

# Lab 13: DESeq

Nicholas Chiu

### 3. Import countData and colData

```
# Import the data
counts <- read.csv("airway_scaledcounts.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

dim(counts)

[1] 38694     8

dim(metadata)

[1] 8 4

length(metadata$dex[metadata$dex == "control"])

[1] 4

Q1: 38694
Q2: 4

```

#### 4. Toy differential gene expression

```

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
ENSG000000000938
         0.75

## Alternate method
library(dplyr)
#control <- metadata %>% filter(dex=="control")
#control.counts <- counts %>% select(control$id)

```

```

#control.mean <- rowSums(control.counts)/4
#head(control.mean)

treated <- metadata[metadata[, "dex"]=="treated",]
treated.mean <- rowSums( counts[ ,treated$id] ) / 4
head(treated.mean)

ENSG00000000003 ENSG00000000005 ENSG000000000419 ENSG000000000457 ENSG000000000460
    658.00          0.00      546.00      316.50      78.75
ENSG000000000938
    0.00

```

```

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

```

Side-note: I prefer the dplyr method because it is more streamlined and clearly shows what is being accessed via the pipe function.

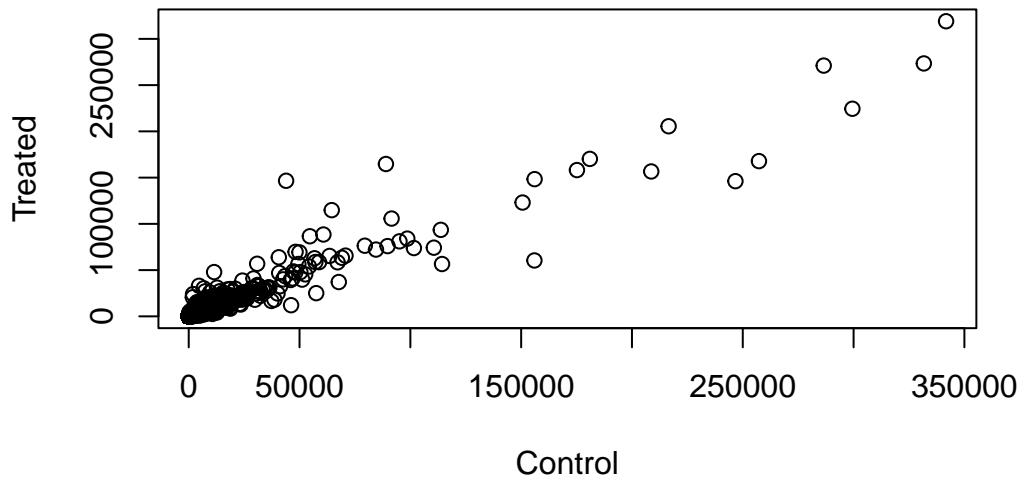
Q3: To make the code more robust, we can use rowMeans instead of rowSums and calculating the mean manually.

Q4: Above

```

# Plot mean of treated against mean of control
plot(meancounts[,1],meancounts[,2], xlab="Control", ylab="Treated")

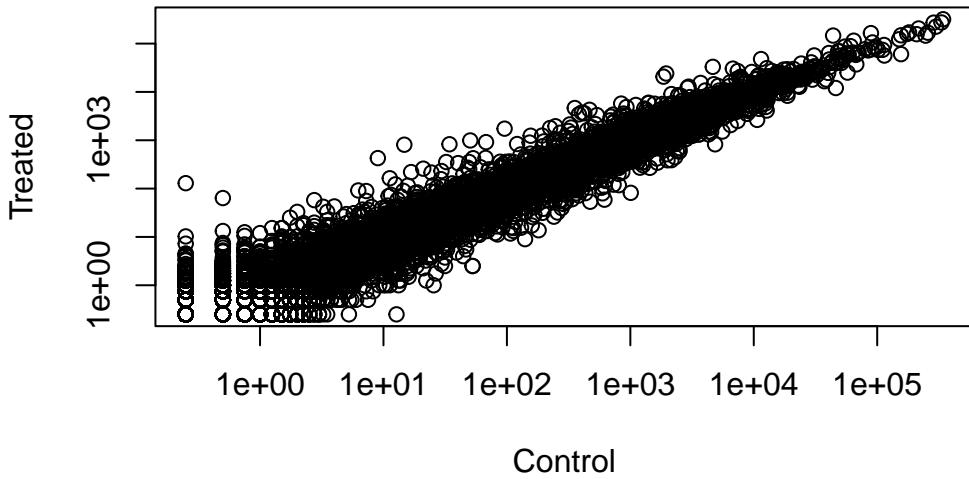
```



