MA-plot example

```
library(tidyverse)
library(BiocManager)
library(airway)
library(DESeq2)
```

The Data

The data used is in the airway package. The data is from an RNA-seq experiment where a glucocorticoid steroid, dexamethasone, is used to treat smooth muscle cells.

```
data(airway)
airway
```

```
## class: RangedSummarizedExperiment
## dim: 64102 8
## metadata(1): ''
## assays(1): counts
## rownames(64102): ENSG00000000003 ENSG00000000005 ... LRG_98 LRG_99
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(9): SampleName cell ... Sample BioSample
```

```
#change to dataframe
sample_info <- as.data.frame(colData(airway))
#columns needed are the cell and dex column
sample_info <- sample_info[,c(2,3)]
#need to change values in dex column
sample_info$dex <- gsub('trt', 'treated', sample_info$dex)
sample_info$dex <- gsub('untrt', 'untreated', sample_info$dex)
names(sample_info) <- c('cellLine', 'dexamethasone')

#Write to csv files
write.table(sample_info, file = "sample_info.csv", sep = ',', col.names = T, row.names = T, quote = F)
countsData <- assay(airway)
write.table(countsData, file = "counts_data.csv", sep = ',', col.names = T, row.names = T, quote = F)
```

```
counts_data <- read.csv('counts_data.csv')

head(counts_data)
```

```
##                 SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003        679        448        873        408       1138
## ENSG00000000005          0          0          0          0          0
```

```
## ENSG00000000419        467        515        621        365        587
## ENSG00000000457        260        211        263        164        245
## ENSG00000000460         60         55         40         35         78
## ENSG00000000938          0          0          2          0          1
##                 SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003       1047        770        572
## ENSG00000000005          0          0          0
## ENSG00000000419        799        417        508
## ENSG00000000457        331        233        229
## ENSG00000000460         63         76         60
## ENSG00000000938          0          0          0
```

Looking at the dataset

The rows are the Gene IDs and the columns are the sample names. But we need to know the circumstances of the samples: treated or untreated

```
colData <- read.csv('sample_info.csv')

colData
```

```
##            cellLine dexamethasone
## SRR1039508   N61311     untreated
## SRR1039509   N61311       treated
## SRR1039512  N052611     untreated
## SRR1039513  N052611       treated
## SRR1039516  N080611     untreated
## SRR1039517  N080611       treated
## SRR1039520  N061011     untreated
## SRR1039521  N061011       treated
```

It is important to be sure the the samples match and are in the same order in counts_data and colData

```
all(colnames(counts_data) %in% rownames(colData))
```

```
## [1] TRUE
```

```
all(colnames(counts_data) == rownames(colData))
```

```
## [1] TRUE
```

Create the DESeqDataSet object using DESeqDataSetFromMatirx

In the process of creating the DESeqDataset object, I also remove rows with low gene counts, here I chose to keep genes witha total of at least 10 reads. This filtering process changed the number of genes from 64102 to 22369.

```
dds <- DESeqDataSetFromMatrix(countData = counts_data,
                       colData = colData,
                       design  = ~ dexamethasone)


keep <- rowSums(counts(dds)) >= 10

dds <- dds[keep,]

dds
```

```
## class: DESeqDataSet
## dim: 22369 8
## metadata(1): version
## assays(1): counts
## rownames(22369): ENSG00000000003 ENSG00000000419 ... ENSG00000273487
##   ENSG00000273488
## rowData names(0):
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(2): cellLine dexamethasone
```

Now there needs to be a factor level set as there are two different types of samples, treated and untreated

```
dds$dexamethasone <- relevel(dds$dexamethasone, ref = 'untreated')

dds$dexamethasone
```

```
## [1] untreated treated   untreated treated   untreated treated   untreated
## [8] treated
## Levels: untreated treated
```

Running DESeq2

```
dds <- DESeq(dds)
```

```
## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
results_dds <- results(dds)

head(results_dds)
```

```
## log2 fold change (MLE): dexamethasone treated vs untreated
## Wald test p-value: dexamethasone treated vs untreated
## DataFrame with 6 rows and 6 columns
##                   baseMean log2FoldChange     lfcSE      stat    pvalue
##                  <numeric>      <numeric> <numeric> <numeric> <numeric>
## ENSG00000000003  708.5979     -0.3788229  0.173155 -2.187769 0.0286865
## ENSG00000000419  520.2963      0.2037893  0.100742  2.022878 0.0430857
## ENSG00000000457  237.1621      0.0340631  0.126476  0.269325 0.7876795
## ENSG00000000460   57.9324     -0.1171564  0.301583 -0.388472 0.6976669
## ENSG00000000971 5817.3108      0.4409793  0.258776  1.704099 0.0883626
## ENSG00000001036 1282.1007     -0.2419097  0.119713 -2.020751 0.0433055
##                       padj
##                  <numeric>
## ENSG00000000003  0.138470
## ENSG00000000419  0.182998
## ENSG00000000457  0.929805
## ENSG00000000460  0.894231
## ENSG00000000971  0.297042
## ENSG00000001036  0.183498
```

