

## VI. Bioinformatics in R (presentation)

12 May, 2022

### Bioconductor

Bioconductor provides tools for computational biology and bioinformatics analysis in R - it is open source and open development and it has an active user community.

Mostly when we install R-packages we use `install.packages('name_of_package')`. When we use this command we refer to the CRAN repository of packages, however sometimes we want a package from **Bioconductor** instead. For this we use the command `BiocManager::install('name_of_package')`. In order to use this installer, you need to download the R-package **BiocManager** e.g. `install.packages('BiocManager')`.

### Gene Expression Analysis in R with DEseq2

DEseq2 is one of the many packages/frameworks which exists for analysis of bulk gene expression data in R. For more information on DEseq2, please have a look at the original publication [here](#).

Other highly used packages for differential expression analysis *DEA* are:

- limma
- edgeR
- NOIseq

DEseq2 has many advantages over classical models and post hoc tests, as it is specifically developed for handling common issues and biases in expression data, including differences in sequencing depth and highly variable dispersion of counts between genes.

In brief, **DEseq2** fits a generalized linear model (GLM) for each gene in the dataset. In the case where we compare two groups i.e. treatment vs control, the GLM fit returns coefficients indicating the overall expression strength of a gene, along with the log2 fold change between groups. **DEseq2** adjusts variable gene dispersion estimates using an empirical Bayes approach which borrows information across genes and shrinks gene-wise dispersions towards a common dispersion trend to increase accuracy of differential expression testing.

---

### About the Dataset

The dataset used for this presentation was acquired from the following github tutorial on RNAseq analysis: <https://combine-australia.github.io/RNAseq-R/06-rnaseq-day1.html>.

RNA sequencing data generated from luminal and basal cell sub-populations in the mammary gland of three groups of mice:

- Control
- Pregnant
- Lactating

The objective of the original study (found [here](#)) was to identify genes specifically expressed in lactating mammary glands, the gene expression profiles of luminal and basal cells from different developmental stages were compared.

---

## Load R-packages:

```
# Data Wrangling
# install.packages("tidyverse")
# install.packages("readxl")
library(tidyverse)
library(readxl)

# For Plotting
# install.packages("ggplot2")
library(ggplot2)

# For DEA
# install.packages("BiocManager")
# BiocManager::install("DESeq2")
library(DESeq2)
library(dplyr)
```

---

## Importing Data

Reading in data:

```
exprDat <- read_excel("MouseRNAseq.xlsx")
exprInfo <- read_excel("MouseSampleInfo.xlsx")
```

```
# Look at the data:
head(exprDat, n=5)
```

```
## # A tibble: 5 x 13
##   GeneName MCL1.DG MCL1.DH MCL1.DI MCL1.DJ MCL1.DK MCL1.DL MCL1.LA MCL1.LB
##   <chr>      <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Xkr4        438    300    65    237    354    287     0     0
## 2 Rp1           1     1     0     0     0     0    10     3
## 3 Sox17       106    182    82    105    43    82    16    25
## 4 Mrpl15      309    234   337    300    290    270   560   464
## 5 Lypla1      652    515   948    935    928    791   826   862
## # ... with 4 more variables: MCL1.LC <dbl>, MCL1.LD <dbl>, MCL1.LE <dbl>,
## #   MCL1.LF <dbl>
```

```
dim(exprDat)
```

```
## [1] 23151    13
```

```
head(exprInfo)
```

```
## # A tibble: 6 x 4
##   SampleName CellType Status CellType.colors
##   <chr>      <chr>  <chr>    <chr>
## 1 MCL1.DG    basal   control #79ADDC
## 2 MCL1.DH    basal   control #79ADDC
```

```
## 3 MCL1.DI      basal    pregnant #79ADDC
## 4 MCL1.DJ      basal    pregnant #79ADDC
## 5 MCL1.DK      basal    lactate  #79ADDC
## 6 MCL1.DL      basal    lactate  #79ADDC
```

Convert character columns to factor types:

```
exprInfo <- exprInfo %>%
  mutate(CellType = as.factor(CellType),
         Status = as.factor(Status))

head(exprInfo)
```

```
## # A tibble: 6 x 4
##   SampleName CellType Status   CellType.colors
##   <chr>      <fct>   <fct>   <chr>
## 1 MCL1.DG    basal    control #79ADDC
## 2 MCL1.DH    basal    control #79ADDC
## 3 MCL1.DI    basal    pregnant #79ADDC
## 4 MCL1.DJ    basal    pregnant #79ADDC
## 5 MCL1.DK    basal    lactate  #79ADDC
## 6 MCL1.DL    basal    lactate  #79ADDC
```

