Bioinformatics and Systems Biology: Expression, Regulation and Networks Exam project

Emiel Vanhulle r0308983, Jasper Vanhecke r0309321, Lena Smets r0387223

2e master Biomedical Sciences 2016-2017

RNAseq analysis workflow using DESeq2

We first loaded the packages needed for using DESeq2

In [85]:

```
library("DESeq2")
library("scales")
#scales is helper function for colours
options(repr.plot.width=4, repr.plot.height=3)
```

In [86]:

```
setwd(path.expand("/mnt/storage/r0308983/jupyternotebooks/examwork/RNASeq/DeSEQ"))
```

We define the file containing the raw counts (the ones we generated in the other notebook : gene.counts.test1.tsv)

In [87]:

```
matrixFile = path.expand("/mnt/storage/r0308983/jupyternotebooks/examwork/RNASeq/gene.count
```

In [88]:

```
countData <- as.matrix(read.csv(matrixFile, sep="\t", row.names="Geneid"))
head(countData)
#tab (t) separated
```

	SRR1654369.bam	SRR1654370.bam
Xkr4	155	98
Gm39585	8	12
LOC108167595	0	0
Gm38717	0	0
Rp1	0	0
Sox17	29	11

In [89]:

```
tail(countData)
```

	SRR1654369.bam	SRR1654370.bam
TrnH	83	131
TrnS2	6	16
TrnL2	44	47
TrnE	1341	1865
TrnT	135	272
TrnP	298	506

Now we will make this better looking by changing the names of the samples to the corresponding condition. The first sample is the WT sample, the second sample is the KO sample.

In [90]:

```
colnames(countData) <- c("Mouse.ES.WT", "Mouse.ES.Prep1_KO")
```

In [91]:

```
head(countData)
```

	Mouse.ES.WT	Mouse.ES.Prep1_KO
Xkr4	155	98
Gm39585	8	12
LOC108167595	0	0
Gm38717	0	0
Rp1	0	0
Sox17	29	11

Next, we define the experimental conditions. We also check the order of the levels. In this case WT comes first, meaning we will be looking at the ratio of KO/WT, meaning that negative fold changes will be downregulated genes and vice versa.

In [92]:

```
condition <- factor (c("WT", "Prep1_KO"))
```

In [93]:

```
print(condition)
```

```
[1] WT       Prep1_KO
Levels: Prep1_KO WT
```

Next, we will prepare a dataframe with the names and conditions. This will be fed to `DESeq2`.

In [94]:

```
colData <- data.frame(sampleName = colnames(countData),
                      condition = condition)
colData
```

sampleName	condition
Mouse.ES.WT	WT
Mouse.ES.Prep1_KO	Prep1_KO

Then, we create a DESeq `DESeqDataSet` object for subsequent analysis:

In [95]:

```
dds <- DESeqDataSetFromMatrix(countData = countData,
                              colData = colData,
                              design = ~ condition)
#we need to do this to ensure the column names are correct.

colnames(dds) <- colnames(countData)
```

In [96]:

```
colnames(dds)
```

'Mouse.ES.WT' 'Mouse.ES.Prep1_KO'

In [97]:

```
dds
```

```
class: DESeqDataSet
dim: 41575 2
exptData(0):
assays(1): counts
rownames(41575): Xkr4 Gm39585 ... TrnT TrnP
rowRanges metadata column names(0):
colnames(2): Mouse.ES.WT Mouse.ES.Prep1_KO
colData names(2): sampleName condition
```

By comparing the `condition` factor in the `dds` object to the column names, we can confirm that the order is still correct.

In [98]:

```
dds$condition
```

WT Prep1_KO

Run the differential expression:

Within `DESeq2` , one function will run all steps required.

In [100]:

```
?DESeq
```

DESeq2 is able to estimate a factor to correct for the library size:

