of the output and need to call the unique() function?

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

```
nrow(mycounts)
```

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

Arr.ind() is to see which gene rows have zeros. The first column is for the row numbers and using the unique() would help us remove each gene once since a gene can be in both columns (treated and in control).

Use fold change to see up and down regulated genes.

A common threshold used for calling something differentially expressed is a log2(FoldChange) of greater than 2 or less than -2. Let's filter the dataset both ways to see how many genes are upi or down-regulated.

> 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(mycounts$log2fc>2)
```

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

> Q9. Using the down.ind vector above, can you detemrine how many down regulated genes we have at the greater than 2 fc level?

and down regulated

```
sum(mycounts$log2fc< -2)
```

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

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

We cannot trust these results fully because we don't yet know if these changes are significant.

DESeq2 analysis

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 upDESeq2
library(DESeq2)
```

```
## Loading required package: S4Vectors

## Loading required package: stats4

## Loading required package: BiocGenerics

## Loading required package: generics

##
## Attaching package: 'generics'

## The following objects are masked from 'package:base':
##
##     as.difftime, as.factor, as.ordered, intersect, is.element, setdiff,
##     setequal, union

##
## 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, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, saveRDS, table, tapply, 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

##
## Attaching package: 'IRanges'

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

## Loading required package: GenomicRanges

## Loading required package: Seqinfo

## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

## Warning: package 'matrixStats' was built under R version 4.5.2

##
## 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
```

```
dds<-DESeqDataSetFromMatrix(countData=counts,
      colData=metadata,
      design=~dex)
```