## Initial Data Check & Filtering:

Let's try to sample 16 (n) random genes and plot their count distribution.

```
# Sample 16 random rows
expr16 <- exprDat %>%
  sample_n(.,16)
```

```
expr16
```

```
## # A tibble: 16 x 13
##   GeneName      MCL1.DG MCL1.DH MCL1.DI MCL1.DJ MCL1.DK MCL1.DL MCL1.LA MCL1.LB
##   <chr>        <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 Nlrp5-ps         8       6       8       6       6       8       4       7
## 2 Baz2a          2348     1929    1458    1334    1523    1647    2940    3076
## 3 Serpina1d        5      10       3       2       3       1       4       0
## 4 Rsl24d1         426     392     469     317     187     170     563     544
## 5 Krtap3-2         0       1       0       0       0       1       0       0
## 6 Nudt3          1586    1530    2182    1947    1550    1462    1124    1365
## 7 Ptpn22          17      20      49      42      20      11      13      36
## 8 Mmab           167     151     200     159     341     353     154     131
## 9 Zan            19      15      28      13      15       5       4       6
## 10 5430403N17Rik   4       5       6       1       5       4       3       1
## 11 Vmn1r169        0       0       0       0       0       0       0       0
## 12 Ptgs2os         0       0       0       0       2       1       0       0
## 13 Sh2b3          940     915    1068     907     558     579     420     503
## 14 A26c3          0       0       0       0       0       0       0       0
## 15 Olfr1425        0       0       0       0       0       0       0       0
## 16 Brsk1          559     477     483     420     548     551     122     136
## # ... with 4 more variables: MCL1.LC <dbl>, MCL1.LD <dbl>, MCL1.LE <dbl>,
```

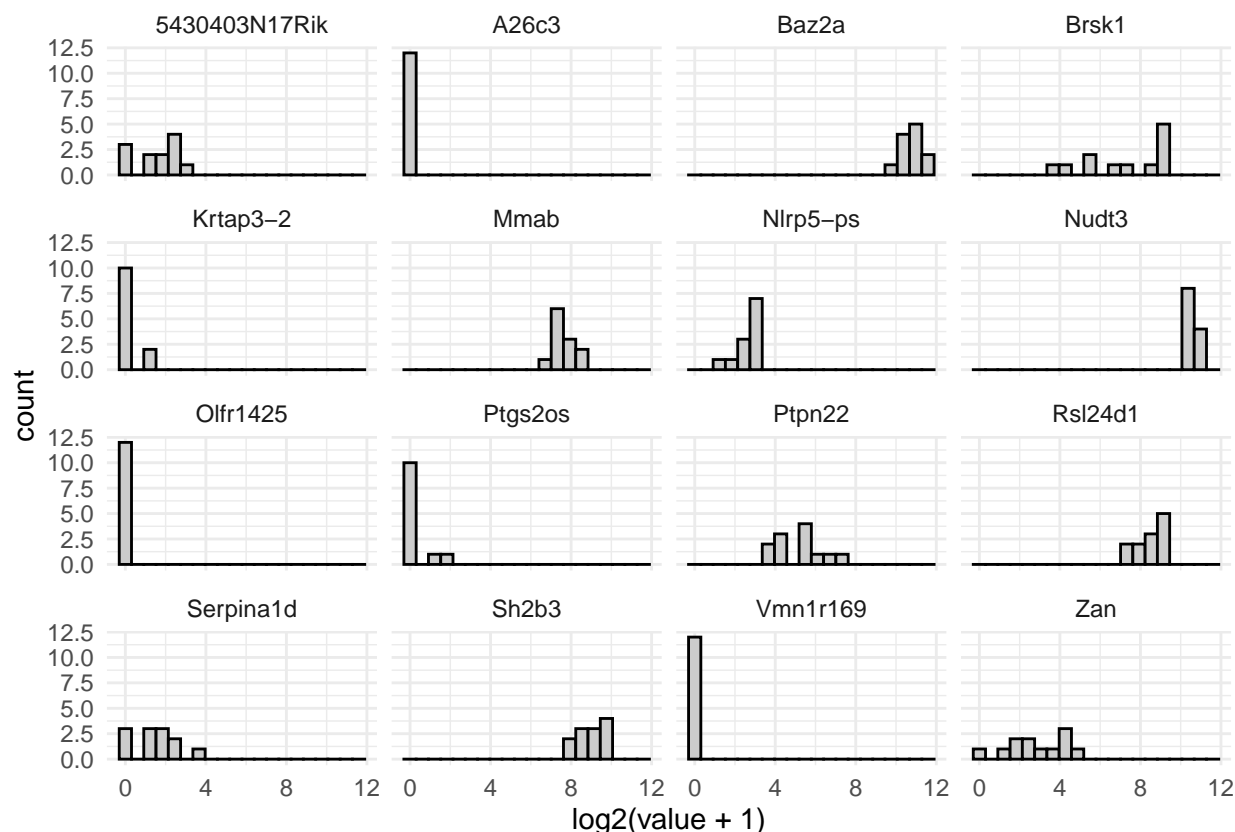
```
## # MCL1.LF <dbl>
# Gather counts
# Gather counts
expr16 <- expr16 %>%
  column_to_rownames(var = "GeneName") %>%
  t() %>%
  as_tibble() %>%
  gather()

# Give it a look:
expr16
```

```
## # A tibble: 192 x 2
##   key      value
##   <chr>    <dbl>
## 1 Nlrp5-ps      8
## 2 Nlrp5-ps      6
## 3 Nlrp5-ps      8
## 4 Nlrp5-ps      6
## 5 Nlrp5-ps      6
## 6 Nlrp5-ps      8
## 7 Nlrp5-ps      4
## 8 Nlrp5-ps      7
## 9 Nlrp5-ps      4
## 10 Nlrp5-ps     3
## # ... with 182 more rows
```