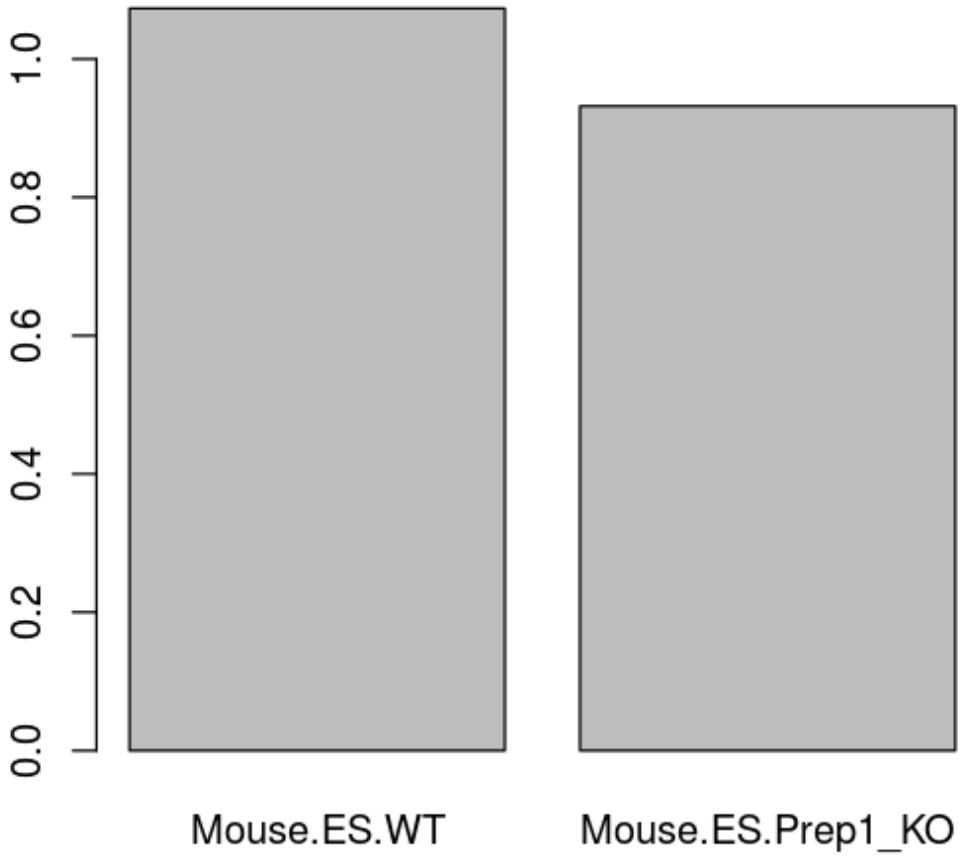
In [99]:

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

```
estimating size factors
estimating dispersions
Warning message:
In checkForExperimentalReplicates(object, modelMatrix): same number of sampl
es and coefficients to fit,
  estimating dispersion by treating samples as replicates.
  read the ?DESeq section on 'Experiments without replicates'gene-wise dispe
rsion estimates
mean-dispersion relationship
final dispersion estimates
fitting model and testing
```

In [101]:

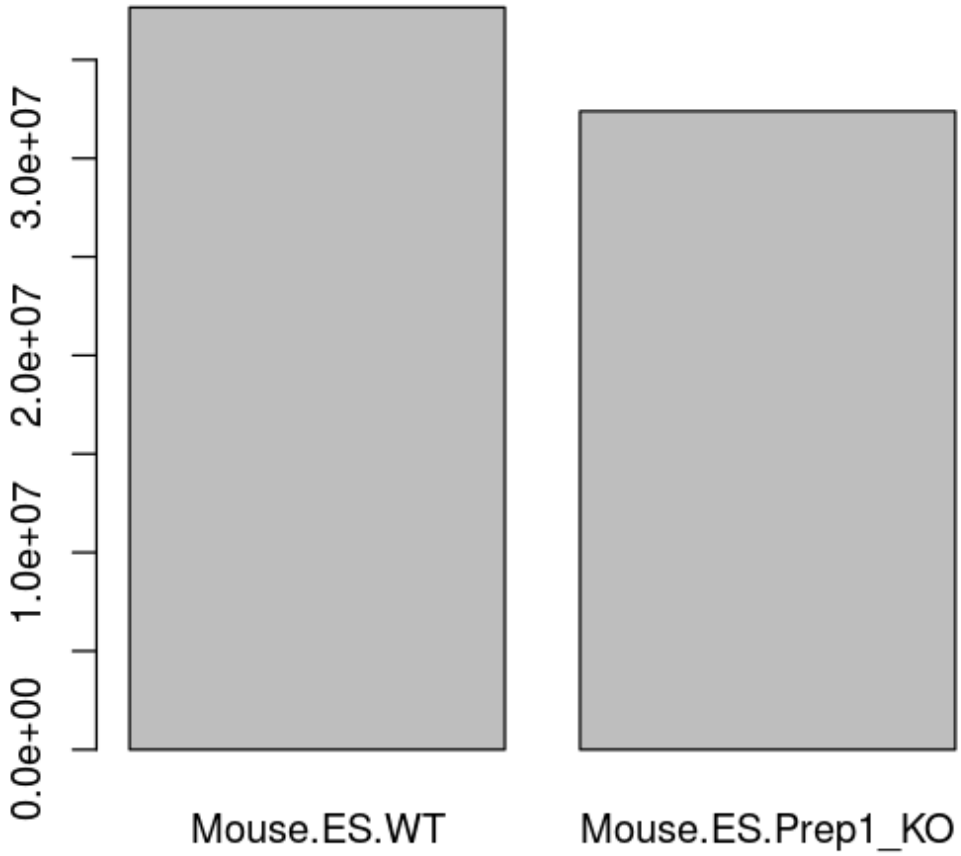
```
options(repr.plot.width=5, repr.plot.height=5)
barplot(sizeFactors(dds))
```



This is very comparable to the sum of all reads per sample of the raw data. However, `DESeq2` is less sensitive to outliers.