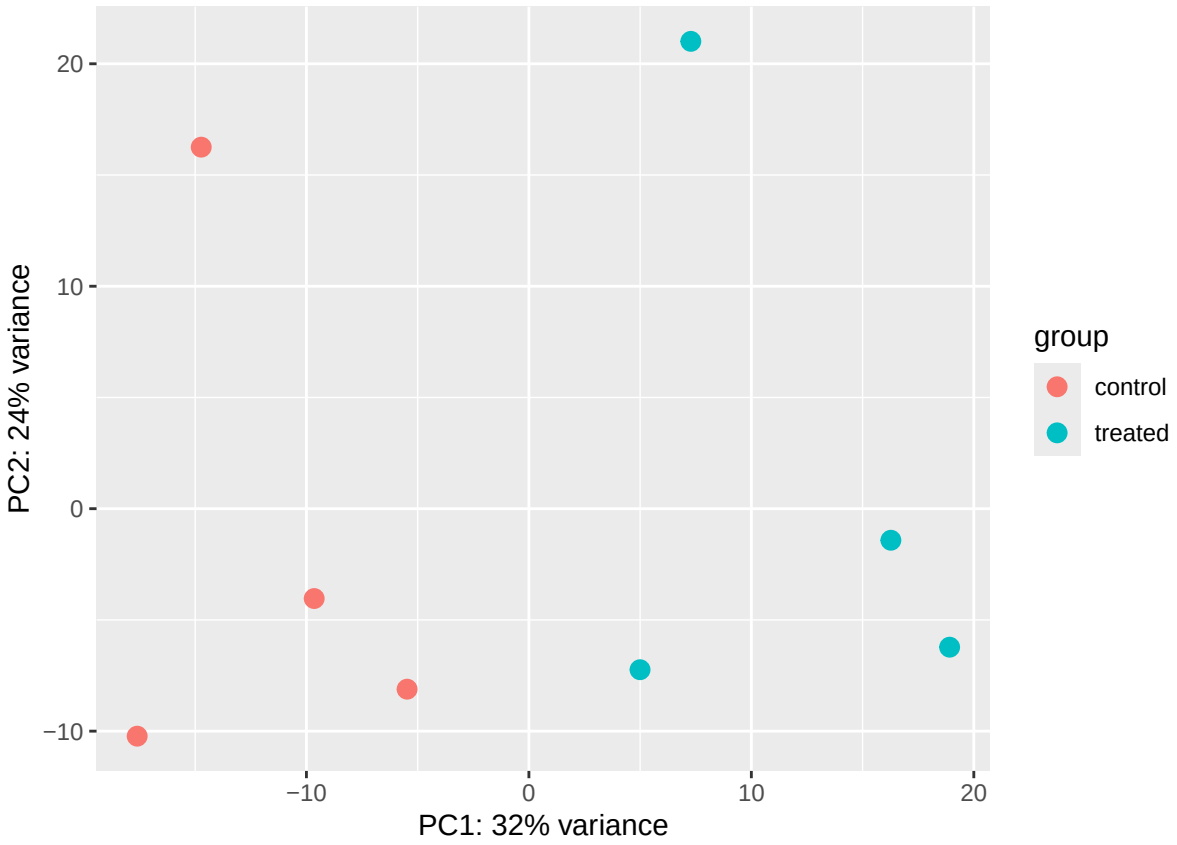
```
## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

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

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

## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## i The deprecated feature was likely used in the DESeq2 package.
##   Please report the issue to the authors.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




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

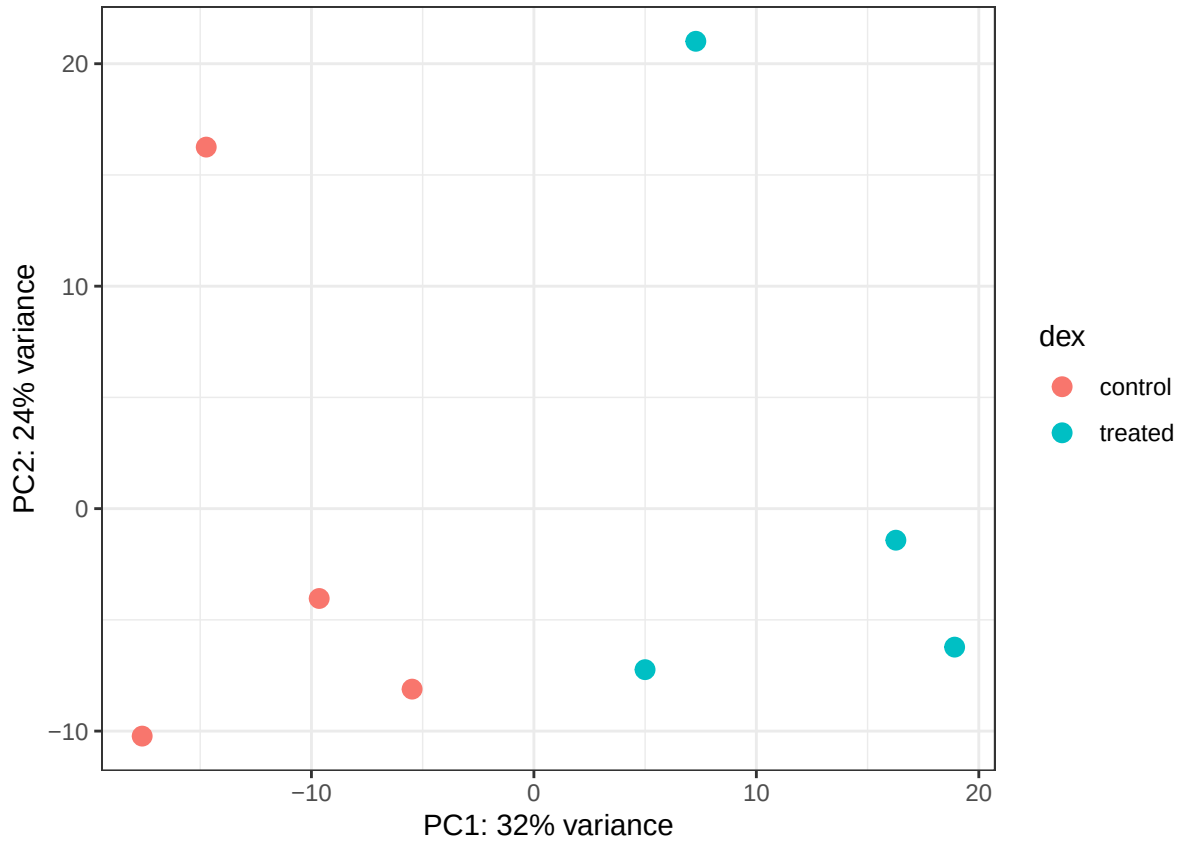
```
## using ntop=500 top features by variance
```

```
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
```

```
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()
```



DESeq analysis

```
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     -0.3507030  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
```

We can get some basic summaries 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
```

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"
```

```
res$symbol<-mapIds(org.Hs.eg.db,
                   keys=row.names(res), keytype="ENSEMBL", column="SYMBOL", 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 7 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      symbol
##                 <numeric> <character>
## ENSG00000000003  0.163035      TSPAN6
## ENSG00000000005        NA        TNMD
## ENSG00000000419  0.176032        DPM1
## ENSG00000000457  0.961694       SCYL3
## ENSG00000000460  0.815849       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.