Plot:

```
ggplot(expr16, aes(log2(value+1))) +
  geom_histogram(color="black", fill="grey80", bins=20) +
  theme_minimal() +
  facet_wrap(~key)
```



We will filter out low expressed genes. There are many strategies for doing so, but here we will filter out genes that have less than 3 counts in at least  $n$  samples. We select  $n$  as the smallest number of biologically meaningful groups. In this case, it is 3.

```
table(exprInfo$CellType, exprInfo$Status)
```

```
##
##          control lactate pregnant
##   basal         2         2         2
##   luminal        2         2         2
```

```
# 2 samples in each group
```

```
# Count number of samples with min. count size of 4 for a given gene.
# Filter for genes were min. 4 samples have a count equal to or greater than 4.
```

```
exprDat <- exprDat %>%
  mutate(ncount = rowSums(dplyr::select(., -GeneName) >= 4)) %>%
  filter(ncount >= 4) %>%
  dplyr::select(-ncount)
```

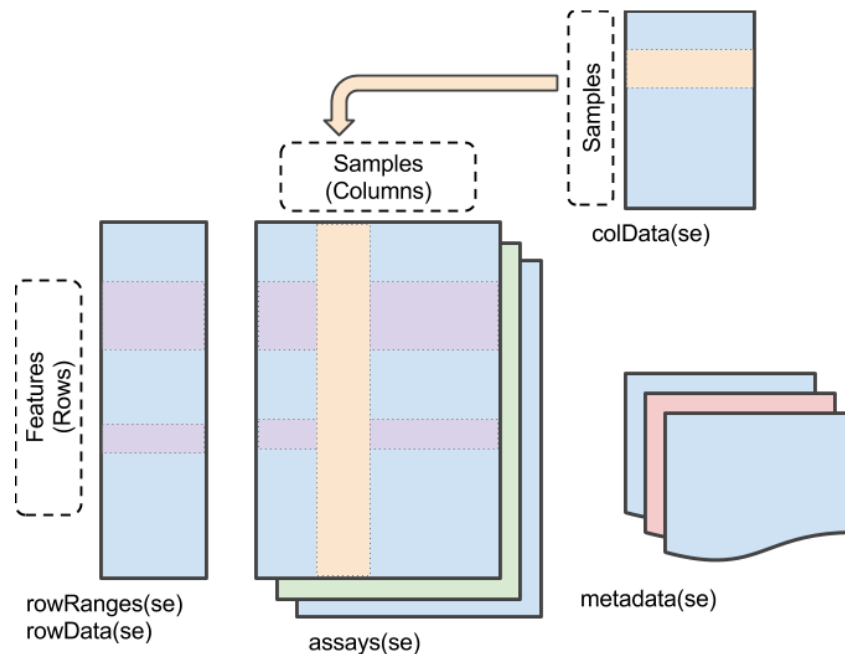
```
# How many genes do we have left:
dim(exprDat)
```

```
## [1] 15372    13
```

## Differential Expression Analysis- DESeq2

We will now make a DESeq2 object. For this we use the function `DESeqDataSetFromMatrix` from the DESeq2 package.

