

Class_12_Lab

Tessa Sterns PID: A18482353

Table of contents

Import countData and colData	4
Toy differential gene expression	5
Principal Component Analysis (PCA)	11
DESeq analysis	13
Adding annotation data	15

```
library(BiocManager)
```

Bioconductor version '3.21' is out-of-date; the current release version '3.22'
is available with R version '4.5'; see <https://bioconductor.org/install>

```
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

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, colAnyNs, colAvgsPerRowSet, colCollapse,
colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
colDiffs, 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, rowAnyNs, rowAvgsPerColSet,
rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
rowCumsums, rowDiffs, 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
```

Import countData and colData

```
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

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.

Toy differential gene expression

```
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, setequal, union

The following object is masked from 'package:generics':

explain

```
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
      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?

In the third line of code instead of dividing by 4 we could divide by *nrow(control.count)* which would make the code capable of functioning on any sized datatable.

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)

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

```
ENSG00000000003 ENSG00000000005 ENSG00000000419 ENSG00000000457 ENSG00000000460
      658.00          0.00        546.00        316.50        78.75
ENSG00000000938
      0.00
```

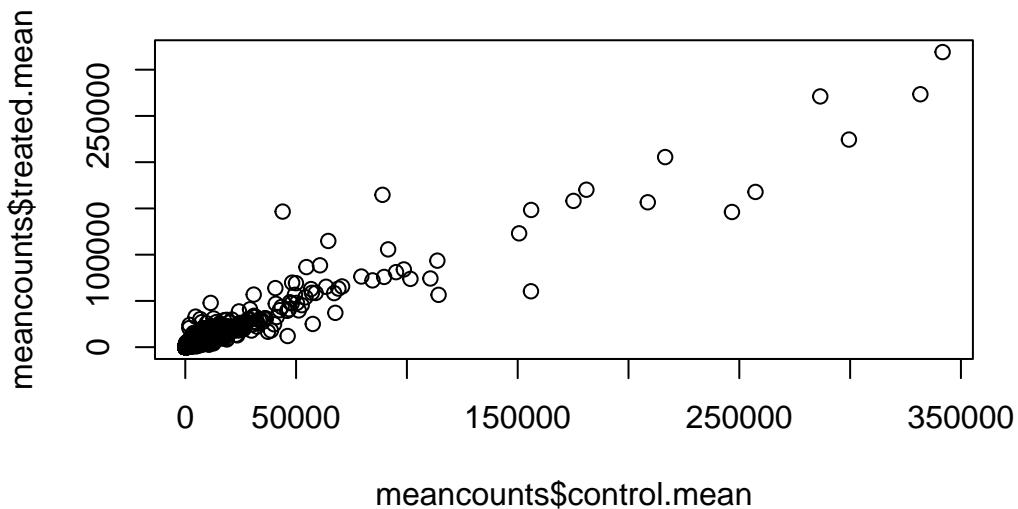
```
meancounts <- data.frame(control.mean, treated.mean)

colSums(meancounts)
```

```
control.mean treated.mean  
23005324     22196524
```

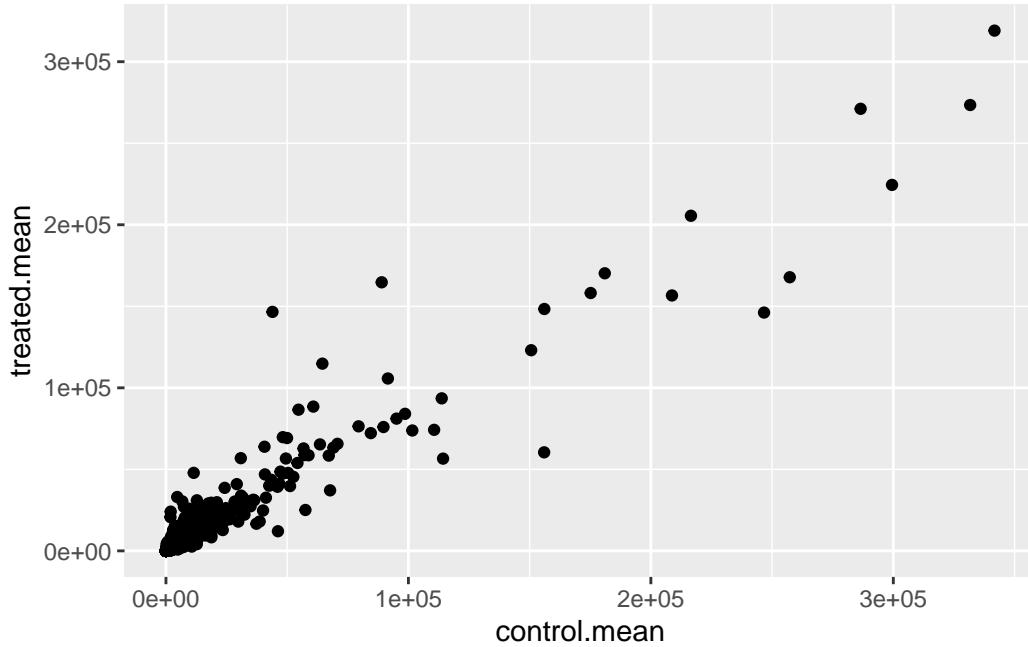
Q5 (a). Create a scatter plot showing the mean of the treated samples against the mean of the control samples.

