

# Class 13: RNASeq Analysis with DESeq2

Hailey Wheeler (A13312713)

```
library(BiocManager)
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
```

```
Loading required package: GenomicRanges
```

```
Loading required package: GenomeInfoDb
```

```
Loading required package: SummarizedExperiment
```

```
Warning: package 'SummarizedExperiment' was built under R version 4.3.2
```

```
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
```

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

	SRR1039508	SRR1039509	SRR1039512	SRR1039513	SRR1039516
ENSG000000000003	723	486	904	445	1170
ENSG000000000005	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		
ENSG000000000003	1097	806	604		
ENSG000000000005	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
```

How many genes are in this data set? 38694 genes

```
dim(counts)
```

```
[1] 38694     8
```

How many ‘control’ cell lines do we have? 4

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

```
[1] 4
```

I want to compare the control to the treated columns. To do this I will: 1. Identify and extract the ‘control’ columns 2. Calculate the mean value per gene for all these ‘control’ columns 3. Do the same for treated 4. Compare ‘control.mean’ to ‘treated.mean’

Q3: How would you make the above code in either approach more robust? Is there a function that could help here?

Method 1

```
control inds <- metadata$dex == "control"
control mean <- rowMeans(counts[,control inds])
head(control mean)
```

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
         900.75          0.00        520.50        339.75         97.25
ENSG00000000938
         0.75
```

Method 2

```
## Calculate mean counts per gene
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
```

Method 3

```
library(dplyr)
```

```
Attaching package: 'dplyr'
```

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

```
combine
```

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

```
count
```

```
The following objects are masked from 'package:GenomicRanges':
```

```
intersect, setdiff, union
```

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

```
intersect
```

```
The following objects are masked from 'package:IRanges':
```

```
collapse, desc, intersect, setdiff, slice, union
```

```
The following objects are masked from 'package:S4Vectors':
```

```
  first, intersect, rename, setdiff, setequal, union
```

```
The following objects are masked from 'package:BiocGenerics':
```

```
  combine, intersect, setdiff, union
```

```
The following objects are masked from 'package:stats':
```

```
  filter, lag
```

```
The following objects are masked from 'package:base':
```

```
  intersect, setdiff, setequal, union
```

```
control <- metadata %>% filter(dex=="control")
control.counts <- counts %>% select(control$id)
control.mean <- rowSums(control.counts)/4
head(control.mean)
```

	ENSG00000000003	ENSG00000000005	ENSG00000000419	ENSG00000000457	ENSG00000000460
control.mean	900.75	0.00	520.50	339.75	97.25
ENSG00000000938					
	0.75				

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 <- rowSums(treated.counts)/4
head(treated.mean)
```

	ENSG00000000003	ENSG00000000005	ENSG00000000419	ENSG00000000457	ENSG00000000460
treated.mean	658.00	0.00	546.00	316.50	78.75
ENSG00000000938					
	0.00				

## Compare the two means

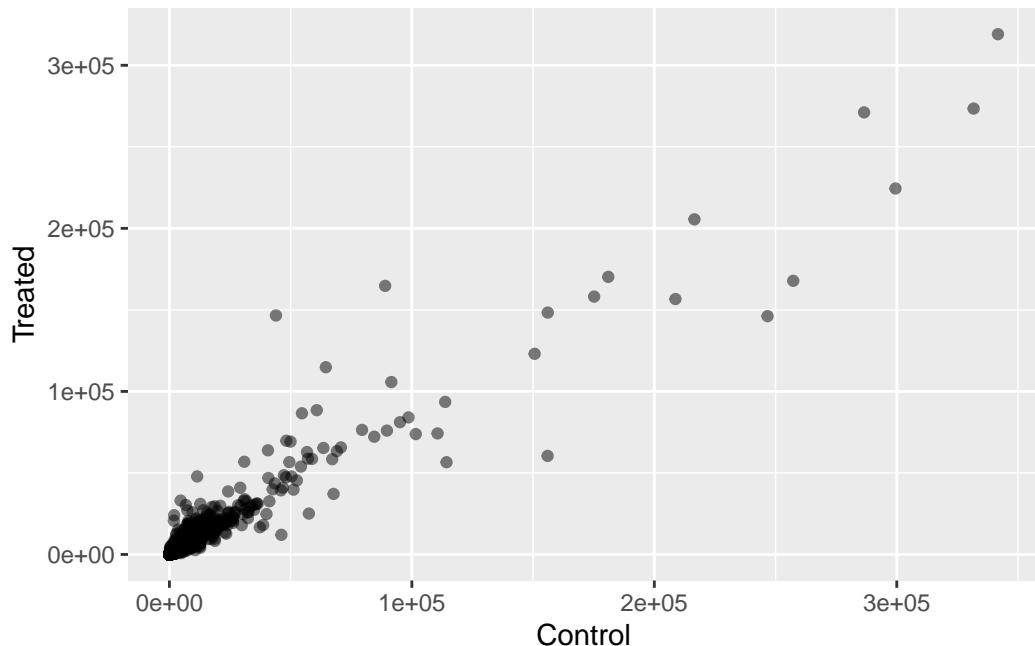
```
meancounts <- data.frame(control.mean, treated.mean)
```

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.

5 (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? geom\_point()