DESeq object is a type of SummarizedExperiment container used to store the input values, intermediate calculations and results of an analysis of differential expression. The rows typically represent Genes (genomic ranges) of interest and the columns represent samples.



First, Convert `exprDat` to a dataframe and make `GeneNames` column into rownames:

```
# Pull out GeneNames and EntrezGeneID for later use
GeneNames <- exprDat %>%
  dplyr::select(GeneName)

exprDat <- exprDat %>%
  column_to_rownames(., var = "GeneName")
```

Make a DESeq2 object: As input we give our count matrix, our gene IDs and our meta data (`exprInfo`). Additionally we include a design for DE contrasts. In this case we add `CellType` (luminal or basal) and `Status` (control, pregnant or lactating).

```
exprObj <- DESeqDataSetFromMatrix(countData = exprDat,
                                  colData = exprInfo,
                                  design= ~CellType+Status)

exprObj
```

```
## class: DESeqDataSet
## dim: 15372 12
## metadata(1): version
## assays(1): counts
## rownames(15372): Xkr4 Sox17 ... Asmt Gm47283,
## rowData names(0):
## colnames(12): MCL1.DG MCL1.DH ... MCL1.LE MCL1.LF
```

```
## colData names(4): SampleName CellType Status CellType.colors
```

### Preliminary analysis:

There are multiple biases in RNAseq experiment: library size, genes length, genes GC composition, etc. Library size is the most well-known bias. For the purpose of DEA - genes length and GC composition are not so important because it is supposed to be about the same for the gene across different samples.

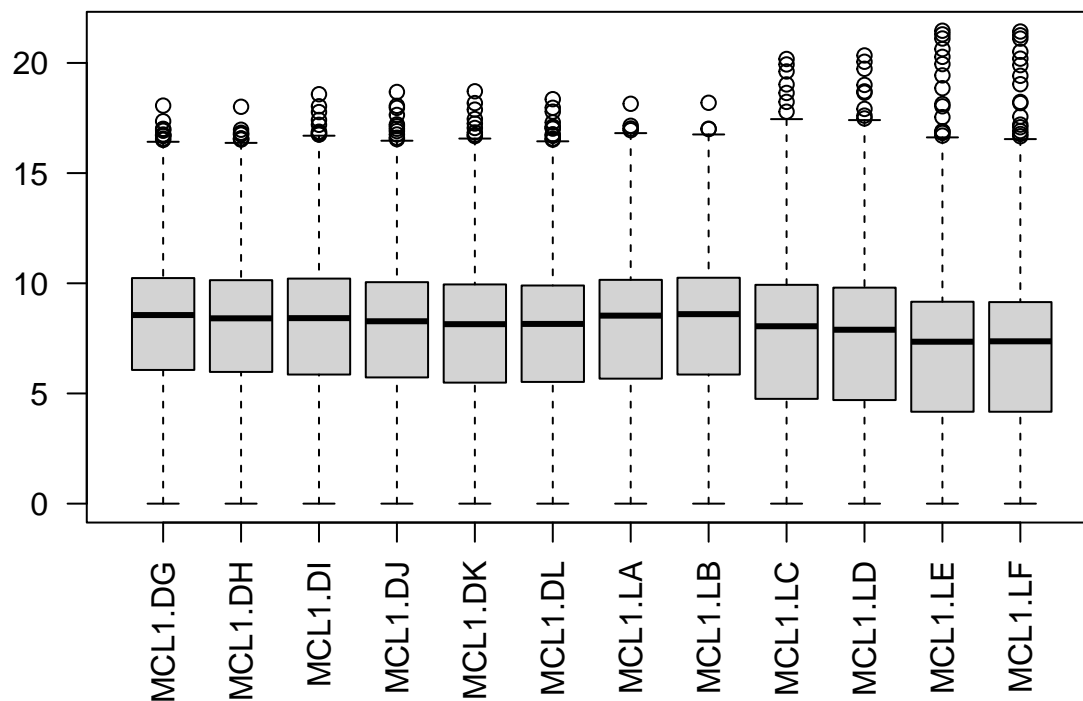
Let's have a look at the library sizes:

```
colSums(assay(exprObj))
```

```
## MCL1.DG MCL1.DH MCL1.DI MCL1.DJ MCL1.DK MCL1.DL MCL1.LA MCL1.LB
## 22630775 21151076 23485041 22097159 21054745 19580842 19694599 20939198
## MCL1.LC MCL1.LD MCL1.LE MCL1.LF
## 21672769 21455773 24417508 24364577
```