```
plot(meancounts$control.mean, meancounts$treated.mean)
```



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(x = control.mean, y = treated.mean) +  
  geom_point()
```

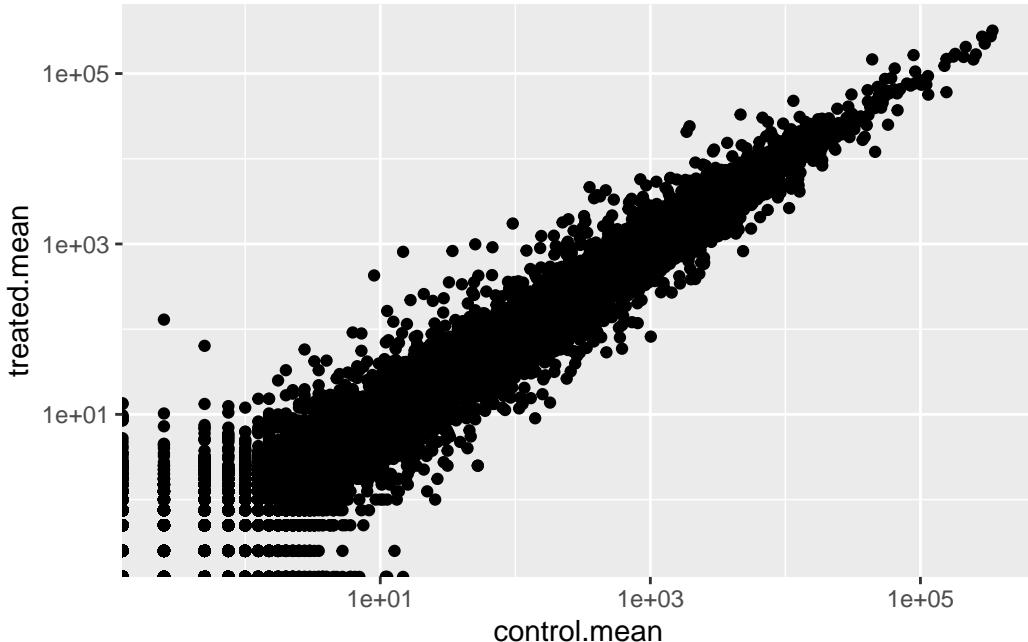


Q6. Try plotting both axes on a log scale. What is the argument to plot() that allows you to do this?

```
library(ggplot2)
ggplot(meancounts) +
  aes(x = control.mean, y = treated.mean) +
  geom_point() +
  scale_x_log10() +
  scale_y_log10()
```

Warning in scale_x_log10(): log-10 transformation introduced infinite values.

Warning in scale_y_log10(): log-10 transformation introduced infinite values.



```
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

```
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

```
ENSG00000000971      5219.00      6687.50  0.35769358
ENSG00000001036      2327.00      1785.75 -0.38194109
```

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=TRUE removes any of the points with a zero value that were causing the errors when attempting to convert the data to a log10 scale. The unique function ensures that there are no repeats in the new dataframe.

log2 is commonly used for up/down regulation metrics.

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

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

There are 250 up regulated genes.

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

There are 367 down regulated genes.