```
library(ggplot2)

ggplot(meancounts, aes(control.mean, treated.mean)) +
  labs(x="Control", y="Treated") +
  geom_point(alpha=0.5)
```



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

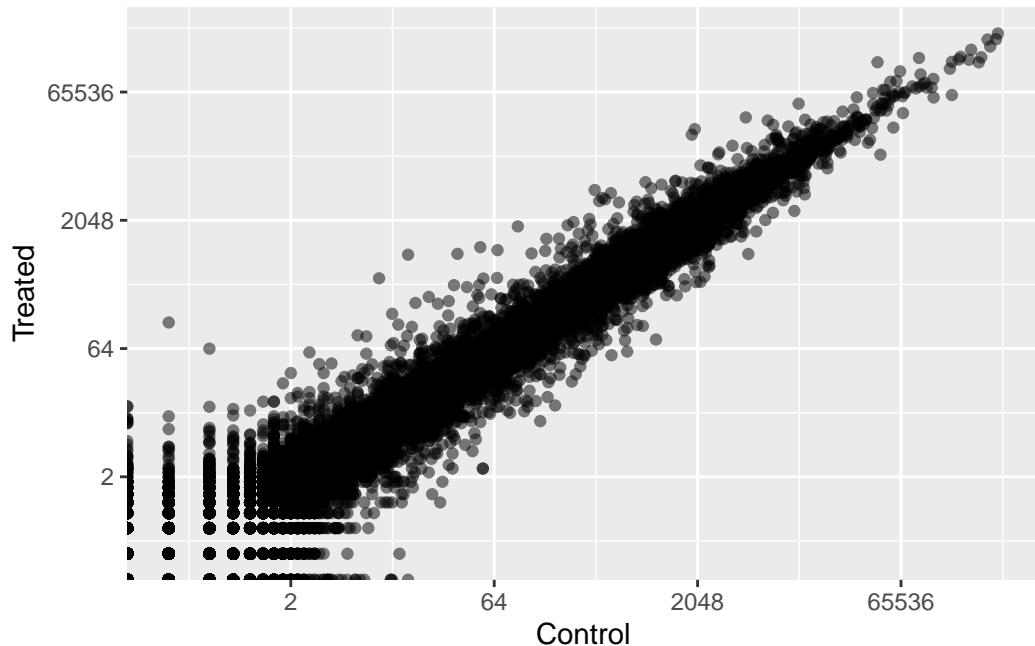
```
library(ggplot2)

ggplot(meancounts, aes(control.mean, treated.mean)) +
  labs(x="Control", y="Treated") +
```