In [102]:

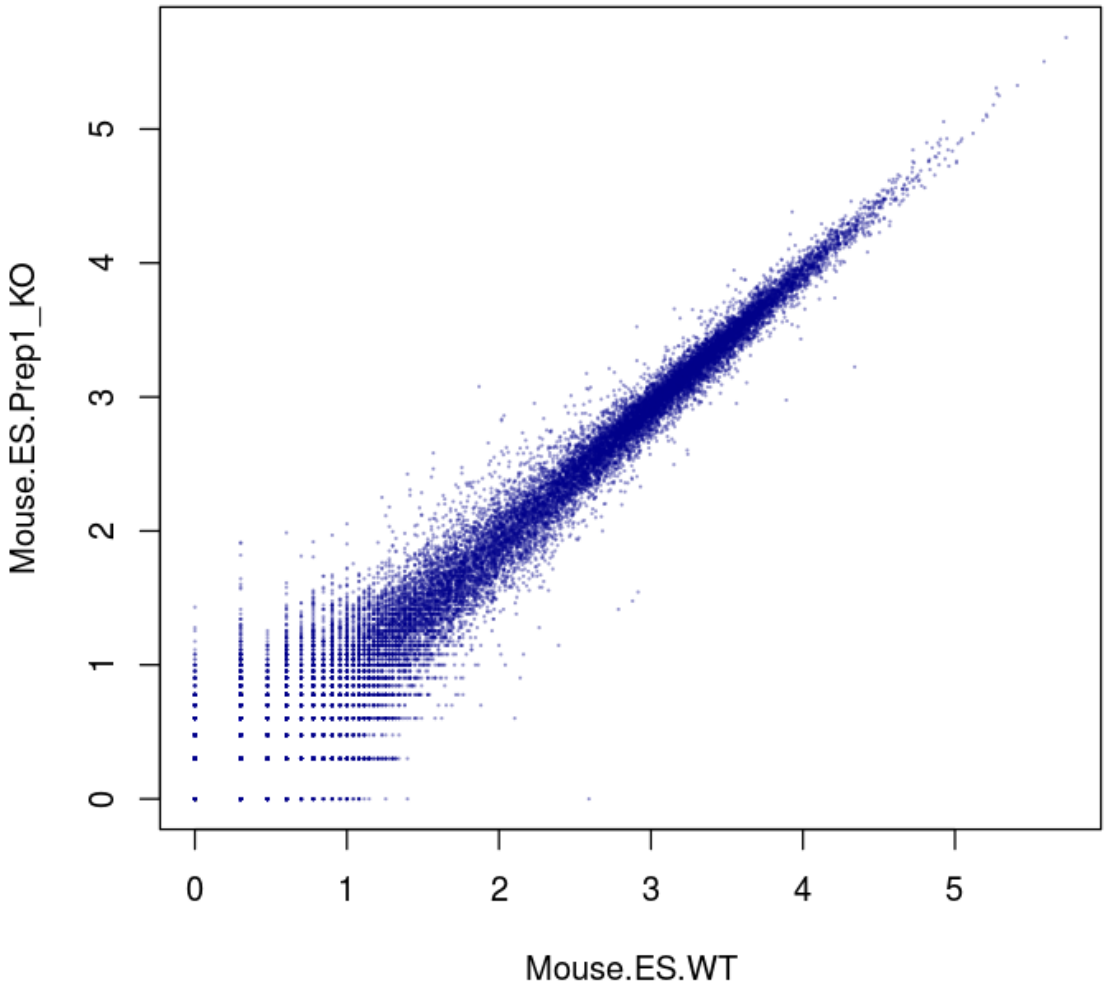
```
barplot(colSums(countData))
```



Scatter plots allows us to compare samples. A scatter plot of sample vs sample shows how the samples compare.

In [126]:

```
options(repr.plot.width=6, repr.plot.height=6)
plot(data.frame(log10(counts(dds))), cex=0.1, col=alpha("dark blue", 0.4))
```