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

Because the results are based on a fold change they might be misleading. It is possible for there to be a fold change that doesn't effect the activity of the protein.

```
##Setting up for DESeq
```

```
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},
}
```

```
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): ENSG00000000003 ENSG00000000005 ... ENSG00000283120
  ENSG00000283123
rowData names(0):
colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
colData names(4): id dex celltype geo_id
```

Principal Component Analysis (PCA)

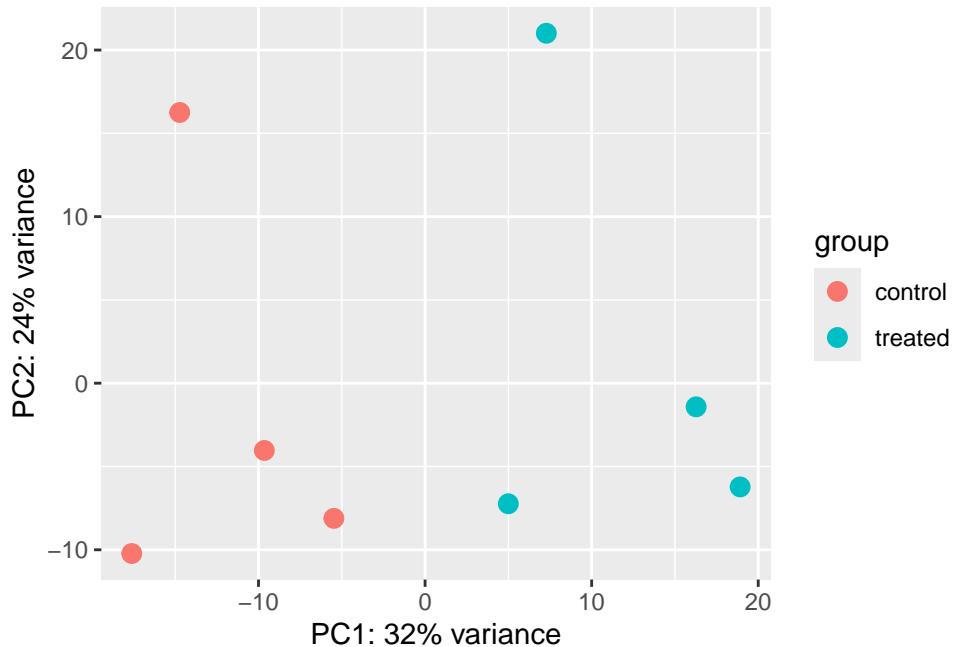
```
vsd <- vst(dds, blind = FALSE)
plotPCA(vsd, intgroup = c("dex"))
```

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.

```



```
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

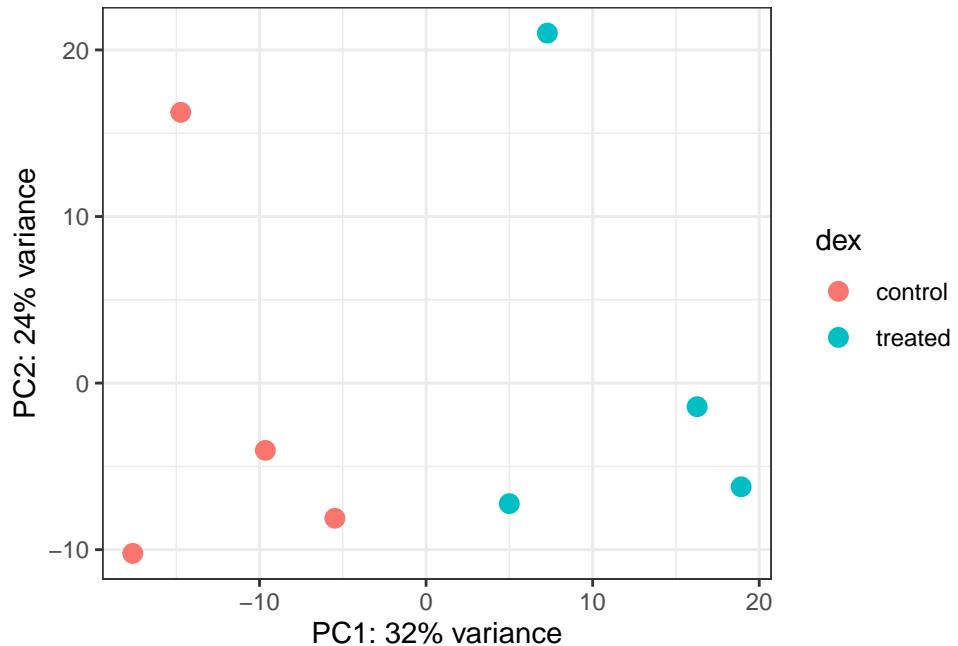
```

```

# 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()

```



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    -0.350703  0.168242 -2.084514 0.0371134
ENSG000000000005    0.0000       NA        NA        NA        NA
ENSG000000000419   520.1342    0.206107  0.101042  2.039828 0.0413675
ENSG000000000457   322.6648    0.024527  0.145134  0.168996 0.8658000
ENSG000000000460   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>
ENSG000000000003   0.163017
ENSG000000000005       NA
ENSG000000000419   0.175937
ENSG000000000457   0.961682
ENSG000000000460   0.815805
...
          ...
ENSG00000283115       NA