```
scale_x_continuous(trans="log2") +  
scale_y_continuous(trans="log2") +  
geom_point(alpha=0.5)
```

Warning: Transformation introduced infinite values in continuous x-axis

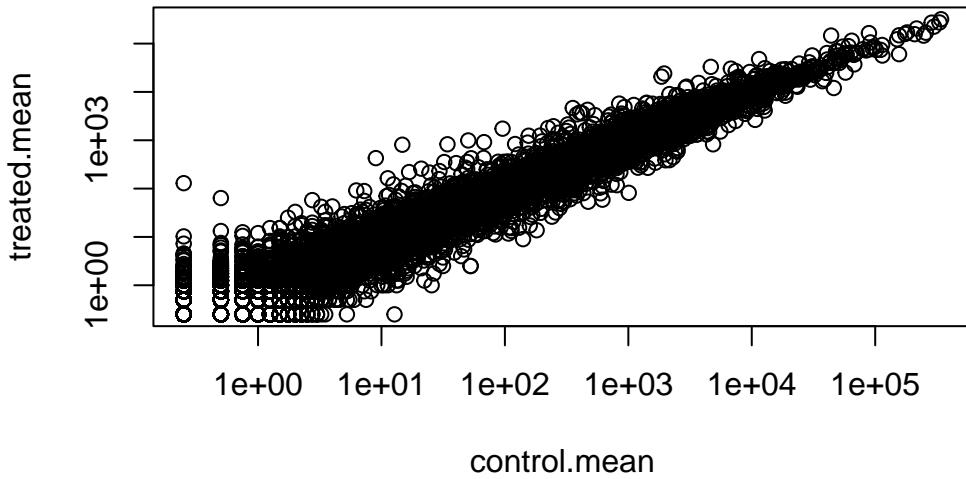
Warning: Transformation introduced infinite values in continuous y-axis



```
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



### Compare the two means

```
## Treated/Control
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
ENSG000000000419	520.50	546.00	0.06900279
ENSG000000000457	339.75	316.50	-0.10226805
ENSG000000000460	97.25	78.75	-0.30441833
ENSG000000000938	0.75	0.00	-Inf

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?

The arr.ind argument included in the which() function will call row and column positions where there are TRUE values (aka, which rows/genes and column/samples are 0) -> Calling unique() will make sure we don't call two of the same samples in case they have zeros in both rows.

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

```
[1] 21817      3
```

```
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
ENSG00000001036	2327.00	1785.75	-0.38194109

Q8. Using the up.ind vector above can you determine how many up regulated genes we have at the greater than 2 fc level? 250.

```
up.ind <- mycounts$log2fc > 2
sum(up.ind == TRUE)
```

```
[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? 367.

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

```
[1] 367
```

Q10. Do you trust these results? Why or why not? No, it is based on fold change which can be large without being statistically significant and thus, misleading.

## D**ESeq2**

```
library(DESeq2)
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},
}
```

## Build the DESeqDataSet (dds) object

```
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
```

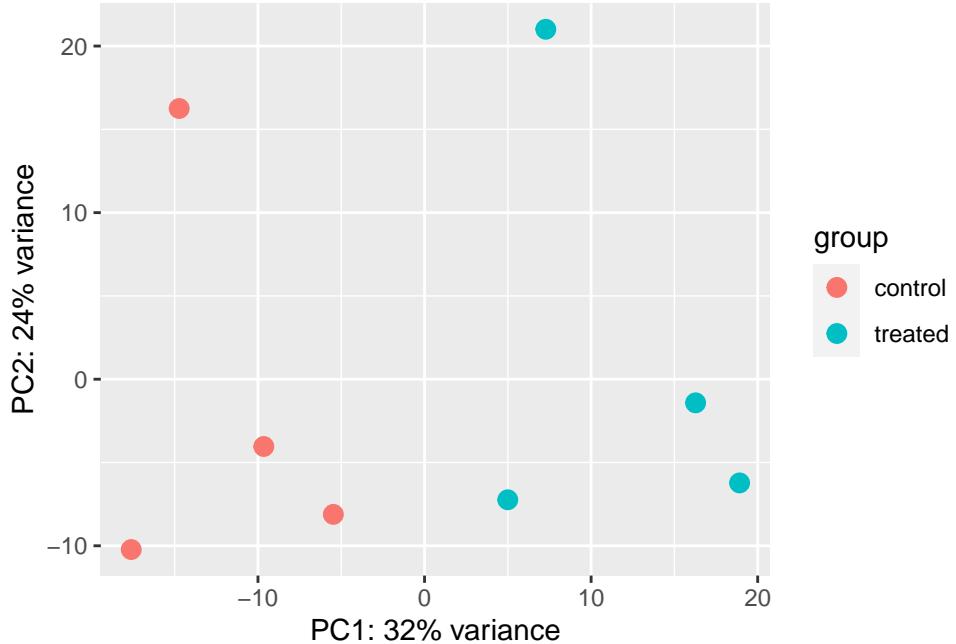
```
class: DESeqDataSet
dim: 38694 8
metadata(1): version
```