The count distributions may be dominated by a few genes with very large counts. These genes will drive plotting e.g. heatmaps, PCA analysis etc. Let's see if we have any "outlier" genes in our dataset and at the same time inspect the sample library sizes. For convenience I am using the base R boxplot function:

```
#boxplot(assay(exprObj), las=2)
boxplot(log2(assay(exprObj)+1), las=2)
```



As you can see we do not have any extreme outliers, but we do see some differences between libraries.

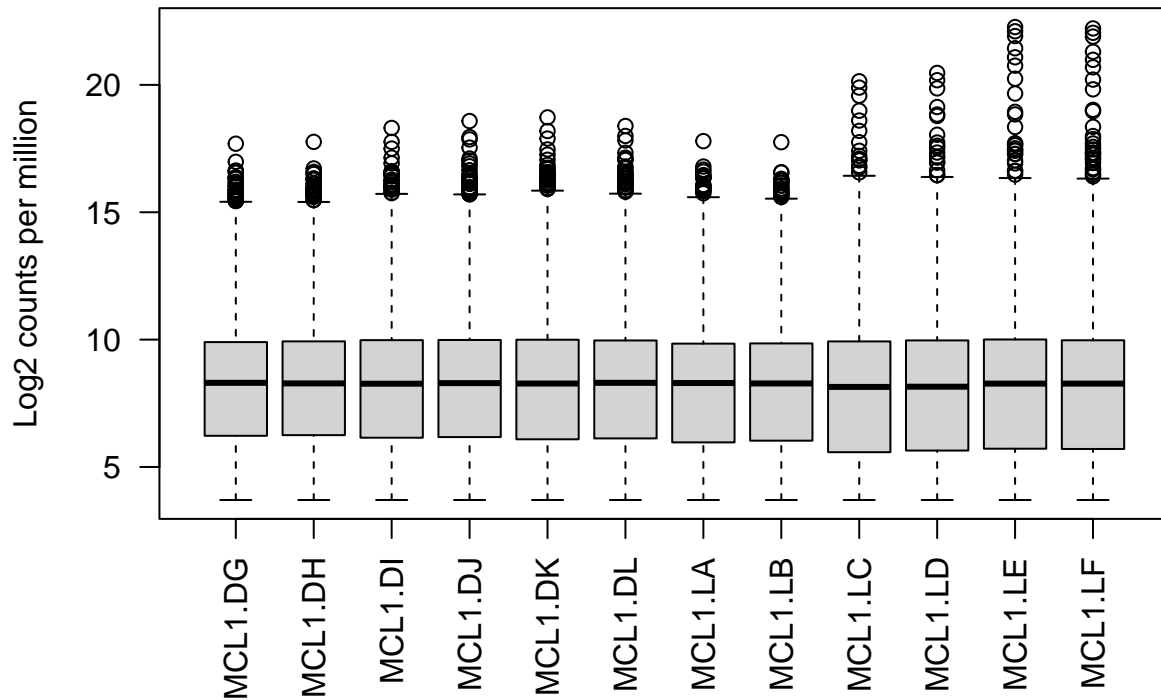
Next, we will apply the "vst" function to do a couple of things

- normalize library size to obtain counts per million mapped reads
- log2 transform the data to get more normally distributed data
- apply variance stabilizing transformation which we will discuss below.

```
exprObjvst <- vst(exprObj, blind=FALSE)
```