```

```
ENSG00000283116      NA
ENSG00000283119      NA
ENSG00000283120      NA
ENSG00000283123      NA
```

```
View(as.data.frame(res))
```

Those with a base mean of zero are not represented with a fold change or p-value.

```
summary(res)
```

```
out of 25258 with nonzero total read count
adjusted p-value < 0.1
LFC > 0 (up)      : 1564, 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
```

Base the alpha is set to 0.01 for most biological stats alpha is typically set to 0.05.

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

```
out of 25258 with nonzero total read count
adjusted p-value < 0.05
LFC > 0 (up)      : 1237, 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
```

Adding annotation data

```
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] "ACCNUM"      "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

```
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>
ENSG000000000003 747.194195     -0.350703  0.168242 -2.084514 0.0371134
ENSG000000000005  0.000000        NA        NA        NA        NA
ENSG000000000419 520.134160     0.206107  0.101042  2.039828 0.0413675
ENSG000000000457 322.664844     0.024527  0.145134  0.168996 0.8658000
ENSG000000000460 87.682625     -0.147143  0.256995 -0.572550 0.5669497
ENSG000000000938 0.319167     -1.732289  3.493601 -0.495846 0.6200029
      padj      symbol
      <numeric> <character>
ENSG000000000003 0.163017      TSPAN6
ENSG000000000005  NA          TNMD
ENSG000000000419 0.175937      DPM1
ENSG000000000457 0.961682      SCYL3
ENSG000000000460 0.815805      FIRRM
ENSG000000000938  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="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 -0.350703  0.168242 -2.084514 0.0371134
ENSG000000000005  0.000000      NA       NA       NA       NA
ENSG000000000419 520.134160  0.206107  0.101042  2.039828 0.0413675
ENSG000000000457 322.664844  0.024527  0.145134  0.168996 0.8658000
ENSG000000000460 87.682625  -0.147143  0.256995 -0.572550 0.5669497
ENSG000000000938 0.319167  -1.732289  3.493601 -0.495846 0.6200029
      padj      symbol     entrez     uniprot
      <numeric> <character> <character> <character>
ENSG000000000003 0.163017   TSPAN6      7105 AOA087WYV6
ENSG000000000005  NA        TNMD       64102 Q9H2S6
ENSG000000000419 0.175937   DPM1       8813 H0Y368
ENSG000000000457 0.961682   SCYL3      57147 X6RHX1
ENSG000000000460 0.815805   FIRRM      55732 A6NFP1
ENSG000000000938  NA        FGR        2268 B7Z6W7
      genename
      <character>
ENSG000000000003      tetraspanin 6
ENSG000000000005      tenomodulin
ENSG000000000419      dolichyl-phosphate m..
ENSG000000000457      SCY1 like pseudokina..
ENSG000000000460      FIGNL1 interacting r..
ENSG000000000938      FGR proto-oncogene, ..

```

Arranging by p-value

```
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

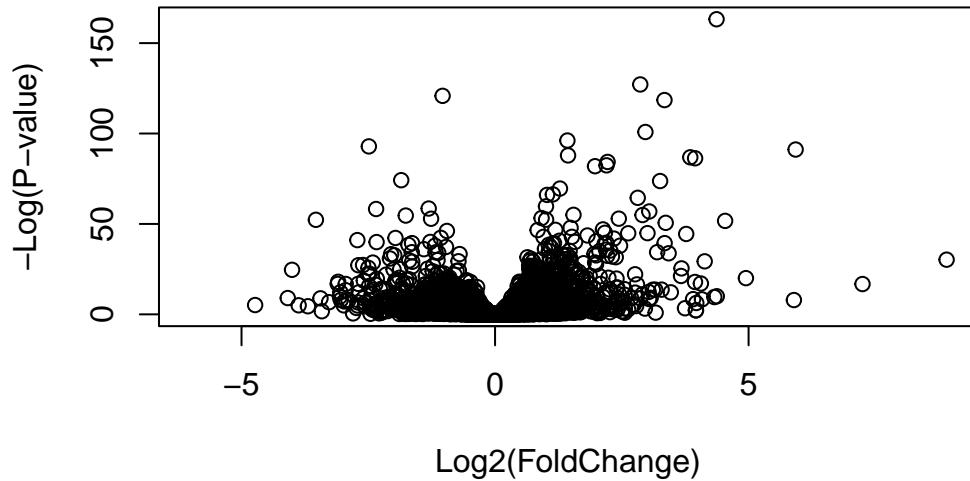
	baseMean	log2FoldChange	lfcSE	stat	pvalue
	<numeric>	<numeric>	<numeric>	<numeric>	<numeric>
ENSG00000152583	954.771	4.36836	0.2371306	18.4217	8.79214e-76
ENSG00000179094	743.253	2.86389	0.1755659	16.3123	8.06568e-60
ENSG00000116584	2277.913	-1.03470	0.0650826	-15.8983	6.51317e-57
ENSG00000189221	2383.754	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	AOA8Q3SIN5	
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			

