

# **Lab\_12**

Runqi Zhang

## **1. Bioconductor and DESeq2 setup**

```
library(BiocManager)
```

```
Warning: package 'BiocManager' was built under R version 4.2.2
```

```
Bioconductor version '3.15' is out-of-date; the current release version '3.16'  
is available with R version '4.2'; see https://bioconductor.org/install
```

```
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, 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 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
```

```
Warning: package 'matrixStats' was built under R version 4.2.2
```

```
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
```

## 2. Import countData and colData

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

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

sum(metadata$dex=="control")

[1] 4
```

**Q1. How many genes are in this dataset?**

A: 38694

**Q2. How many ‘control’ cell lines do we have?**

A: 4

```
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$id)
n_control <- sum(metadata$dex=="control")
```

```

control.mean <- rowSums(control.counts)/n_control
head(control.mean)

ENSG000000000003 ENSG000000000005 ENSG000000000419 ENSG000000000457 ENSG000000000460
         900.75          0.00        520.50        339.75        97.25
ENSG000000000938
         0.75

```

**Q3. How would you make the above code in either approach more robust?**

A: substitute 4 with “n\_control <- sum(metadata\$dex==”control”)“, so the function would still give the correct values when more samples are added.

**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)**

A: See codes below

```

treated <- metadata %>% filter(dex=="treated")
treated.counts <- counts %>% select(treated$id)
n_treated <- sum(metadata$dex=="treated")
treated.mean <- rowSums(treated.counts)/n_treated
head(treated.mean)

ENSG000000000003 ENSG000000000005 ENSG000000000419 ENSG000000000457 ENSG000000000460
         658.00          0.00        546.00        316.50        78.75
ENSG000000000938
         0.00

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

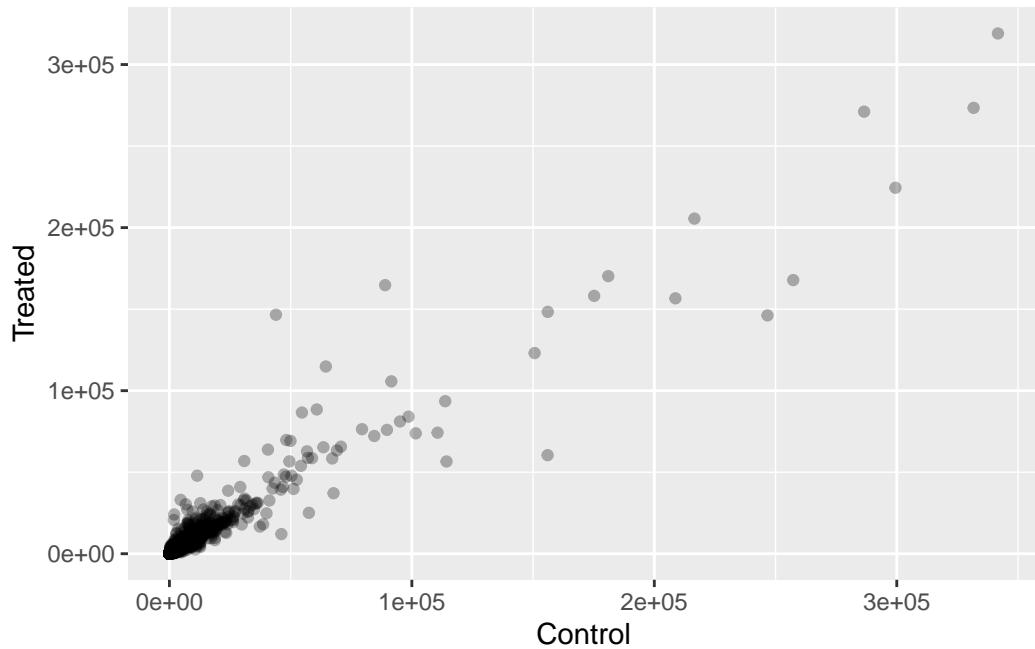
control.mean treated.mean
23005324      22196524

```

**Q5 (a). Create a scatter plot showing the mean of the treated samples against the mean of the control samples. Your plot should look something like the following.**

A: see codes below