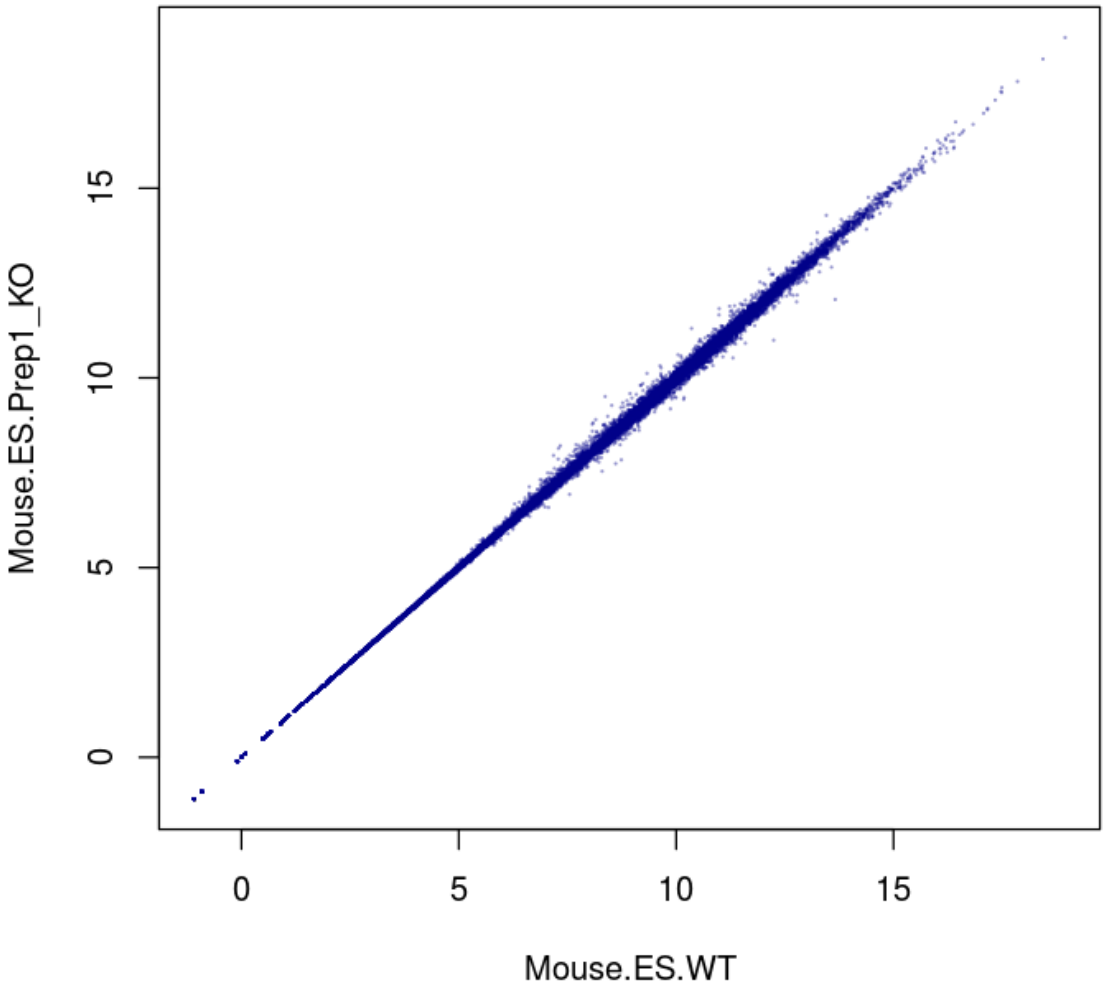
There is a lot of variation (noise) for the low expressing genes. DESeq2 provides therefore a function that minimizes this variation. So next, we will minimize the variation of genes that have low expression, normalize for library size and perform a log transformation on the data.

In [104]:

```
rld <- rlogTransformation(dds, blind=TRUE)
```

In [127]:

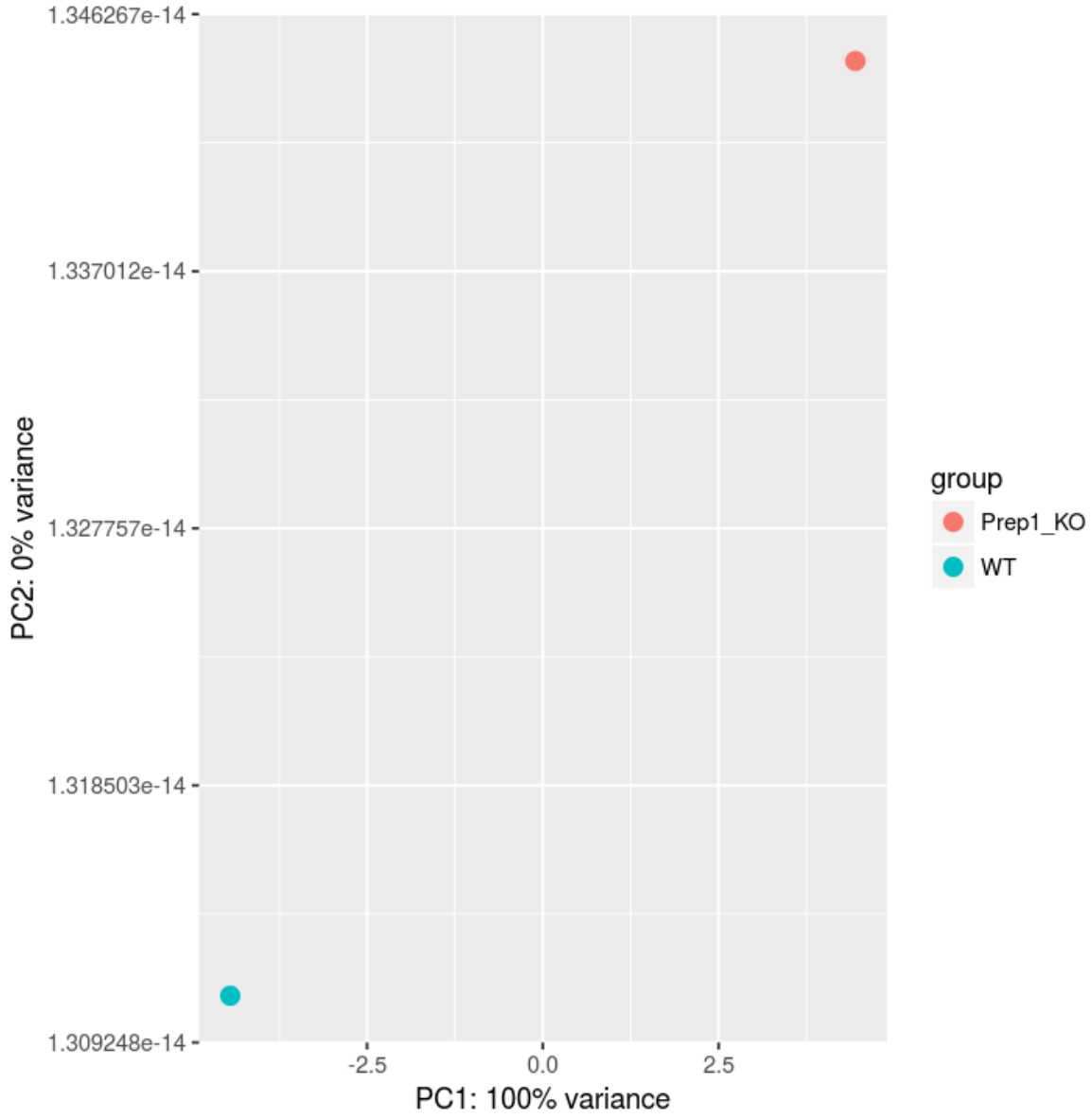
```
options(repr.plot.width=6, repr.plot.height=6)
plot(data.frame(assay(rld)), cex=0.1, col=alpha("dark blue", 0.4))
```