```
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
```

“vst” = variance stabilizing transformation intgroup tells R to code based off the groups in the “dex” column

```
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)

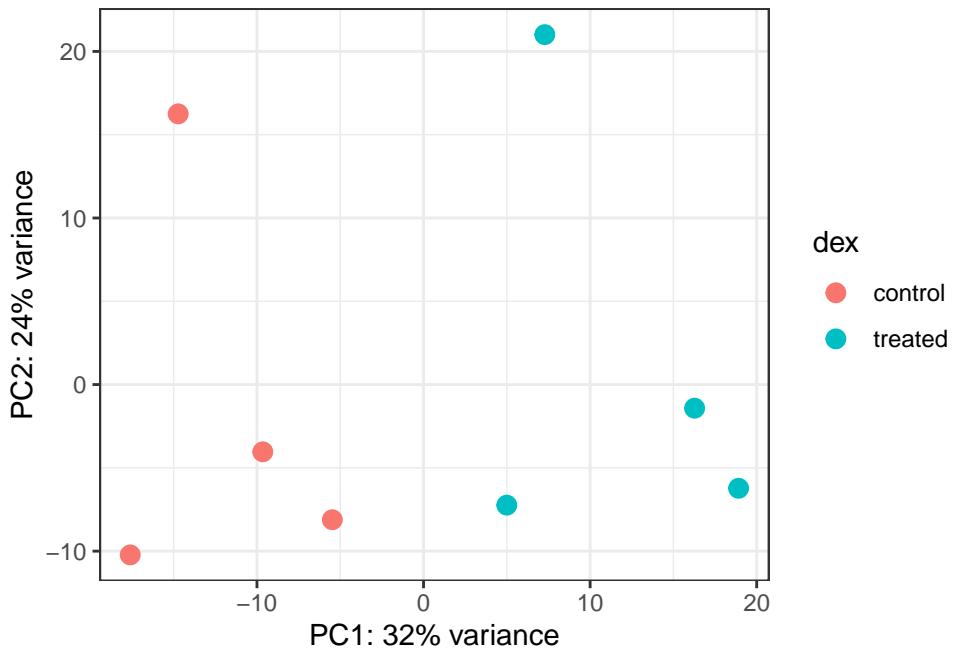
PC1          PC2      group     dex      name
SRR1039508 -17.607922 -10.225252 control control SRR1039508
SRR1039509   4.996738  -7.238117 treated treated SRR1039509
SRR1039512  -5.474456  -8.113993 control control SRR1039512
SRR1039513  18.912974  -6.226041 treated treated SRR1039513
SRR1039516 -14.729173  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"))
percentVar
```

```
[1] 32 24
```

```
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()
```



```

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

res <- as.data.frame(res)
##res

```

```
summary(res)
```

baseMean	log2FoldChange	lfcSE	stat
Min. : 0.0	Min. :-6.030	Min. :0.057	Min. :-15.894
1st Qu.: 0.0	1st Qu.:-0.425	1st Qu.:0.174	1st Qu.: -0.643
Median : 1.1	Median :-0.009	Median :0.445	Median : -0.027
Mean : 570.2	Mean :-0.011	Mean :1.136	Mean : 0.045
3rd Qu.: 201.8	3rd Qu.: 0.306	3rd Qu.:1.848	3rd Qu.: 0.593
Max. :329280.4	Max. : 8.906	Max. :3.534	Max. : 18.422
	NA's :13436	NA's :13436	NA's :13436

pvalue	padj
Min. :0.000	Min. :0.000
1st Qu.:0.168	1st Qu.:0.203
Median :0.533	Median :0.606
Mean :0.495	Mean :0.539
3rd Qu.:0.800	3rd Qu.:0.866
Max. :1.000	Max. :1.000
NA's :13578	NA's :23549