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

##Data Visualization

Volcano Plots

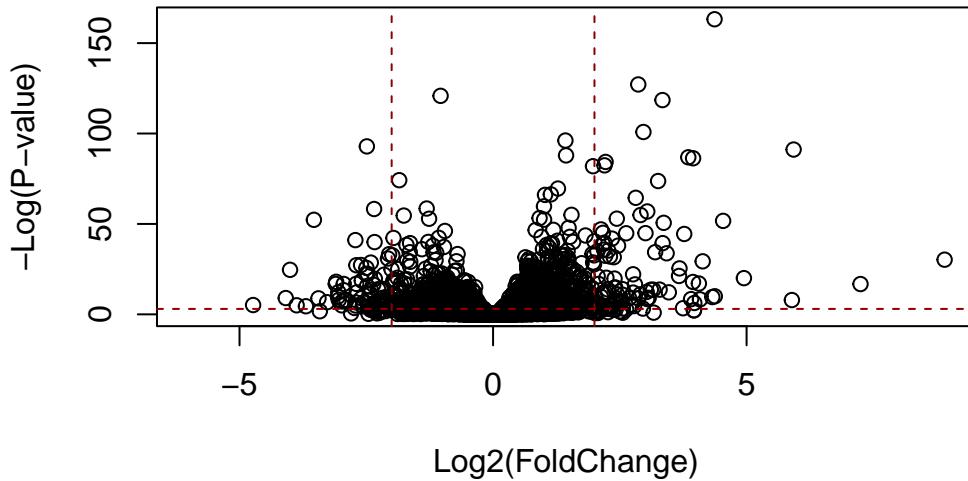
```
plot( res$log2FoldChange, -log(res$padj),
      xlab="Log2(FoldChange)",
      ylab="-Log(P-value)")
```



Adding lines that indicate where a p-value of 0.05 is and where the fold change is 2.

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

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



```

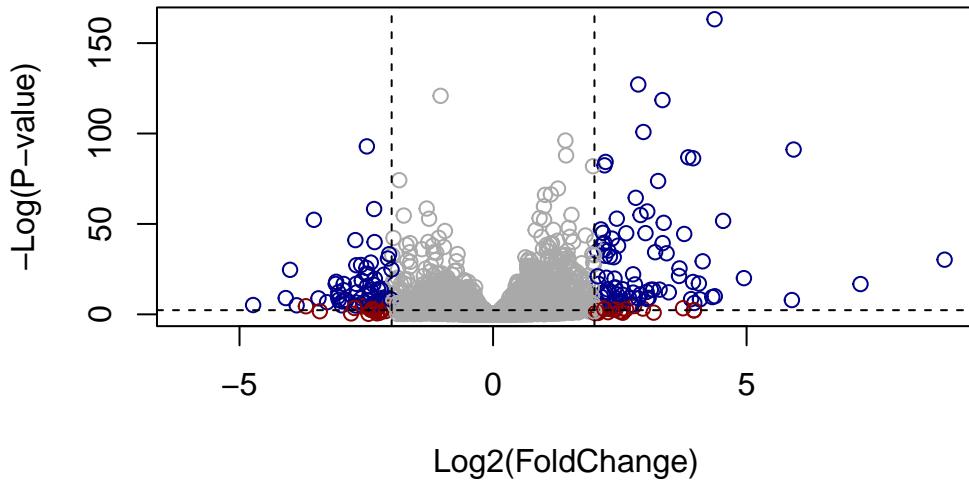
# Setup our custom point color vector
mycols <- rep("darkgrey", nrow(res))
mycols[ abs(res$log2FoldChange) > 2 ] <- "darkred"

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

# 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="black", lty=2)
abline(h=-log(0.1), col="black", lty=2)

```



Enhanced Volcano

```
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.

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.

Volcano plot

EnhancedVolcano