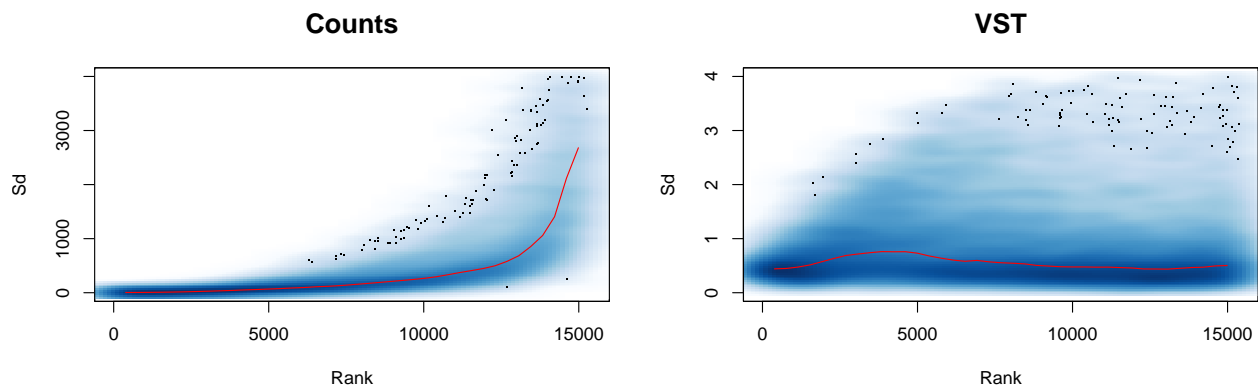
Let's plot normalized data.

```
par(mfrow=c(1,1))
boxplot(assay(exprObjvst), xlab="", ylab="Log2 counts per million", las=2)
```



### Variance stabilizing transformation:

In RNA-Seq data, genes with larger average expression have on average larger observed variances (sd) across samples. This is known as data heteroscedasticity. Expression varies from sample to sample more than other genes with lower average expression.



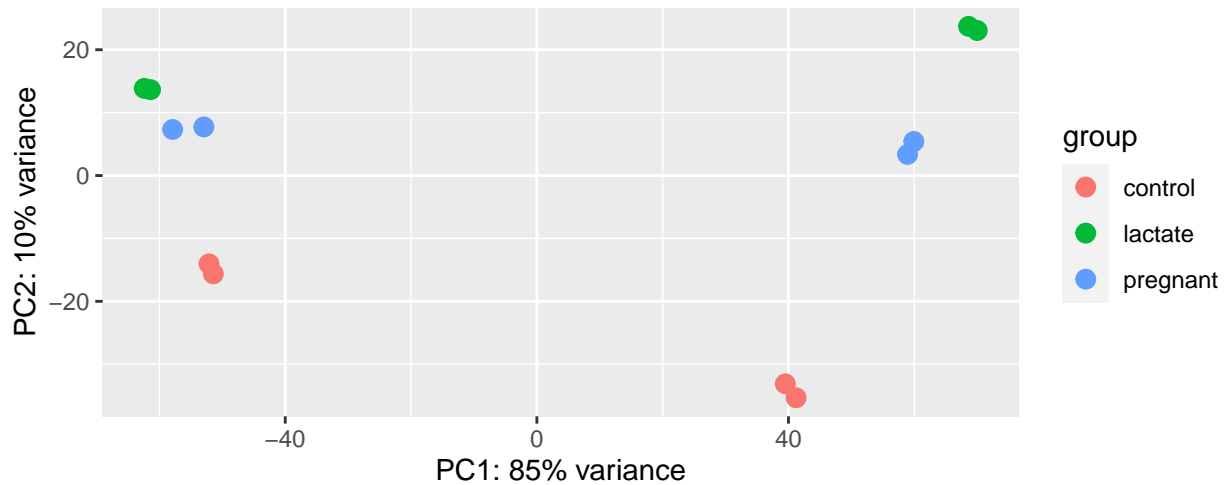
### Principal Component Analysis

Before performing DEA it is a good idea to explore how samples cluster together based on their gene expression profile. The expectation here is that samples from the same group (treatment vs control, condition A vs condition B, etc.) will cluster together. A principal component analysis (PCA) plot can also help us