```
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
```

```
library("AnnotationDbi")
```

```
Attaching package: 'AnnotationDbi'
```

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

```
select
```

```
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",      # genenames  
                      column="SYMBOL",       # new format to add  
                      multiVals="first")
```

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

```
head(res)
```

	baseMean	log2FoldChange	lfcSE	stat	pvalue
ENSG00000000003	747.1941954	-0.35070302	0.1682457	-2.0844697	0.03711747
ENSG00000000005	0.0000000		NA	NA	NA
ENSG00000000419	520.1341601	0.20610777	0.1010592	2.0394752	0.04140263
ENSG00000000457	322.6648439	0.02452695	0.1451451	0.1689823	0.86581056
ENSG00000000460	87.6826252	-0.14714205	0.2570073	-0.5725210	0.56696907
ENSG00000000938	0.3191666	-1.73228897	3.4936010	-0.4958463	0.62000288

	padj	symbol
ENSG00000000003	0.1630348	TSPAN6
ENSG00000000005	NA	TNMD

```
ENSG00000000419 0.1760317 DPM1
ENSG00000000457 0.9616942 SCYL3
ENSG00000000460 0.8158486 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.

entrez

```
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

uniprot

```
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

genename

```
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)
```

	baseMean	log2FoldChange	lfcSE	stat	pvalue
ENSG000000000003	747.1941954	-0.35070302	0.1682457	-2.0844697	0.03711747
ENSG000000000005	0.0000000	NA	NA	NA	NA
ENSG000000000419	520.1341601	0.20610777	0.1010592	2.0394752	0.04140263
ENSG000000000457	322.6648439	0.02452695	0.1451451	0.1689823	0.86581056
ENSG000000000460	87.6826252	-0.14714205	0.2570073	-0.5725210	0.56696907
ENSG000000000938	0.3191666	-1.73228897	3.4936010	-0.4958463	0.62000288
	padj	symbol	entrez	uniprot	
ENSG000000000003	0.1630348	TSPAN6	7105	AOA024RC10	
ENSG000000000005	NA	TNMD	64102	Q9H2S6	
ENSG000000000419	0.1760317	DPM1	8813	060762	
ENSG000000000457	0.9616942	SCYL3	57147	Q8IZE3	
ENSG000000000460	0.8158486	FIRRM	55732	AOA024R922	
ENSG000000000938	NA	FGR	2268	P09769	
					genename
ENSG000000000003					tetraspanin 6
ENSG000000000005					tenomodulin
ENSG000000000419					dolichyl-phosphate mannosyltransferase subunit 1, catalytic
ENSG000000000457					SCY1 like pseudokinase 3
ENSG000000000460					FIGNL1 interacting regulator of recombination and mitosis
ENSG000000000938					FGR proto-oncogene, Src family tyrosine kinase