```
library(ggplot2)
plot_1 <- ggplot(meancounts) +
  aes(control.mean, treated.mean) +
  geom_point(alpha=0.3) +
  labs(x="Control",
       y="Treated")
plot_1
```



**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?**

A: geom\_point(alpha=0.3)

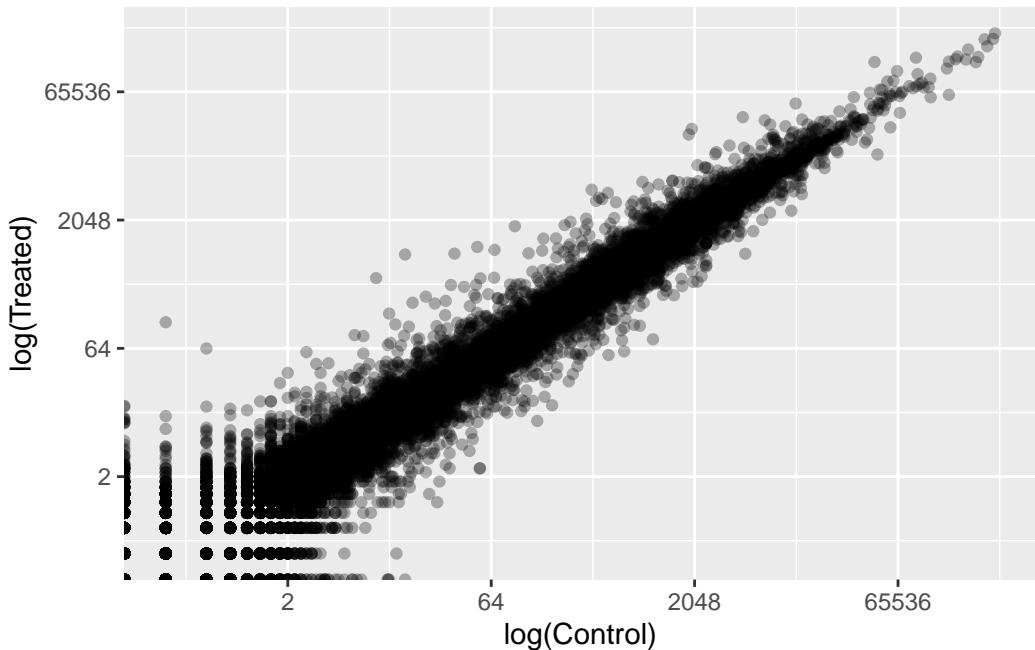
**Q6. Try plotting both axes on a log scale. What is the argument to plot() that allows you to do this?**

A: scale\_x\_log10() or scale\_x\_continuous(trans="log2").

```
plot_1+  
  scale_x_continuous(trans="log2") +  
  scale_y_continuous(trans="log2") +  
  labs(x = "log(Control)",  
       y = "log(Treated)")
```

Warning: Transformation introduced infinite values in continuous x-axis

Warning: Transformation introduced infinite values in continuous y-axis



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

	control.mean	treated.mean	log2fc
ENSG00000000003	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

The NaN is returned when you divide by zero and try to take the log. The -Inf is returned when you try to take the log of zero

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

	row	col
ENSG000000000005	2	1
ENSG00000004848	65	1
ENSG00000004948	70	1
ENSG00000005001	73	1
ENSG00000006059	121	1
ENSG00000006071	123	1

```
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
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?**

A: “arr.ind=TRUE” makes the which() function to return both col and row indices (it tells us which genes in which samples have zero count); We are removing all genes with zero count from our dataset, therefore we only focus on the row, where the gene data is found, but not the column. The unique() function will delete any duplicate values or rows.