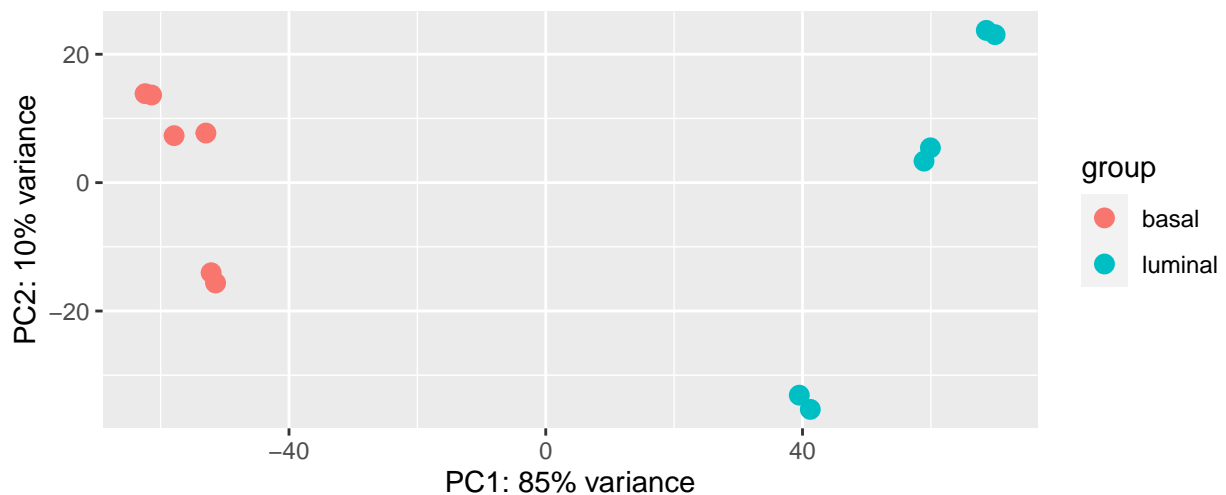


to identify outlier samples which might need to be removed from the analysis. We use our vst counts for principal component analysis:

```
plotPCA(exprObjvst, intgroup="Status")
```



```
plotPCA(exprObjvst, intgroup="CellType")
```



---

### DESeq function for DEA

Next, we use `DESeq()` to estimate library sizes, gene-wise and mean-dispersion, fitting models and post hoc testing:

```
exprObj <- DESeq(exprObj)
```

---

### Testing

Have a look at the group comparisons:

```
resultsNames(exprObj)
```

```
## [1] "Intercept" "CellType_luminal_vs_basal"  
## [3] "Status_lactate_vs_control" "Status_pregnant_vs_control"
```

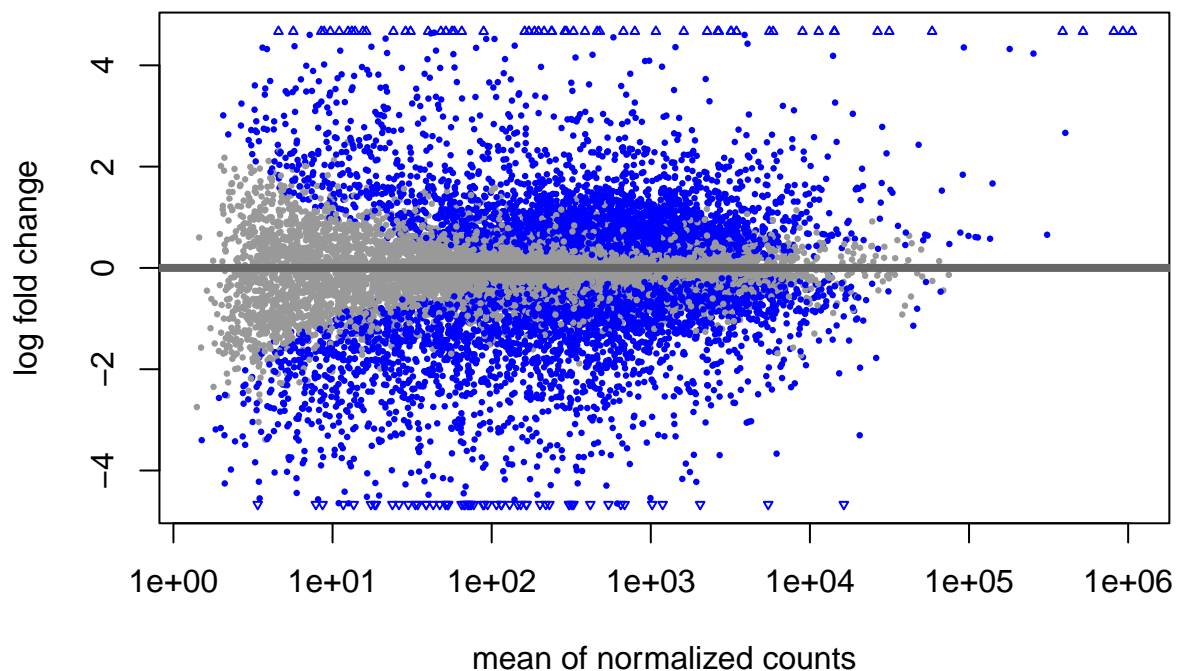
Test for DE genes between the three groups of mice, adjusted for cell type:

(I) lactating and control mice:

```
resLC <- results(exprObj, contrast = c("Status", "lactate", "control"), independentFiltering = FALSE)
```

Summary and plot of DE analysis results:

```
DESeq2::plotMA(resLC)
```



```
summary(resLC)
```

```
##  
## out of 15372 with nonzero total read count  
## adjusted p-value < 0.1  
## LFC > 0 (up) : 3574, 23%  
## LFC < 0 (down) : 3527, 23%  
## outliers [1] : 0, 0%  
## low counts [2] : 0, 0%  
## (mean count < 0)  
## [1] see 'cooksCutoff' argument of ?results  
## [2] see 'independentFiltering' argument of ?results
```

Below we perform the same steps as above to get the DE genes between (II) pregnant and control mice and (III) lactating and pregnant mice:

Convert DESeq2 object to tibble for further analysis:

```
resLC <- resLC %>%
  as.data.frame() %>%
  rownames_to_column(., var = "GeneName")
```

(II) pregnant and control mice:

```
resPC <- results(exprObj, contrast = c("Status", "pregnant", "control"), independentFiltering = FALSE)

#DESeq2::plotMA(resPC)
#summary(resPC)

resPC <- resPC %>%
  as.data.frame() %>%
  rownames_to_column(., var = "GeneName")
```

(III) lactating and pregnant mice:

```
resLP <- results(exprObj, contrast = c("Status", "lactate", "pregnant"), independentFiltering = FALSE)

#DESeq2::plotMA(resLP)
#summary(resLP)

resLP <- resLP %>%
  as.data.frame() %>%
  rownames_to_column(., var = "GeneName")
```

We filter the results of the DEA to only include those genes which are differentially expressed based on logFC ( $\geq 1.0$  or  $\leq -1.0$ ) and adjusted p-value ( $< 0.01$ ).

Firstly, bind the three DE genesets together and convert to a tibble. Then, add a column indicating in which pairwise comparison the gene was DE. Lastly, filter rows (genes) based on logFC and adjusted p-values.

```
resDE <- rbind(resLC, resPC, resLP) %>%
  as_tibble() %>%
  mutate(pair = c(rep("Lactate.Control", nrow(exprDat)),
                  rep("Pregnant.Control", nrow(exprDat)),
                  rep("Lactate.Pregnant", nrow(exprDat)))) %>%
  filter((log2FoldChange >= 1.0 | log2FoldChange <= -1.0) & padj <= 0.05)

# Give it a look:
resDE
```

```
## # A tibble: 5,695 x 8
##   GeneName      baseMean log2FoldChange lfcSE  stat    pvalue    padj pair
##   <chr>          <dbl>          <dbl> <dbl> <dbl>    <dbl>    <dbl> <chr>
## 1 Rgs20          46.0          -1.40 0.566 -2.47 1.37e- 2 3.73e- 2 Lactate.~
## 2 Pcmt1d1       1468.           1.33 0.179  7.45 9.41e-14 4.23e-12 Lactate.~
## 3 Adhfe1        118.           2.33 0.541  4.31 1.64e- 5 1.24e- 4 Lactate.~
## 4 2610203C22Rik  17.0           1.70 0.609  2.79 5.28e- 3 1.72e- 2 Lactate.~
## 5 Vxn           21.3           2.36 0.539  4.38 1.17e- 5 9.23e- 5 Lactate.~
## 6 Mybl1         254.          -1.01 0.266 -3.77 1.60e- 4 9.02e- 4 Lactate.~
## 7 A830018L16Rik   3.56          -3.24 1.31  -2.47 1.36e- 2 3.72e- 2 Lactate.~
## 8 Slco5a1        10.7          -1.72 0.701 -2.46 1.40e- 2 3.80e- 2 Lactate.~
```

```
## 9 Lactb2          243.          1.13 0.197  5.73 1.03e- 8 1.75e- 7 Lactate.~
## 10 Eya1           197.          1.01 0.309  3.25 1.14e- 3 4.83e- 3 Lactate.~
## # ... with 5,685 more rows
```

Number of DE genes:

```
# Number of DE genes:
dim(resDE)
```

```
## [1] 5695    8
```

```
# Number of distinct DE genes:
resDE %>%
  distinct(GeneName) %>%
  nrow()
```

```
## [1] 3678
```