Next, we do a PCA plot to understand inter-sample variation

In [106]:

```
plotPCA(rld, intgroup=c("condition"))
```

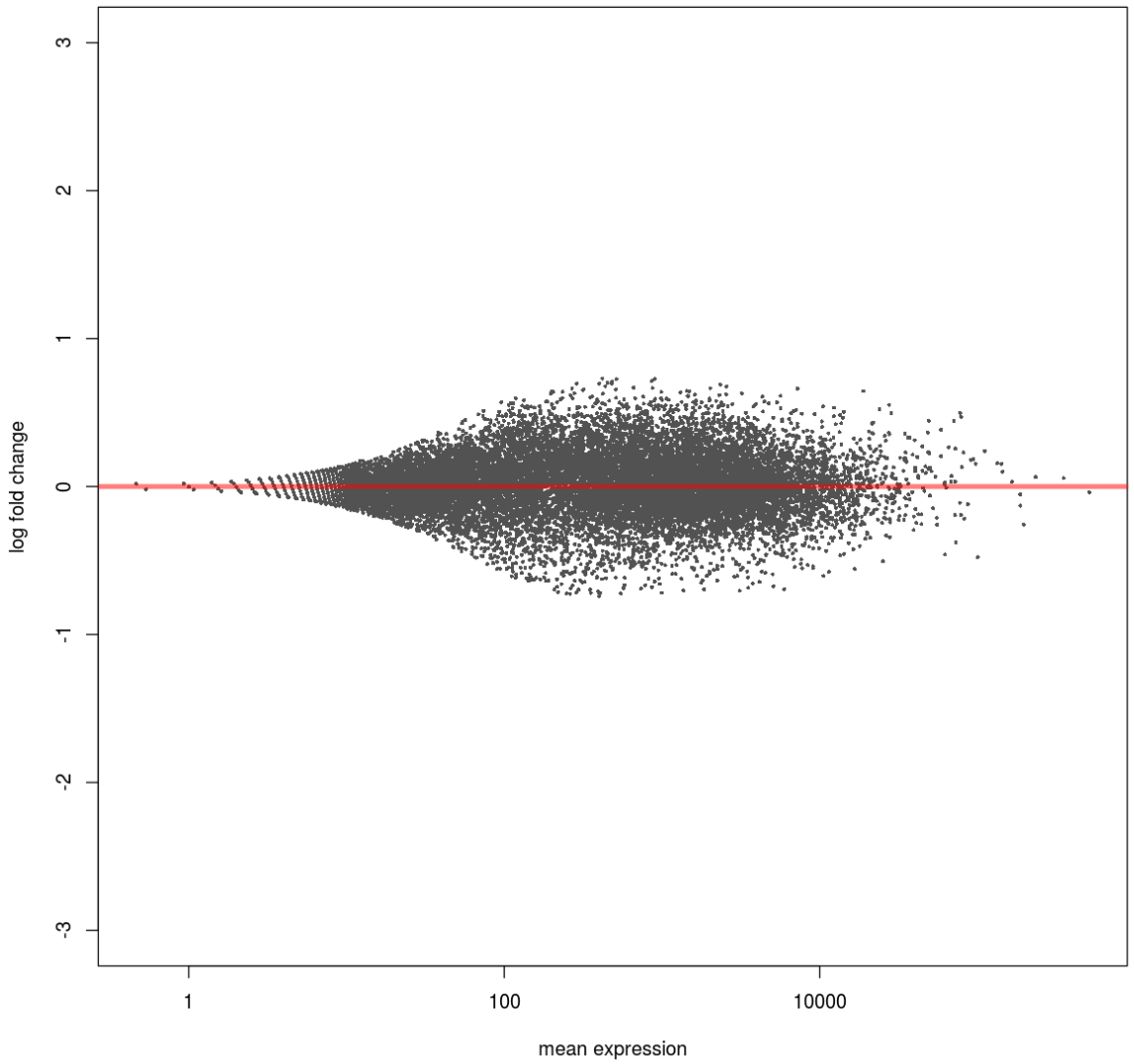


MA plot

An MA plot shows the mean expression vs the log fold change and is a very common method to visualize the results of a differential expression plot. Significantly differentially expressed genes are colored red.