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

sum(up.ind)
```

[1] 250

```
sum(down.ind)
```

[1] 367

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

A: 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?**

A: 367

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

A: No, these resultst are not trust-worthy because the fold-change can be large without being biologically significant.

## 4. DESeq2 analysis

```
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
```

```

  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
ENSG000000000005     NA
ENSG000000000419   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
```

## 5. 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    -0.3507030  0.168246 -2.084470 0.0371175
ENSG000000000005  0.000000        NA        NA        NA        NA
ENSG000000000419 520.134160     0.2061078  0.101059  2.039475 0.0414026
ENSG000000000457 322.664844     0.0245269  0.145145  0.168982 0.8658106
ENSG000000000460 87.682625     -0.1471420  0.257007 -0.572521 0.5669691
ENSG000000000938 0.319167     -1.7322890  3.493601 -0.495846 0.6200029
  padj      symbol
  <numeric> <character>
ENSG000000000003 0.163035      TSPAN6
ENSG000000000005  NA          TNMD
ENSG000000000419 0.176032      DPM1
ENSG000000000457 0.961694      SCYL3
ENSG000000000460 0.815849      C1orf112
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.**

A: see codes below

```

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
ENSG000000000419	520.134160	0.2061078	0.101059	2.039475	0.0414026
ENSG000000000457	322.664844	0.0245269	0.145145	0.168982	0.8658106
ENSG000000000460	87.682625	-0.1471420	0.257007	-0.572521	0.5669691
ENSG000000000938	0.319167	-1.7322890	3.493601	-0.495846	0.6200029
	padj	symbol	entrez	uniprot	
	<numeric>	<character>	<character>	<character>	
ENSG000000000003	0.163035	TSPAN6	7105	AOA024RC10	
ENSG000000000005	NA	TNMD	64102	Q9H2S6	
ENSG000000000419	0.176032	DPM1	8813	060762	
ENSG000000000457	0.961694	SCYL3	57147	Q8IZE3	
ENSG000000000460	0.815849	C1orf112	55732	AOA024R922	
ENSG000000000938	NA	FGR	2268	P09769	
	genename				
	<character>				
ENSG000000000003	tetraspanin 6				
ENSG000000000005	tenomodulin				
ENSG000000000419	dolichyl-phosphate m..				
ENSG000000000457	SCY1 like pseudokina..				
ENSG000000000460	chromosome 1 open re..				
ENSG000000000938	FGR proto-oncogene, ..				

```

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

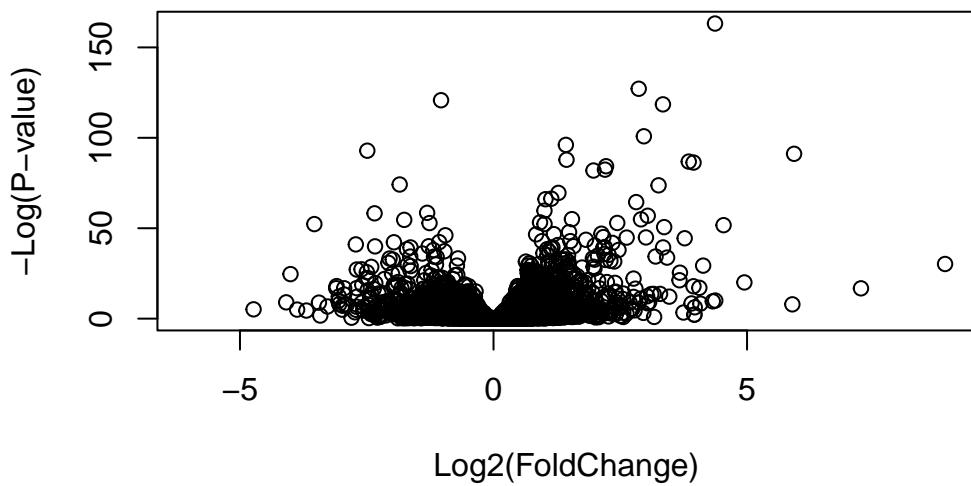
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 6 rows and 10 columns
      baseMean log2FoldChange     lfcSE      stat      pvalue
      <numeric>      <numeric> <numeric> <numeric>    <numeric>
ENSG00000152583   954.771       4.36836  0.2371268   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  015534
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")

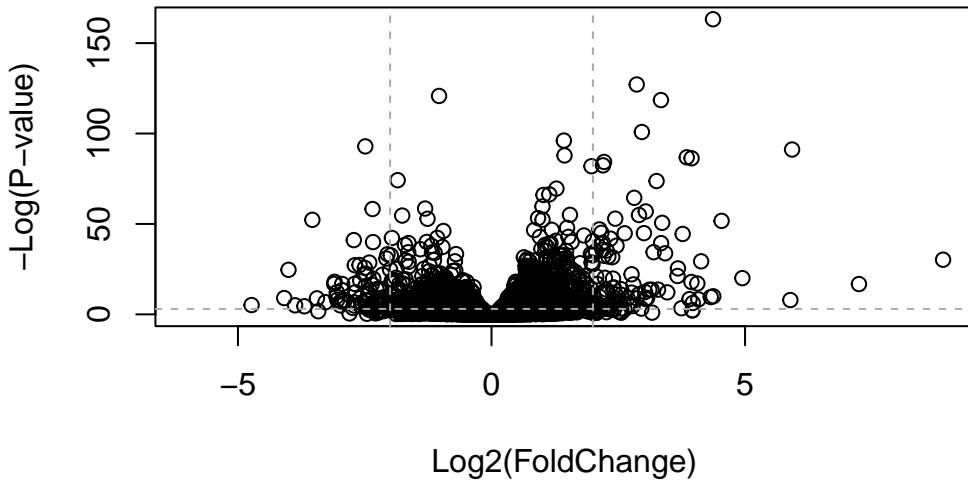
```