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

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: a) Above b) geom\_point()

Q6: Above

```
meancounts$log2fc <- log2(meancounts[, "treated.mean"] / meancounts[, "control.mean"])
head(meancounts)
```

	control.mean	treated.mean	log2fc
ENSG000000000003	900.75	658.00	-0.45303916
ENSG000000000005	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

```
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
ENSG000000000003     900.75     658.00 -0.45303916
ENSG000000000419     520.50     546.00  0.06900279
ENSG000000000457     339.75     316.50 -0.10226805
ENSG000000000460     97.25      78.75 -0.30441833
ENSG000000000971    5219.00    6687.50  0.35769358
ENSG000000001036    2327.00    1785.75 -0.38194109
```

```
up.ind <- mycounts$log2fc > 2
```

```
sum(mycounts$log2fc > 2)
```

```
[1] 250
```

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

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

```
[1] 367
```

Q7: The arr.ind argument returns the array indices if it set to TRUE. We call unique to ensure that we are not double counting the rows with 0 in both samples.

Q8: 250

Q9: 367

Q10: I do not trust these results because there is not enough evidence to support that all fold changes are statistically significant.

## 5. Setting up for DESeq

```
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, setdiff, sort,  
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
```

```
Attaching package: 'IRanges'
```

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

```
windows
```

```
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,
colDiffss, colIQRDiffss, colIQRs, colLogSumExps, colMadDiffss,
colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
colProds, colQuantiles, colRanges, colRanks, colSdDiffss, colSds,
colSums2, colTabulates, colVarDiffss, colVars, colWeightedMads,
colWeightedMeans, colWeightedMedians, colWeightedSds,
colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
rowCumsums, rowDiffss, rowIQRDiffss, rowIQRs, rowLogSumExps,
rowMadDiffss, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
rowSdDiffss, rowSds, rowSums2, rowTabulates, rowVarDiffss, 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
```

```
citation("DESeq2")
```

```
To cite package 'DESeq2' in publications use:
```

```
Love, M.I., Huber, W., Anders, S. Moderated estimation of fold change  
and dispersion for RNA-seq data with DESeq2 Genome Biology 15(12):550  
(2014)
```

```
A BibTeX entry for LaTeX users is
```

```
@Article{,  
  title = {Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2},  
  author = {Michael I. Love and Wolfgang Huber and Simon Anders},  
  year = {2014},  
  journal = {Genome Biology},  
  doi = {10.1186/s13059-014-0550-8},  
  volume = {15},  
  issue = {12},  
  pages = {550},  
}
```

```
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
```