Explanation of the columns

baseMean - is the average of the normalized counts taken over all the samples

log2FoldChange - is the change of the gene in the treated condition compared with the untreated; thus the positive values are upregulated and negative values are down regulated in the treated condition. Note: that the values are in reference to the treated being compared to the untreated in this example.

lfcSE- is the standard error of the log2FoldChange

Stat - the Wald Test Values

pvalue - the p-value from the Wald Test

padj - the p adjusted value for multiple testing

```
summary(results_dds)
```

```
##
## out of 22369 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 1884, 8.4%
## LFC < 0 (down)     : 1502, 6.7%
## outliers [1]       : 51, 0.23%
## low counts [2]     : 3903, 17%
## (mean count < 4)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

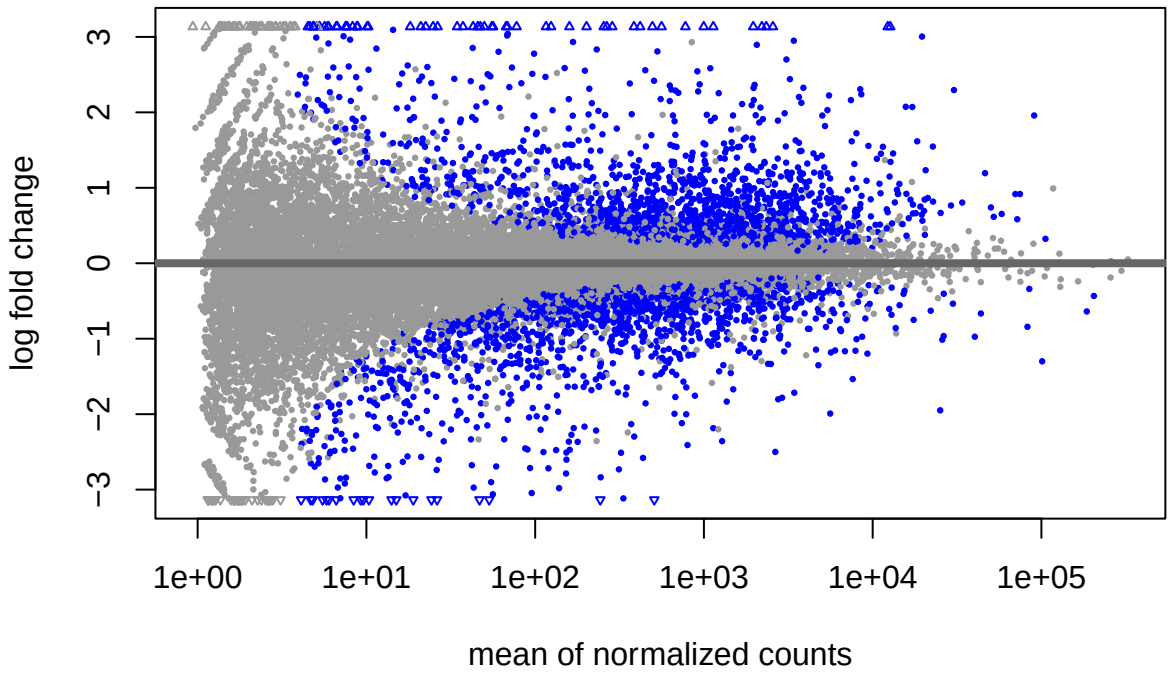
```
summary(results(dds, alpha =  0.01))
```

```
##
## out of 22369 with nonzero total read count
## adjusted p-value < 0.01
## LFC > 0 (up)       : 1030, 4.6%
## LFC < 0 (down)     : 708, 3.2%
## outliers [1]       : 51, 0.23%
## low counts [2]     : 5200, 23%
## (mean count < 6)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Viewing the Summary

The summary function shows the percentage of upregulated and downregulated genes.

```
plotMA(results_dds)
```



Explanation and Interpretation of the MA-plot

The Y-axis showing the change in gene expression from the two conditions. The X-axis shows the normalized expression counts. The dots in blue are the genes that are significantly differentially expressed, that have adjusted p-values of 0.05.

The genes are in the upper right or lower right are possible candidate genes for further infestigation because they would have a high mean of normalized counts and a high log2foldchange.