## 6. 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)
```



```

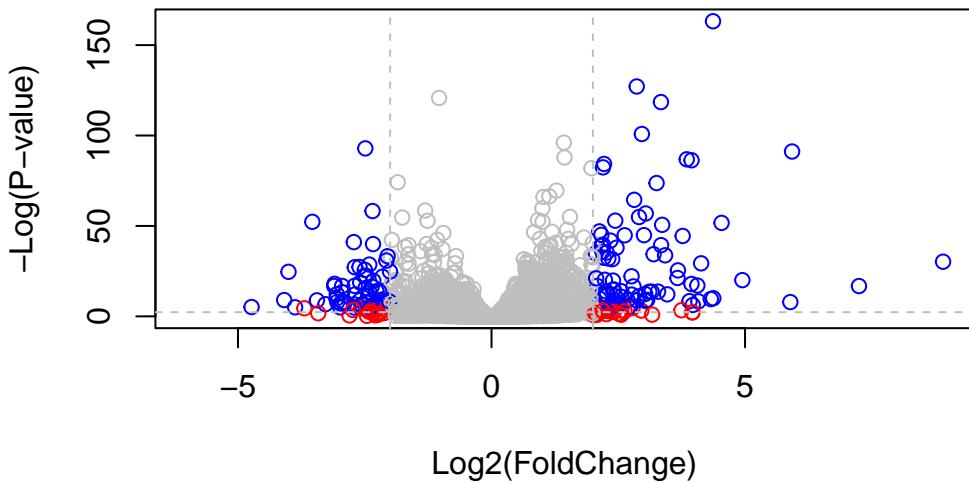
# Setup our custom point color vector
mycols <- rep("gray", nrow(res))
mycols[ abs(res$log2FoldChange) > 2 ] <- "red"