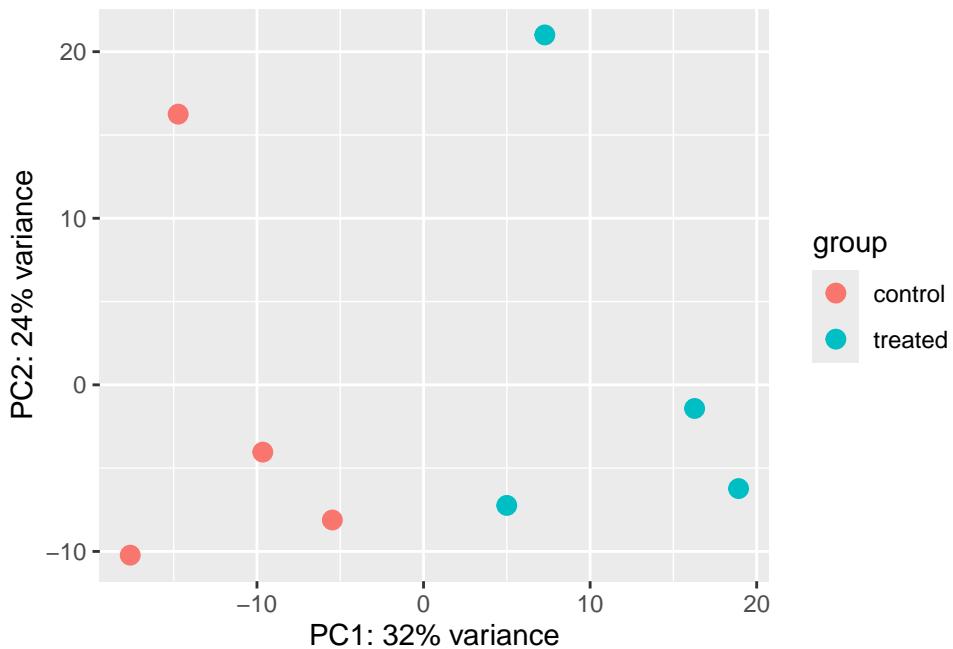
```
class: DESeqDataSet  
dim: 38694 8
```

```
metadata(1): version
assays(1): counts
rownames(38694): ENSG000000000003 ENSG000000000005 ... ENSG00000283120
  ENSG00000283123
rowData names(0):
colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
colData names(4): id dex celltype geo_id
```

## 6. PCA

```
library(ggplot2)
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)
```

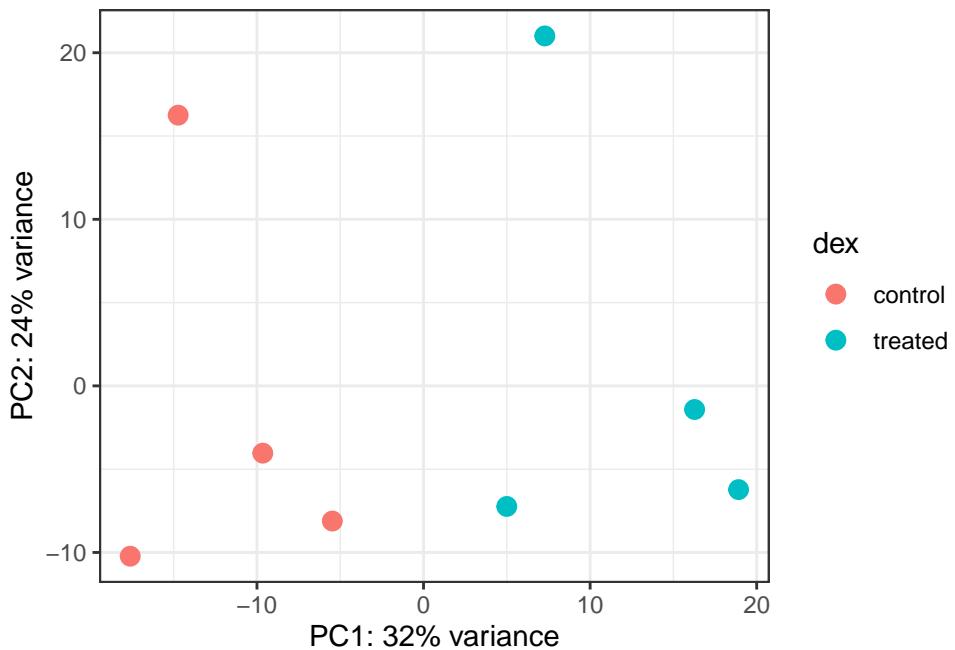
using ntop=500 top features by variance

```
head(pcaData)
```

	PC1	PC2	group	dex	name
SRR1039508	-17.607922	-10.225252	control	control	SRR1039508
SRR1039509	4.996738	-7.238117	treated	treated	SRR1039509
SRR1039512	-5.474456	-8.113993	control	control	SRR1039512
SRR1039513	18.912974	-6.226041	treated	treated	SRR1039513
SRR1039516	-14.729173	16.252000	control	control	SRR1039516
SRR1039517	7.279863	21.008034	treated	treated	SRR1039517

```
# 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()
```