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

	baseMean	log2FoldChange	lfcSE	stat	pvalue
ENSG00000152583	954.7709	4.368359	0.23712679	18.42204	8.744898e-76
ENSG00000179094	743.2527	2.863889	0.17556931	16.31201	8.107836e-60
ENSG00000116584	2277.9135	-1.034701	0.06509844	-15.89440	6.928546e-57
ENSG00000189221	2383.7537	3.341544	0.21240579	15.73189	9.144326e-56
ENSG00000120129	3440.7038	2.965211	0.20369513	14.55710	5.264243e-48
ENSG00000148175	13493.9204	1.427168	0.10038904	14.21638	7.251278e-46
	padj	symbol	entrez	uniprot	
ENSG00000152583	1.324415e-71	SPARCL1	8404	AOA024RDE1	
ENSG00000179094	6.139658e-56	PER1	5187	015534	
ENSG00000116584	3.497761e-53	ARHGEF2	9181	Q92974	
ENSG00000189221	3.462270e-52	MAOA	4128	P21397	
ENSG00000120129	1.594539e-44	DUSP1	1843	B4DU40	
ENSG00000148175	1.830344e-42	STOM	2040	F8VSL7	
					genename
ENSG00000152583					SPARC like 1

```

ENSG00000179094          period circadian regulator 1
ENSG00000116584 Rho/Rac guanine nucleotide exchange factor 2
ENSG00000189221          monoamine oxidase A
ENSG00000120129          dual specificity phosphatase 1
ENSG00000148175          stomatin

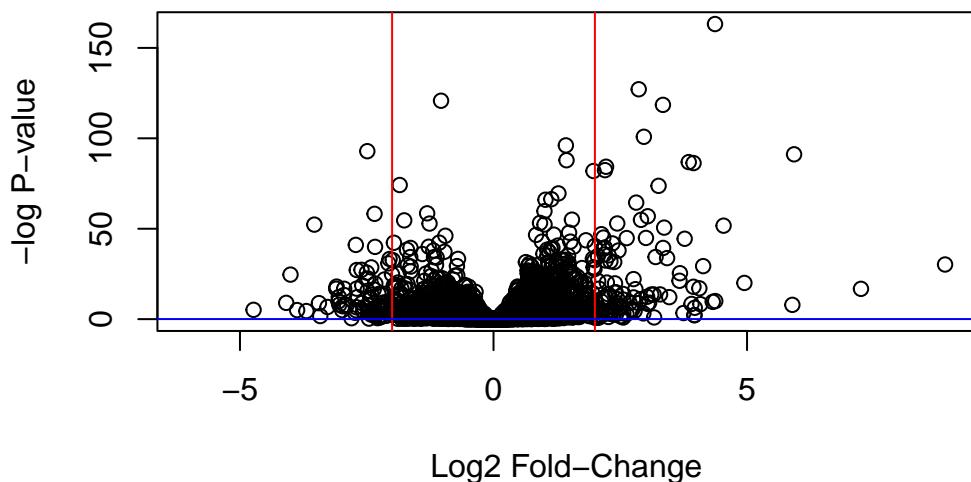
```

```

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

abline(v=c(-2,2), col="red")
abline(h=c(0.05), col="blue")

```



```

mycols <- rep("grey", nrow(res))
##mycols

mycols[res$log2FoldChange > 2] <- "black"
mycols[res$log2FoldChange < -2] <- "black"

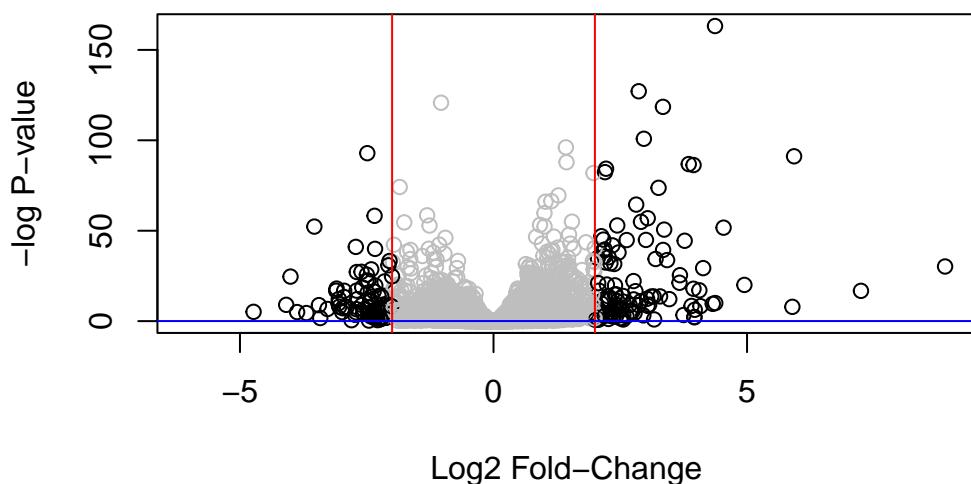
```

```

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

abline(v=c(-2,2), col="red")
abline(h=c(0.05), col="blue")