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)

```



```
library(EnhancedVolcano)
```

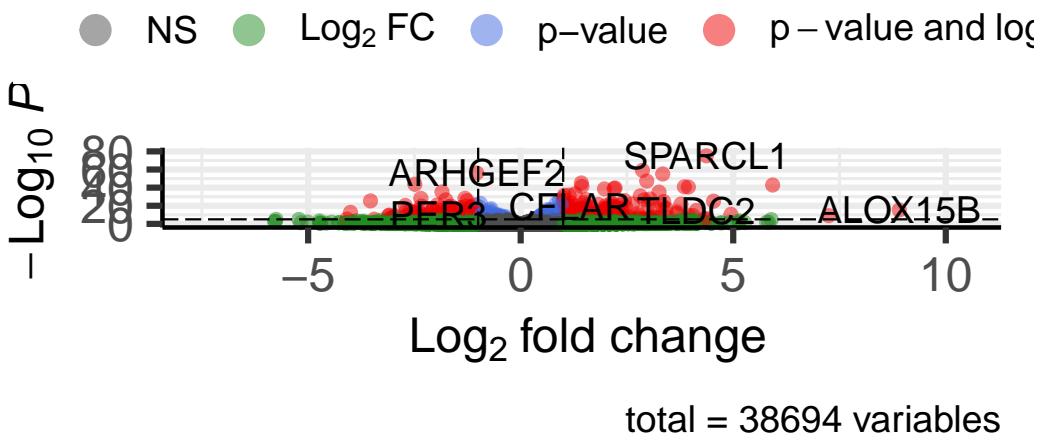
```
Loading required package: ggrepel
```

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

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

## Volcano plot

*Enhanced Volcano*



## 7. Pathway analysis

```
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 C:/Users/zhang/OneDrive/ /BIMM 143 - Bioinformatics Lab/Class\_12

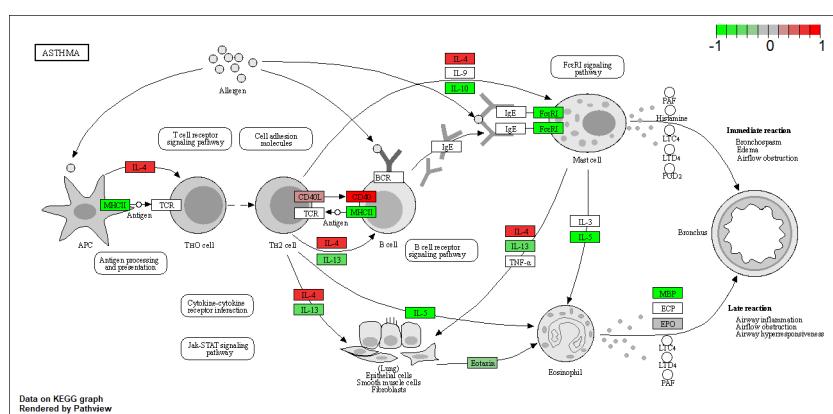
Info: Writing image file hsa05310.pathview.png

```
# A different PDF based output of the same data
pathview(gene.data=foldchanges, pathway.id="hsa05310", kegg.native=FALSE)
```

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

Info: Working in directory C:/Users/zhang/OneDrive/ /BIMM 143 - Bioinformatics Lab/Class\_12

Info: Writing image file hsa05310.pathview.pdf



**Q12. Can you do the same procedure as above to plot the pathview figures for the top 2 down-regulated pathways?**

A: see codes below

This is for the most significantly down-regulated pathway

```
pathview(gene.data=foldchanges, pathway.id="hsa05332")
```

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

Info: Working in directory C:/Users/zhang/OneDrive/ /BIMM 143 - Bioinformatics Lab/Class\_12

Info: Writing image file hsa05332.pathview.png

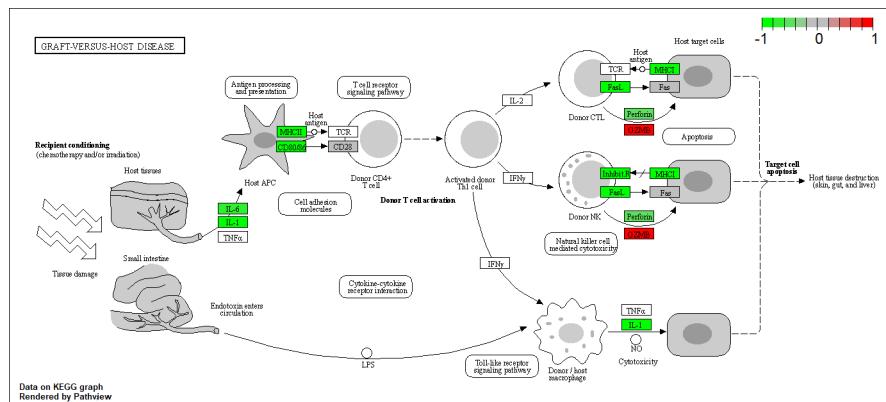
```
# A different PDF based output of the same data
pathview(gene.data=foldchanges, pathway.id="hsa05332", kegg.native=FALSE)
```

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

Warning in .subtypeDisplay(object): Given subtype 'missing interaction' is not found!

Info: Working in directory C:/Users/zhang/OneDrive/ /BIMM 143 - Bioinformatics Lab/Class\_12

Info: Writing image file hsa05332.pathview.pdf



This is for the second most significant down-regulated pathway

```
pathview(gene.data=foldchanges, pathway.id="hsa04940")
```

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

Info: Working in directory C:/Users/zhang/OneDrive/ /BIMM 143 - Bioinformatics Lab/Class\_12

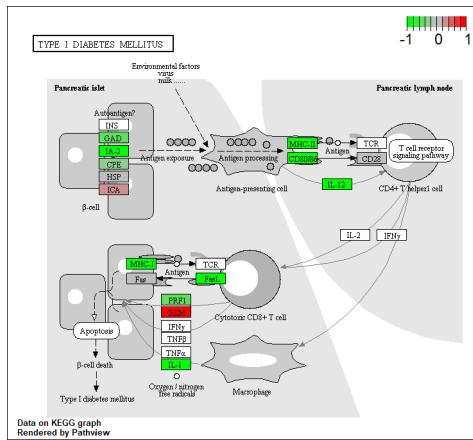
Info: Writing image file hsa04940.pathview.png

```
# A different PDF based output of the same data  
pathview(gene.data=foldchanges, pathway.id="hsa04940", kegg.native=FALSE)
```

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

Info: Working in directory C:/Users/zhang/OneDrive/ /BIMM 143 - Bioinformatics Lab/Class\_12

Info: Writing image file hsa04940.pathview.pdf



## OPTIONAL: Plotting counts for genes of interest

```
i <- grep("CRISPLD2", res$symbol)  
res[i,]
```

```

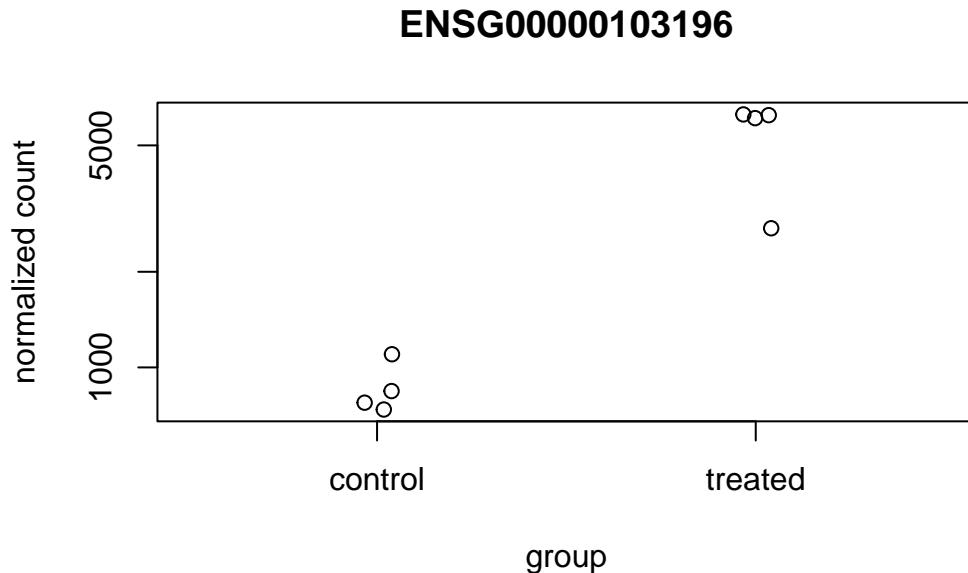
log2 fold change (MLE): dex treated vs control
Wald test p-value: dex treated vs control
DataFrame with 1 row and 10 columns
  baseMean log2FoldChange      lfcSE      stat      pvalue
  <numeric>      <numeric> <numeric> <numeric>    <numeric>
ENSG00000103196   3096.16       2.62603  0.267444  9.81899 9.32747e-23
  padj      symbol      entrez      uniprot
  <numeric> <character> <character> <character>
ENSG00000103196 3.36344e-20    CRISPLD2      83716 AOA140VK80
  genename
  <character>
ENSG00000103196 cysteine rich secret..

```

```
rownames(res[i,])
```

```
[1] "ENSG00000103196"
```

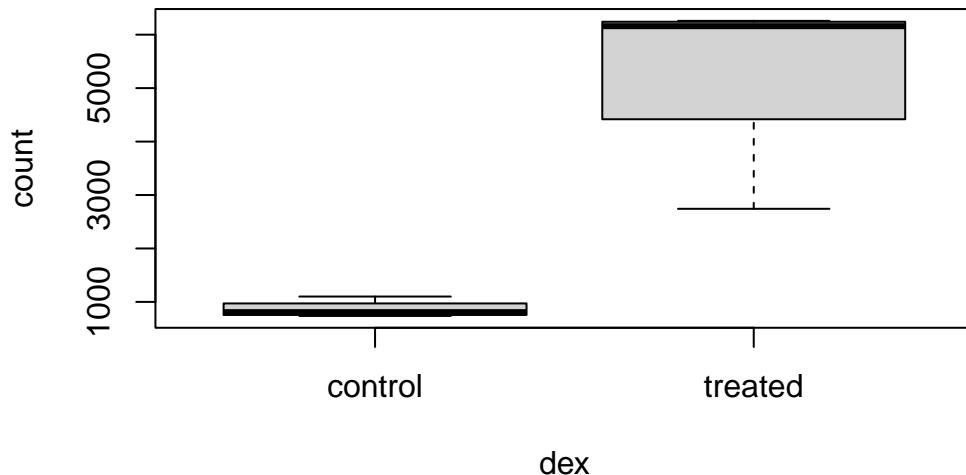
```
plotCounts(dds, gene="ENSG00000103196", intgroup="dex")
```



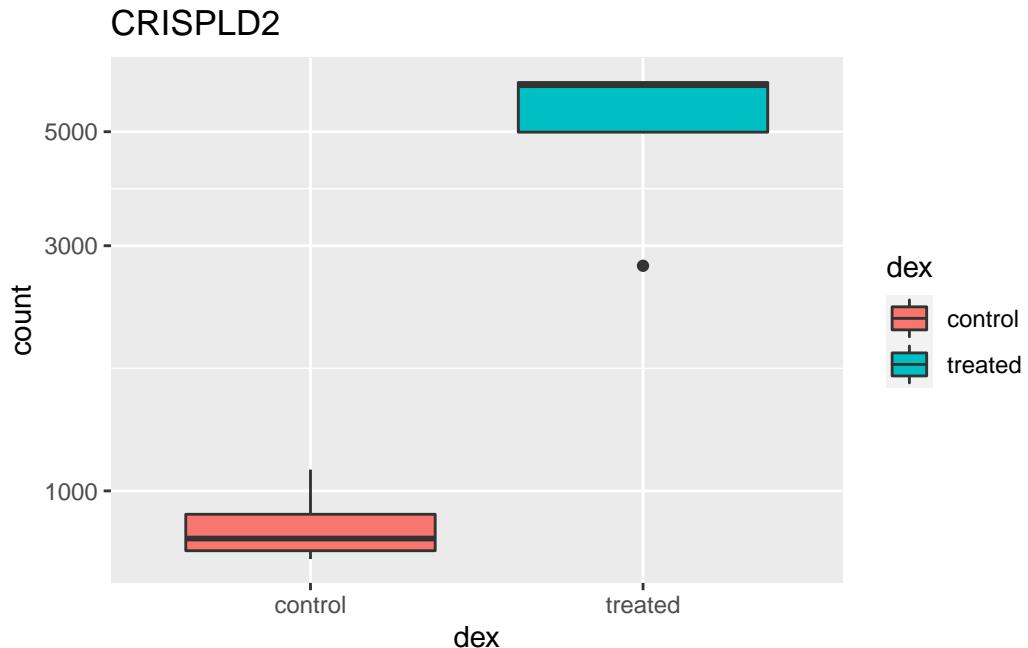
```
# Return the data
d <- plotCounts(dds, gene="ENSG00000103196", intgroup="dex", returnData=TRUE)
head(d)
```