## 7. 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>
ENSG000000000003  747.1942   -0.3507030  0.168246 -2.084470  0.0371175
ENSG000000000005   0.0000      NA        NA        NA        NA
ENSG000000000419  520.1342   0.2061078  0.101059  2.039475  0.0414026
ENSG000000000457  322.6648   0.0245269  0.145145  0.168982  0.8658106
ENSG000000000460   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>
  ENSG000000000003  0.163035
```

```
ENSG00000000005      NA
ENSG00000000419  0.176032
ENSG00000000457  0.961694
ENSG00000000460  0.815849
...
ENSG00000283115      NA
ENSG00000283116      NA
ENSG00000283119      NA
ENSG00000283120      NA
ENSG00000283123      NA
```

```
summary(res)
```

```
out of 25258 with nonzero total read count
adjusted p-value < 0.1
LFC > 0 (up)      : 1563, 6.2%
LFC < 0 (down)     : 1188, 4.7%
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
```

```
res05 <- results(dds, alpha=0.05)
summary(res05)
```

```
out of 25258 with nonzero total read count
adjusted p-value < 0.05
LFC > 0 (up)      : 1236, 4.9%
LFC < 0 (down)     : 933, 3.7%
outliers [1]       : 142, 0.56%
low counts [2]     : 9033, 36%
(mean count < 6)
[1] see 'cooksCutoff' argument of ?results
[2] see 'independentFiltering' argument of ?results
```

## 8. Adding annotation data

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

columns(org.Hs.eg.db)

[1] "ACNUM"      "ALIAS"       "ENSEMBL"      "ENSEMLPROT"   "ENSEMLTRANS"
[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),           # 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

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>
ENSG000000000003 747.194195 -0.3507030  0.168246 -2.084470 0.0371175
ENSG000000000005  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>
ENSG000000000003  0.163035    TSPAN6      7105  AOA024RCI0
ENSG000000000005   NA        TNMD       64102  Q9H2S6
ENSG00000000419   0.176032    DPM1       8813  O60762
ENSG00000000457   0.961694    SCYL3      57147  Q8IZE3
ENSG00000000460   0.815849    FIRRM      55732  AOA024R922
ENSG00000000938   NA        FGR        2268   P09769
      genename
      <character>
ENSG000000000003      tetraspanin 6
ENSG000000000005      tenomodulin
ENSG00000000419      dolichyl-phosphate m..
ENSG00000000457      SCY1 like pseudokina..
```

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

Q11: Above

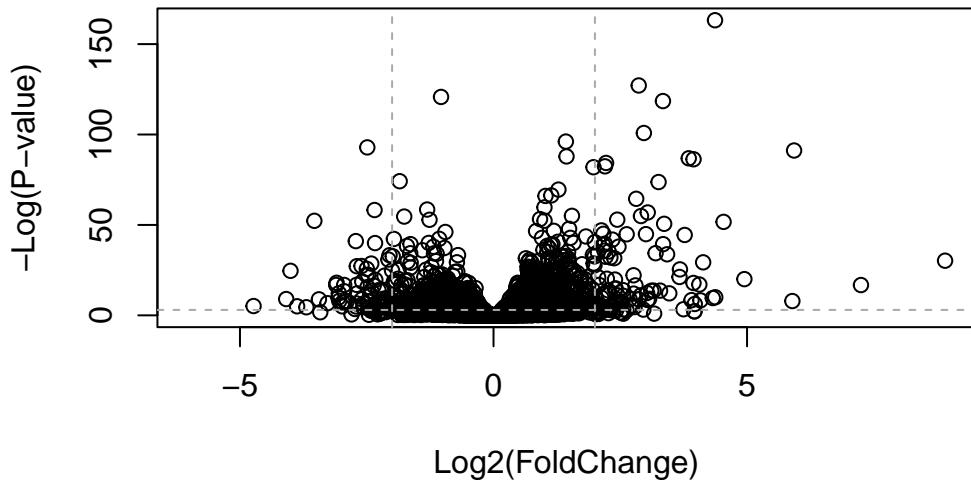
```
ord <- order( res$padj )  
#View(res[ord,])  
head(res[ord,])
```

```
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  0.2371268   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  AOA024RDE1  
ENSG00000179094 6.13966e-56       PER1        5187  O15534  
ENSG00000116584 3.49776e-53      ARHGEF2      9181  Q92974  
ENSG00000189221 3.46227e-52       MAOA        4128  P21397  
ENSG00000120129 1.59454e-44      DUSP1        1843  B4DU40  
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")
```