In [121]:

```
options(repr.plot.width=10, repr.plot.height=10)
plotMA(dds,ylim=c(-3,3))
```



There are no red dots or red coloured genes. This means that there are no significantly differentially expressed genes.

Exporting of the results

In [114]:

```
res <- results(dds)
res <- res[order(res$pvalue),]
#sort on adjusted in R, $means give me column of res and ordered op padj (adjusted p-value)
```

In [115]:

```
head(res)
```

```
log2 fold change (MAP): condition WT vs Prep1_KO
Wald test p-value: condition WT vs Prep1_KO
DataFrame with 6 rows and 6 columns
              baseMean log2FoldChange     lfcSE      stat      pvalue      p
adj
             <numeric>      <numeric> <numeric> <numeric>   <numeric> <numer
ic>
Mt4           219.6784     -0.7130599 0.2721121 -2.620464 0.008781030 0.9993
353
Slc4a5        248.9206     -0.7220979 0.2758054 -2.618143 0.008840982 0.9993
353
Cmklr1        261.2331     -0.7251409 0.2770133 -2.617712 0.008852156 0.9993
353
LOC102634296  399.1412     -0.7416170 0.2834774 -2.616141 0.008892973 0.9993
353
TrnL1        3105.0563     -0.7046302 0.2728770 -2.582227 0.009816510 0.9993
353
Pknox1        901.3472      0.7282193 0.2823107  2.579496 0.009894465 0.9993
353
```

We use the following command to see how many significant differentially expressed genes we have (with a P value lower that 0.05). We filtered in P value instead of on adjusted P value, because if we filter on adjusted P value, there will be no differentially expressed genes...

In [116]:

```
res.sign <- res[(!is.na(res$pvalue)) & (res$pvalue < 0.05),]
dim(res.sign)
```

```
    303  6
```

303 genes are differentially expressed if we look at the pvalues

Note: Replicates

We didn't work with the p-adjusted, because in the experiment from the article, there are no replicates. It is not possible to conduct a meaningful statistical analysis such as DESeq2 when comparing just two samples (WT and KO). If a DESeqDataSet is provided with an experimental design without replicates, a warning is printed, that the samples are treated as replicates for estimation of dispersion. This kind of analysis is only useful for exploring the data, but will not provide the kind of proper statistical inference on differences between groups. Without biological replicates, it is not possible to estimate the biological variability of each gene. Experiments without replicates do not allow for estimation of the dispersion of counts around the expected value for each group, which is critical for differential expression analysis. If an experimental design is supplied which does not contain the necessary degrees of freedom for differential analysis, DESeq will provide a message to the user and follow the strategy outlined in Anders and Huber (2010) under the section 'Working without replicates', wherein all the samples are considered as replicates of a single group for the estimation of dispersion. Furthermore, while one may not want to draw strong conclusions from such an analysis, it may still be useful for exploration and hypothesis generation.

The difference with what they did in the article, is that they modified their dataset. We worked with the whole dataset as it was original. They considered only those RNAs whose level of expression (number of reads) was >1 and in this group, they set an arbitrary threshold to define Prep1-regulated genes (those in which the effect of the absence of Prep1 resulted in an over 30% decrease or 50% increase in expression). Then they found 910 overexpressed genes and 1802 down-regulated genes. We found (looking at the p-values) 303 genes that were differentially expressed.

But we still question why in the article they used DESeq2 when they didn't have replicates?

Save to disk

Finally, we write our results to a tsv file which we can download and open in Excel.