	count	dex
SRR1039508	774.5002	control
SRR1039509	6258.7915	treated
SRR1039512	1100.2741	control
SRR1039513	6093.0324	treated
SRR1039516	736.9483	control
SRR1039517	2742.1908	treated

```
boxplot(count ~ dex , data=d)
```



```
library(ggplot2)
ggplot(d, aes(dex, count, fill=dex)) +
  geom_boxplot() +
  scale_y_log10() +
  ggtitle("CRISPLD2")
```



## Session Info

```
sessionInfo()

R version 4.2.1 (2022-06-23 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 22000)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

attached base packages:
[1] stats4      stats       graphics   grDevices  utils      datasets   methods
[8] base
```

other attached packages:

```
[1] gageData_2.34.0          gage_2.46.1
[3] pathview_1.36.1          EnhancedVolcano_1.14.0
[5] ggrepel_0.9.1            org.Hs.eg.db_3.15.0
[7] AnnotationDbi_1.58.0    ggplot2_3.3.6
[9] dplyr_1.0.10             DESeq2_1.36.0
[11] SummarizedExperiment_1.26.1 Biobase_2.56.0
[13] MatrixGenerics_1.8.1    matrixStats_0.62.0
[15] GenomicRanges_1.48.0    GenomeInfoDb_1.32.4
[17] IRanges_2.30.1           S4Vectors_0.34.0
[19] BiocGenerics_0.42.0     BiocManager_1.30.19
```

loaded via a namespace (and not attached):

```
[1] httr_1.4.4                bit64_4.0.5          jsonlite_1.8.3
[4] splines_4.2.1              assertthat_0.2.1      blob_1.2.3
[7] GenomeInfoDbData_1.2.8    yaml_2.3.6            pillar_1.8.1
[10] RSQLite_2.2.18             lattice_0.20-45       glue_1.6.2
[13] digest_0.6.30              RColorBrewer_1.1-3    XVector_0.36.0
[16] colorspace_2.0-3           htmltools_0.5.3        Matrix_1.5-1
[19] XML_3.99-0.12             pkgconfig_2.0.3        genefilter_1.78.0
[22] zlibbioc_1.42.0            GO.db_3.15.0          xtable_1.8-4
[25] scales_1.2.1               BiocParallel_1.30.4   tibble_3.1.8
[28] annotate_1.74.0            KEGGREST_1.36.3       farver_2.1.1
[31] generics_0.1.3              withr_2.5.0            cachem_1.0.6
[34] cli_3.4.1                 survival_3.4-0         magrittr_2.0.3
[37] crayon_1.5.2              KEGGgraph_1.56.0       memoise_2.0.1
[40] evaluate_0.17              fansi_1.0.3            graph_1.74.0
[43] tools_4.2.1                lifecycle_1.0.3        stringr_1.4.1
[46] locfit_1.5-9.6             munsell_0.5.0          DelayedArray_0.22.0
[49] Biostrings_2.64.1           compiler_4.2.1         rlang_1.0.6
[52] grid_4.2.1                 RCurl_1.98-1.9         rstudioapi_0.14
[55] labeling_0.4.2              bitops_1.0-7           rmarkdown_2.17
[58] gtable_0.3.1               codetools_0.2-18       DBI_1.1.3
[61] R6_2.5.1                  knitr_1.40             fastmap_1.1.0
[64] bit_4.0.4                  utf8_1.2.2             Rgraphviz_2.40.0
[67] stringi_1.7.8              parallel_4.2.1         Rcpp_1.0.9
[70] vctrs_0.5.0                geneplotter_1.74.0    png_0.1-7
[73] tidyselect_1.2.0            xfun_0.34
```