## 9. Data Visualization

```
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)
```



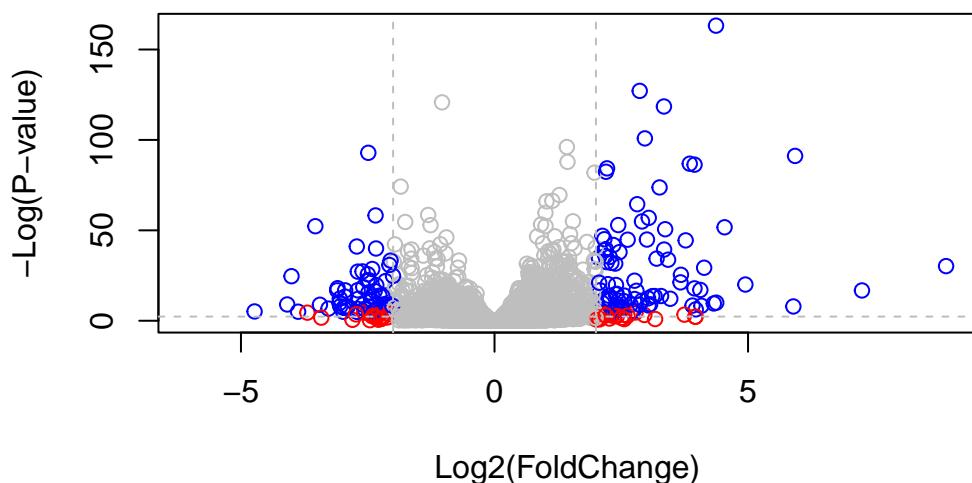
```
# Setup our custom point color vector  
mycols <- rep("gray", nrow(res))  
mycols[ abs(res$log2FoldChange) > 2 ] <- "red"  
  
inds <- (res$padj < 0.01) & (abs(res$log2FoldChange) > 2 )  
mycols[ inds ] <- "blue"  
  
# Volcano plot with custom colors
```

```

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)

```



```

#BiocManager::install("EnhancedVolcano")
library(EnhancedVolcano)

```

Loading required package: ggrepel

```

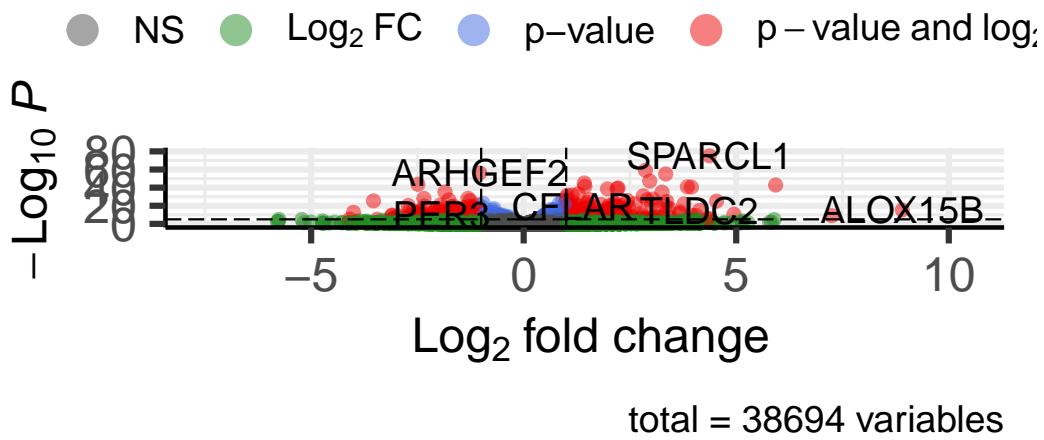
x <- as.data.frame(res)

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

```

## Volcano plot

*Enhanced Volcano*



## 10. Pathway analysis

Optional