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

```
## '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="
```

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

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

```
## '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     -0.3507030  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      symbol      entrez     uniprot
##                 <numeric> <character> <character> <character>
## ENSG00000000003  0.163035      TSPAN6        7105  A0A087WYV6
## ENSG00000000005        NA        TNMD       64102      Q9H2S6
## ENSG00000000419  0.176032        DPM1        8813      H0Y368
## ENSG00000000457  0.961694       SCYL3       57147      X6RHX1
## ENSG00000000460  0.815849       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)
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 0.2371268   18.4220 8.74490e-76
## ENSG00000179094   743.253        2.86389 0.1755693   16.3120 8.10784e-60
## ENSG00000116584  2277.913       -1.03470 0.0650984  -15.8944 6.92855e-57
## ENSG00000189221  2383.754        3.34154 0.2124058   15.7319 9.14433e-56
## ENSG00000120129  3440.704        2.96521 0.2036951   14.5571 5.26424e-48
```

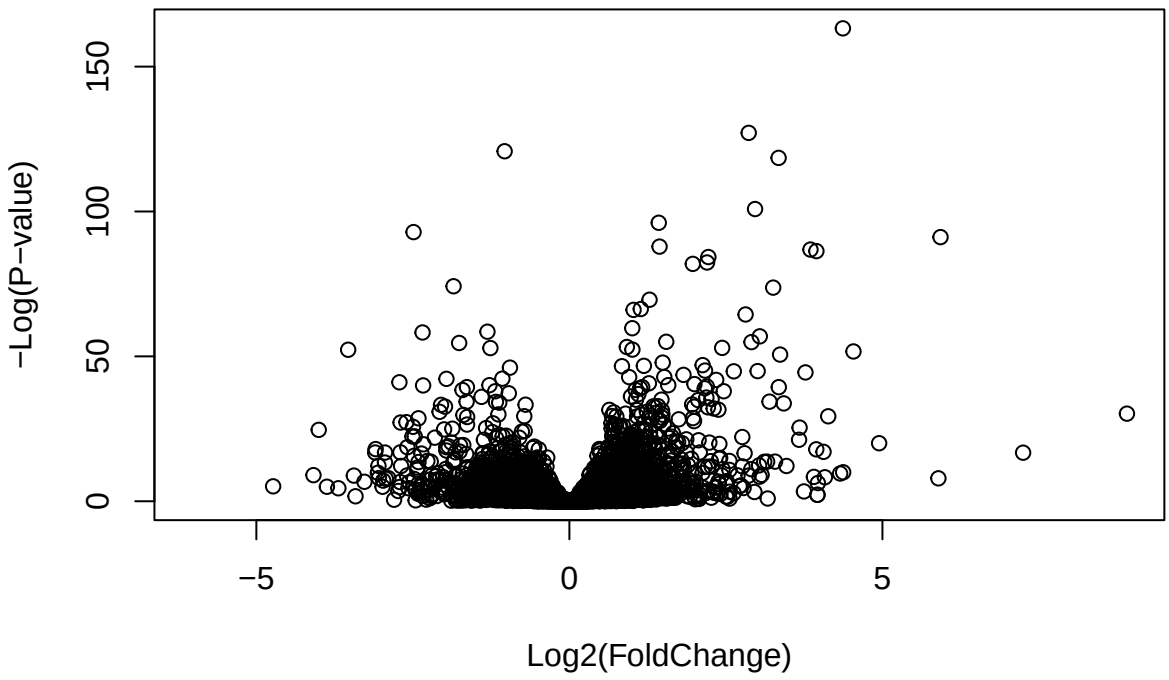
```
## ENSG00000148175 13493.920        1.42717 0.1003890   14.2164 7.25128e-46
##                       padj      symbol      entrez     uniprot
##                  <numeric> <character> <character> <character>
## ENSG00000152583 1.32441e-71     SPARCL1        8404      B4E2Z0
## ENSG00000179094 6.13966e-56        PER1        5187      A2I2P6
## ENSG00000116584 3.49776e-53     ARHGEF2        9181  A0A8Q3SIN5
## ENSG00000189221 3.46227e-52        MAOA        4128      B4DF46
## ENSG00000120129 1.59454e-44       DUSP1        1843      B4DRR4
## ENSG00000148175 1.83034e-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
```

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

Volcano plot

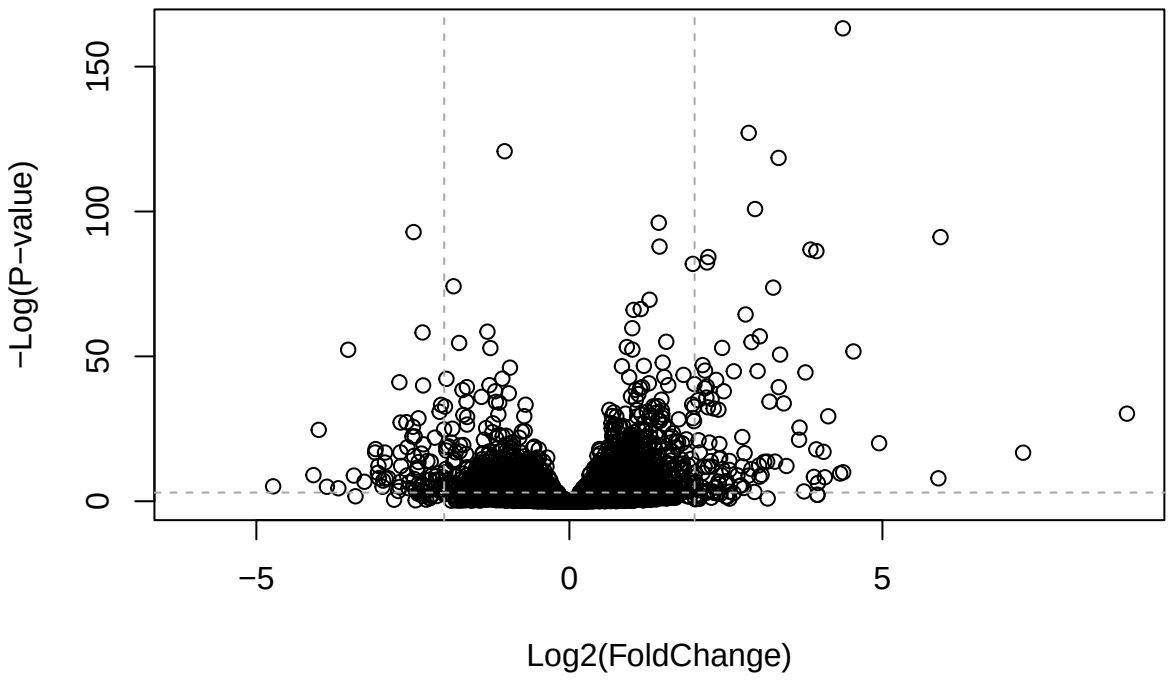
Make a summary plot of our results.

```
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)")

abline(v=c(-2,2),col="darkgray", lty=2)
abline(h=-log(0.05), col="darkgray", lty=2)
```

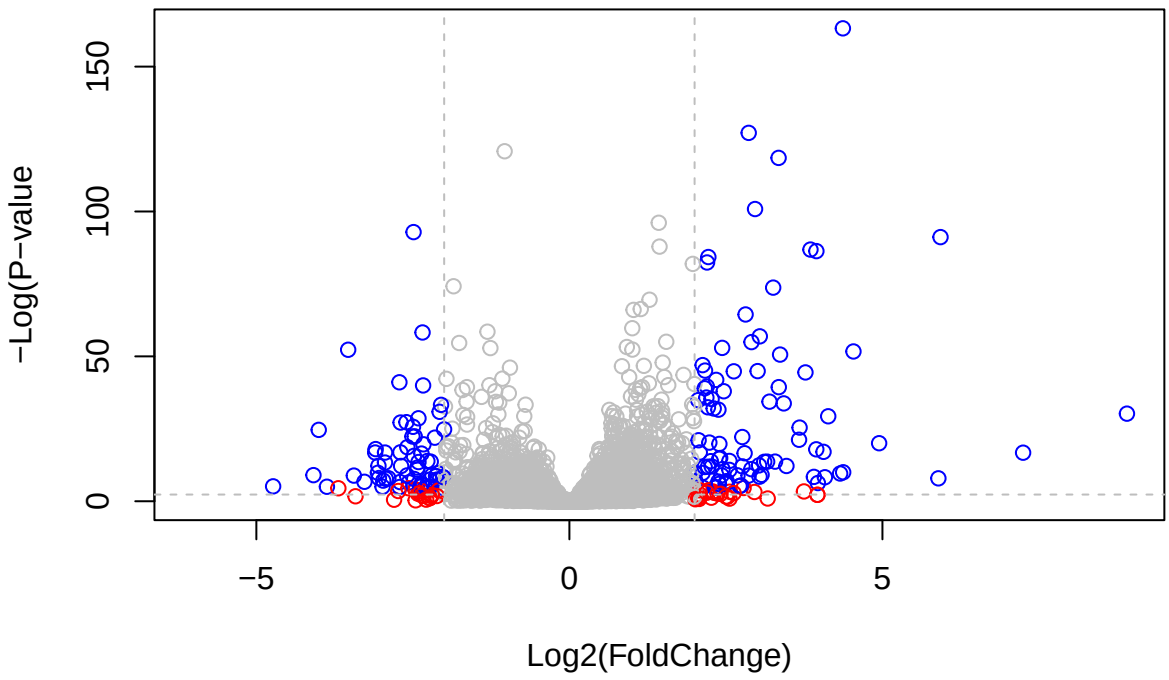



```
mycols<-rep("gray", nrow(res))
mycols[abs(res$log2FoldChange)>2]<-"red"

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

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

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




```
log(0.1)
```

```
## [1] -2.302585
```

```
log(0.005)
```

```
## [1] -5.298317
```

Finish for today by saving our results

```
write.csv(res, file="DESeq2_results.csv")
```