```



```

library(ggrepel)
library(EnhancedVolcano)

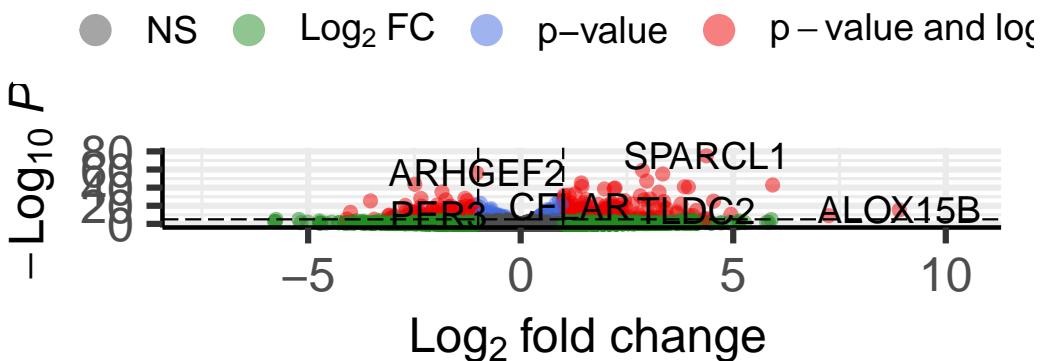
x <- as.data.frame(res)

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

```

# Volcano plot

## Enhanced Volcano



total = 38694 variables

```
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
-0.35070302          NA  0.20610777  0.02452695 -0.14714205 -1.73228897

# Get the results
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 0.0004250461 -3.473346 0.0004250461
hsa04940 Type I diabetes mellitus  0.0017820293 -3.002352 0.0017820293

```

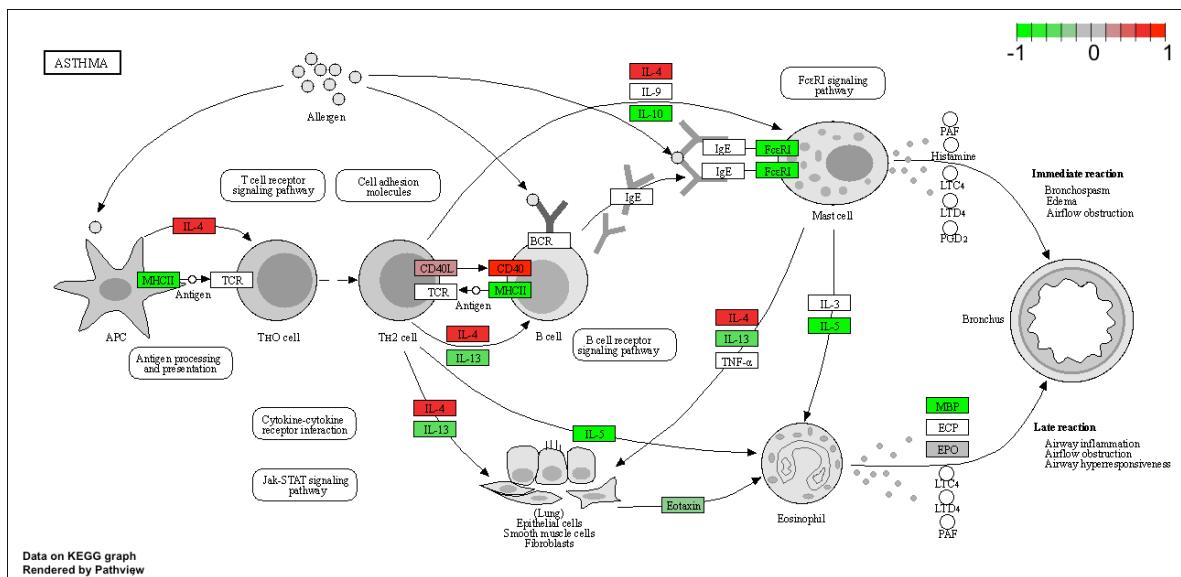
hsa05310 Asthma	0.0020045888	-3.009050	0.0020045888
	q.val	set.size	exp1
hsa05332 Graft-versus-host disease	0.09053483	40	0.0004250461
hsa04940 Type I diabetes mellitus	0.14232581	42	0.0017820293
hsa05310 Asthma	0.14232581	29	0.0020045888

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

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

Info: Working in directory /Users/haileywheeler/Documents/Bioinformatics/R Studio/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/haileywheeler/Documents/Bioinformatics/R Studio/Class13

Info: Writing image file hsa05310.pathview.pdf