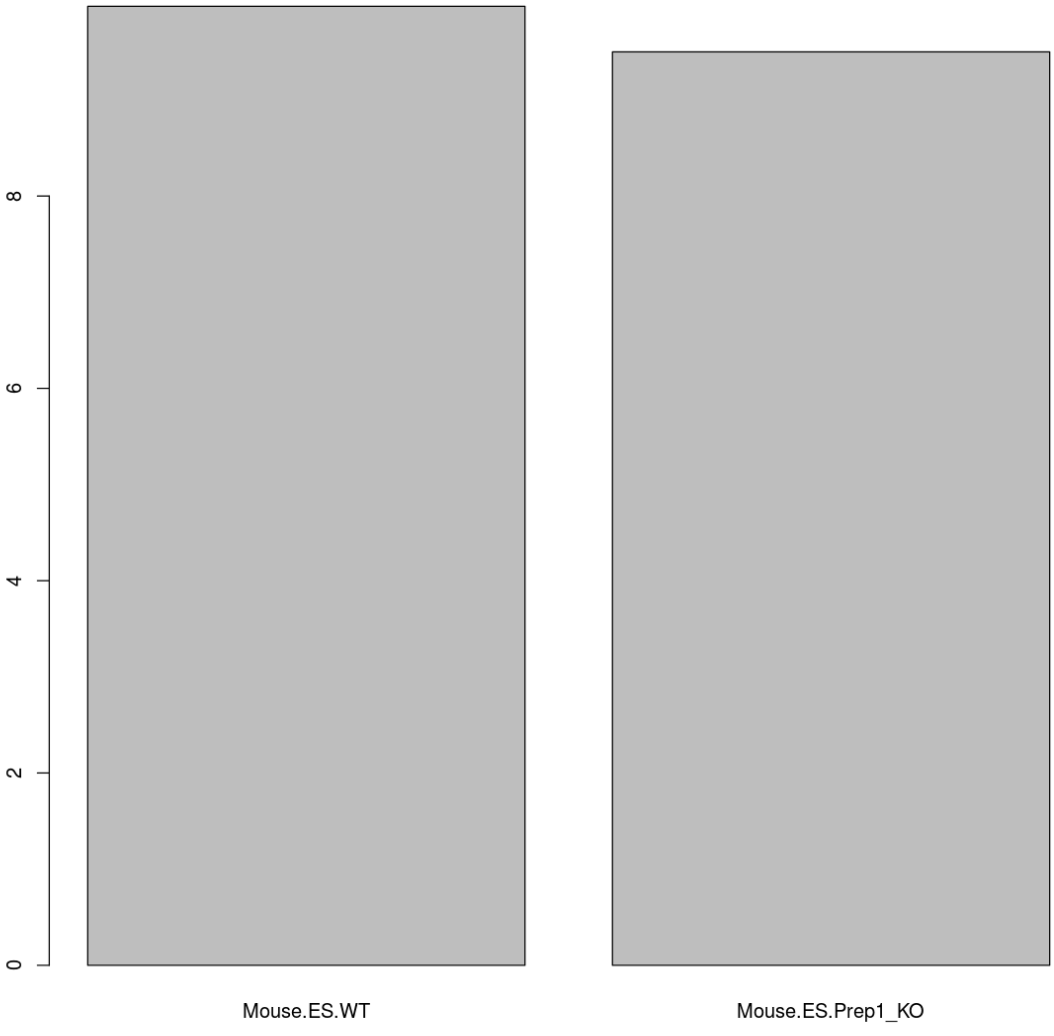
In [117]:

```
write.table(res, "deseq.resultsRNAseq.mouse.tsv", sep="\t", col.names=NA, quote=FALSE)
```

We will manually check a number of diff. expressing genes.

In [118]:

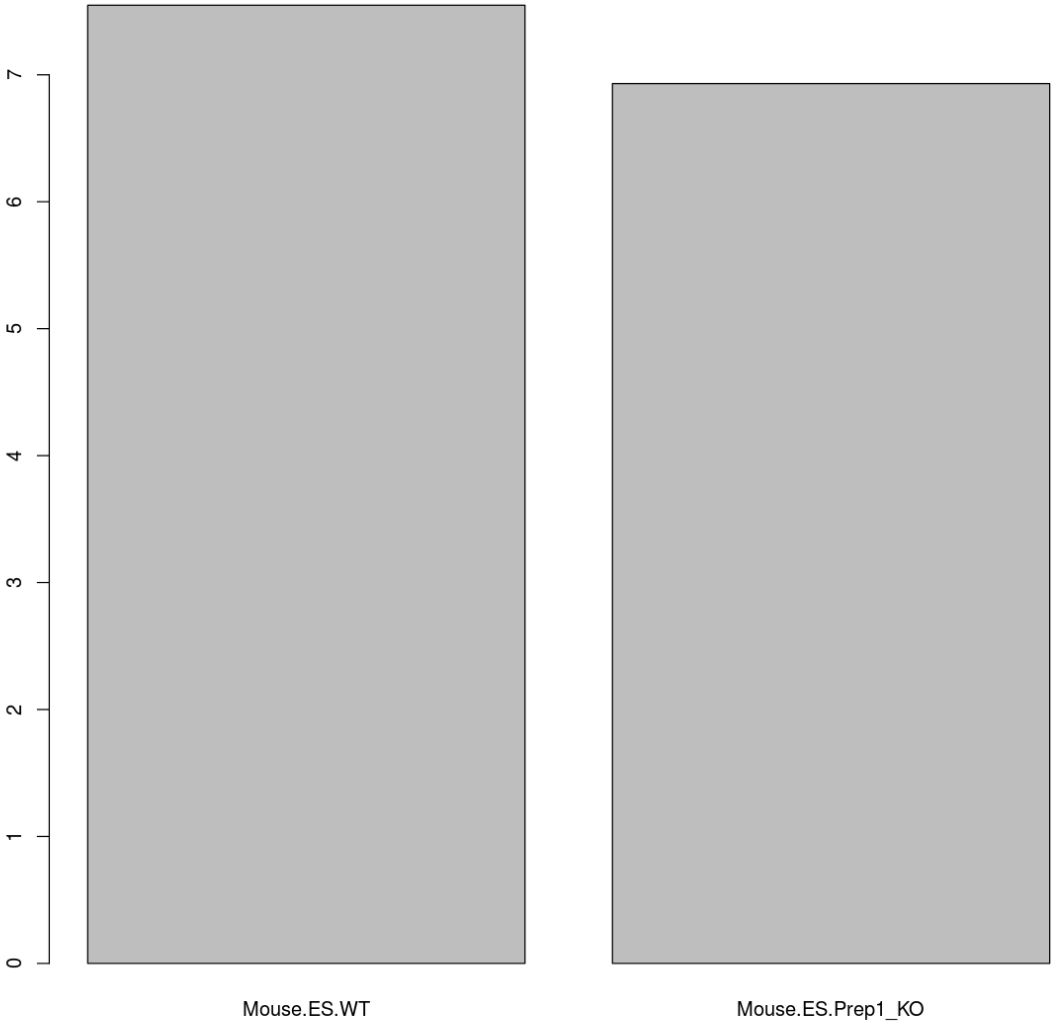
```
barplot(assay(rld)["Pknox1",])
```



As expected, Pknox1 goes down in the Prep1(Pknox1)_KO cells

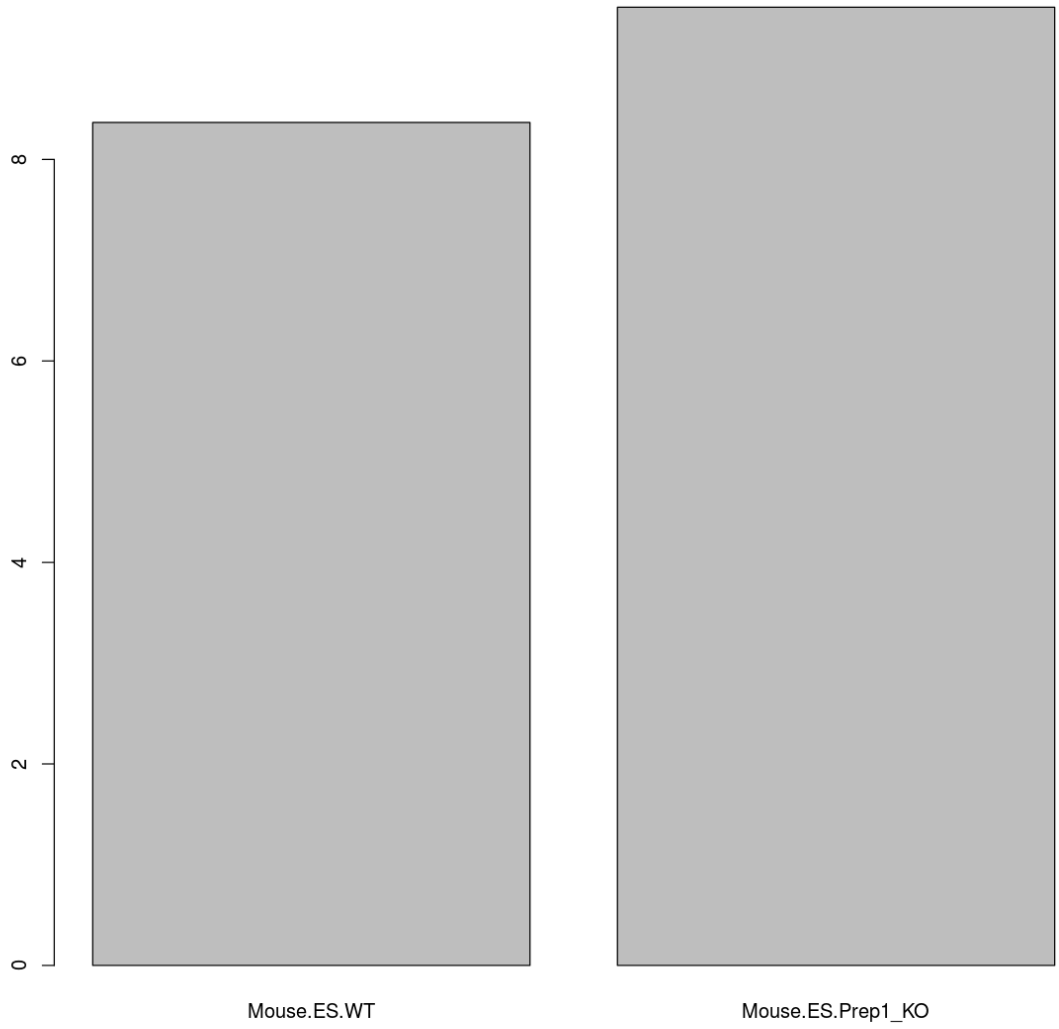
In [119]:

```
barplot(assay(rld)["Gm10499",])
```



In [120]:

```
barplot(assay(rld)["Nxf3",])
```



Note: EXTRA

As noted before (under replicates) it is not meaningful to use DESeq2 on a dataset without replicates. However, we still wanted to further analyze this data. Now we will use DESeq2's new "rlog transformation". It transforms the average of the genes across samples to a log2 scale but "pulls in" those genes for which the evidence for strong fold changes is weak due to low counts.

In [74]:

```
rld <- rlogTransformation(dds)
```

As this is a logarithm-like scale, the differences between the samples can be considered as a "regularized log2 fold change". Next, we made a result data frame: