Practical: exploring RNA-Seq counts

Hugo Varet, Julie Aubert and Jacques van Helden

2016-11-24

Contents

Requirements **2**

Context **2**

Loading a data table **2**

Checking the content of the count tables **3**

Factors and levels in R **3**

Basic description of the data: number of reads per sample **4**

Basic description of the data: percentage of null counts per sample **4**

Differential analysis with DESeq2 **5**

Get the results using two command lines **5**

Sub-sampling: analysis using a few replicates **7**

Normalization **7**

Mean-variance relationship **8**

Principal Component Analysis (PCA) **9**

Statistical test for each gene **9**

Histogram of raw P-values **10**

MA-plot **10**

Volcano-plot **11**

Differential analysis using edgeR with a few replicates **12**

Normalization & dispersions estimation with edgeR **13**

Modeling and testing with edgeR **13**

Histogram of raw P-values **14**

Compare DESeq2 and edgeR results: normalization factors **15**

Re-order the results according to the gene names **15**

Comparing log2(Fold-Change) estimations **16**

Comparing raw P-values **16**

Number of differentially expressed genes **17**

Venn diagram **17**

What's edgeR or DESeq2-specific? **18**

DESeq2 results for one gene **18**

Differential analysis under H_0 **19**

Differential analysis with DESeq2 under H_0 **20**

sessionInfo **20**

Requirements

For people using their own laptop, install some R packages:

```
source("http://bioconductor.org/biocLite.R")
biocLite(c("DESeq2","edgeR","gplots"), ask=FALSE)
```

Context

- Study of 48 WT yeast samples vs 48 Snf2 (KO) samples: Gierliński et al. *Statistical models for RNA-seq data derived from a two-condition 48-replicate experiment*, Bioinformatics, 2015.
- RNA-Seq reads have been cleaned, mapped and counted to generated a count data matrix containing 7126 genes.

Loading a data table

R enables to download data directly from the Web. Load the counts table containing one row per gene and one column per sample.

```
# Load the files content in an R data.frame
path.counts <- "http://jvanheld.github.io/stats_avec_RStudio_EBA/practicals/yeast_2x48_replicates/data/
counts <- read.table(file=path.counts, sep="\t", header=TRUE, row.names=1)

path.expDesign <- "http://jvanheld.github.io/stats_avec_RStudio_EBA/practicals/yeast_2x48_replicates/da
expDesign <- read.table(file=path.expDesign, sep="\t", header=TRUE)
```

Checking the content of the count tables

```
# look at the beginning of the counts and design table:
print(counts[1:4,1:4])
```

```
##          WT1 WT2 WT3 WT4
## 15s_rrna   2  12  31   8
## 21s_rrna  20  76 101  99
## hra1       3   2   2   2
## icr1      75 123 107 157
```

```
print(expDesign[1:4,])
```

```
##   label strain
## 1   WT1     WT
## 2   WT2     WT
## 3   WT3     WT
## 4   WT4     WT
```

```
# dimension of each table
dim(counts)
```

```
## [1] 7126   96
```

```
dim(expDesign)
```

```
## [1] 96  2
```

```
View(counts)
View(expDesign)
```

Factors and levels in R

Be careful to the reference level in the factor variables:

```
print(expDesign$strain)
```

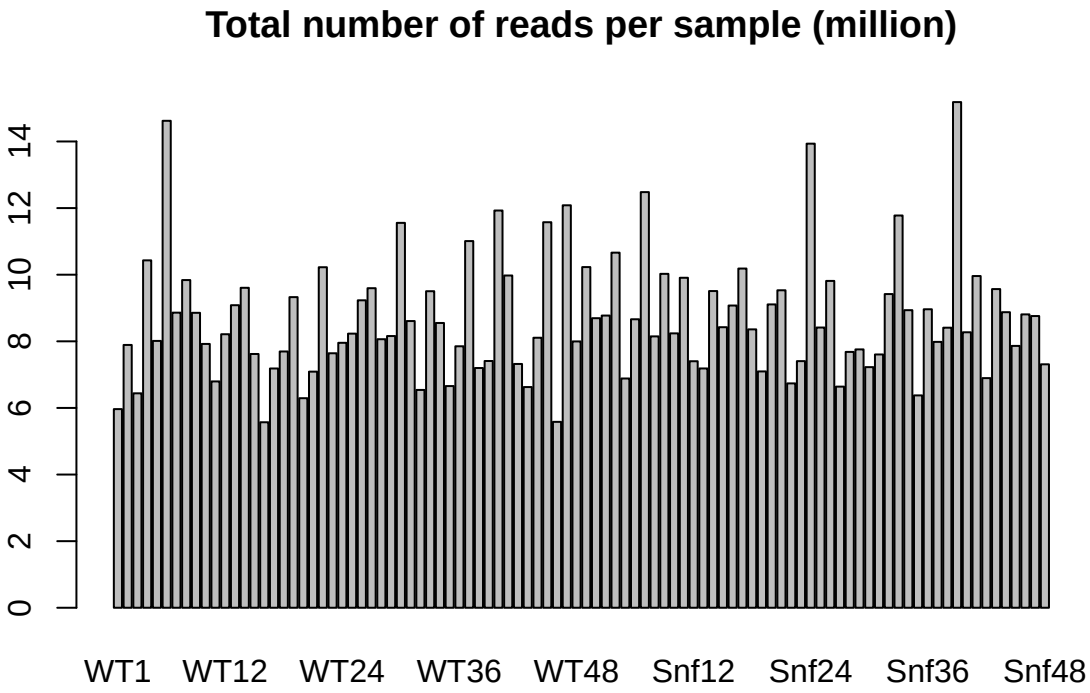
```
##  [1] WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT
## [75] Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf
## Levels: Snf WT
```

```
expDesign$strain <- relevel(expDesign$strain, "WT")
print(expDesign$strain)
```

```
##  [1] WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT  WT
## [75] Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf Snf
## Levels: WT Snf
```

Basic description of the data: number of reads per sample

```
barplot(colSums(counts)/1000000, main="Total number of reads per sample (million)")
```

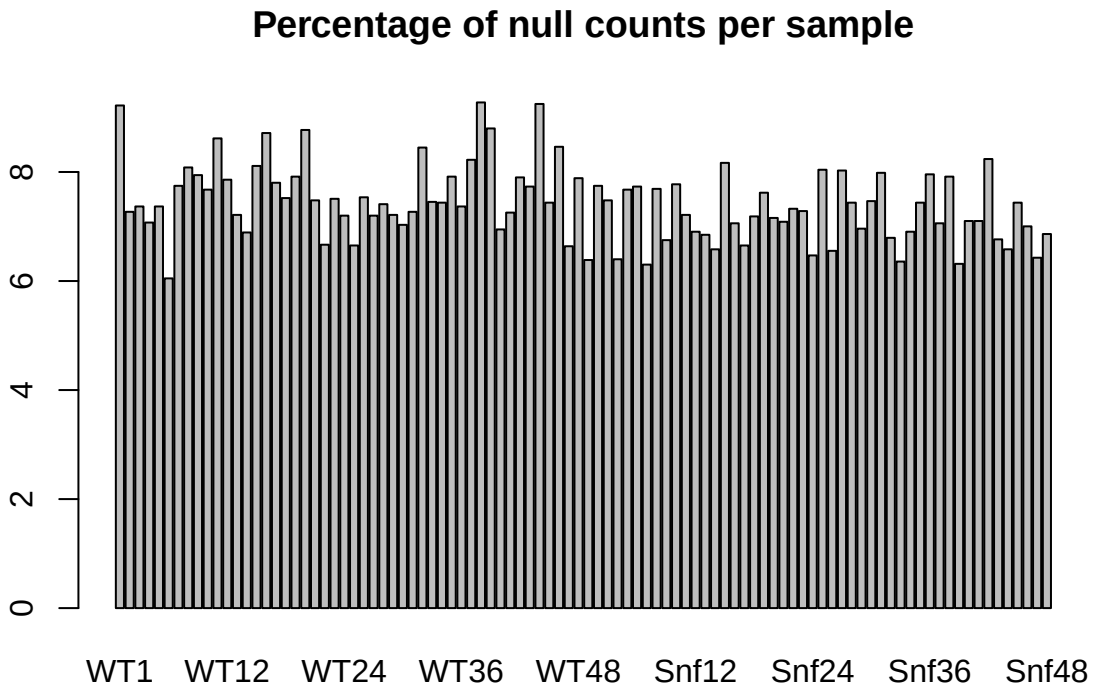


Basic description of the data: percentage of null counts per sample

```
prop.null <- apply(counts, 2, function(x) 100*mean(x==0))
print(head(prop.null))
```

```
##      WT1      WT2      WT3      WT4      WT5      WT6
## 9.219759 7.269155 7.367387 7.072692 7.367387 6.048274
```

```
barplot(prop.null, main="Percentage of null counts per sample")
```



Differential analysis with DESeq2

```
# load the DESeq2 R package
library(DESeq2)
# create the DESeq2 main object
dds0 <- DESeqDataSetFromMatrix(countData = counts, colData = expDesign, design = ~ strain)
print(dds0)
```

```
## class: DESeqDataSet
## dim: 7126 96
## metadata(1): version
## assays(1): counts
## rownames(7126): 15s_rrna 21s_rrna ... ty(gua)o ty(gua)q
## rowData names(0):
## colnames(96): WT1 WT2 ... Snf47 Snf48
## colData names(2): label strain
```

Get the results using two command lines

```
dds0 <- DESeq(dds0)
```

```
## estimating size factors

## estimating dispersions
```

```
## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 10 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing
```

```
res0 <- results(dds0)
print(res0)
```

```
## log2 fold change (MAP): strain Snf vs WT
## Wald test p-value: strain Snf vs WT
## DataFrame with 7126 rows and 6 columns
##             baseMean log2FoldChange      lfcSE       stat       pvalue         padj
##            <numeric>      <numeric>  <numeric>  <numeric>    <numeric>    <numeric>
## 15s_rrna   18.336648     0.14533034 0.29148181  0.4985914 6.180672e-01 6.737305e-01
## 21s_rrna  107.325458    -0.08431727 0.25187200 -0.3347624 7.378043e-01 7.824543e-01
## hra1        2.526211    -0.74324768 0.20824687 -3.5690701 3.582505e-04 6.503087e-04
## icr1      141.574248     0.21494348 0.03695485  5.8163804 6.013555e-09 1.626683e-08
## lsr1      207.526479    -0.13222494 0.15297933 -0.8643321 3.874055e-01 4.497744e-01
## ...              ...            ...        ...        ...          ...          ...
## ty(gua)j2  0.1433690    -0.09418330  0.4293068 -0.2193846    0.8263505    0.8592897
## ty(gua)m1  0.3670378    -0.24446546  0.3635025 -0.6725277    0.5012478    0.5634232
## ty(gua)m2  0.1079545    -0.14671918  0.4051056 -0.3621752    0.7172211    0.7635234
## ty(gua)o   0.1136899    -0.05217501  0.4147428 -0.1258009    0.8998895    0.9210194
## ty(gua)q   0.0000000             NA         NA         NA           NA           NA
```

```
print(summary(res0))
```

```
##
## out of 6887 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)     : 2613, 38%
## LFC < 0 (down)   : 2522, 37%
## outliers [1]     : 0, 0%
## low counts [2]   : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
##
## NULL
```

```
print(mcols(res0))
```

```
## DataFrame with 6 rows and 2 columns
##           type                                description
##    <character>                                <character>
## 1 intermediate mean of normalized counts for all samples
## 2      results  log2 fold change (MAP): strain Snf vs WT
## 3      results          standard error: strain Snf vs WT
## 4      results          Wald statistic: strain Snf vs WT
## 5      results       Wald test p-value: strain Snf vs WT
## 6      results                       BH adjusted p-values
```

Sub-sampling: analysis using a few replicates

```
nb.replicates <- 4
samples.WT <- sample(1:48, size=nb.replicates, replace=FALSE)
samples.Snf2 <- sample(49:96, size=nb.replicates, replace=FALSE)
print(c(samples.WT, samples.Snf2))
```

```
## [1] 39 16 11 26 90 69 87 68
```

```
dds <- DESeqDataSetFromMatrix(countData = counts[,c(samples.WT, samples.Snf2)],
                              colData = expDesign[c(samples.WT, samples.Snf2),],
                              design = ~ strain)
print(dds)
```

```
## class: DESeqDataSet
## dim: 7126 8
## metadata(1): version
## assays(1): counts
## rownames(7126): 15s_rrna 21s_rrna ... ty(gua)o ty(gua)q
## rowData names(0):
## colnames(8): WT39 WT16 ... Snf39 Snf20
## colData names(2): label strain
```

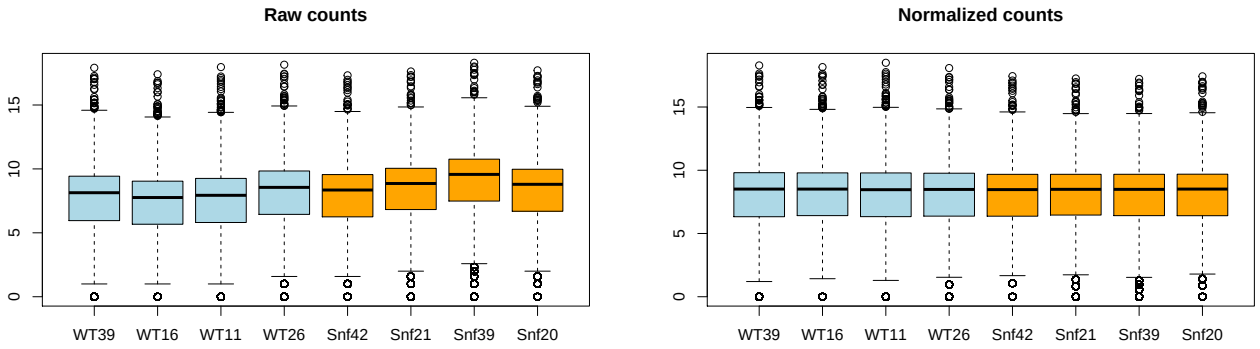
We now perform a differential analysis with DESeq2 step by step with some quality controls.

Normalization

```
dds <- estimateSizeFactors(dds)
print(sizeFactors(dds))
```

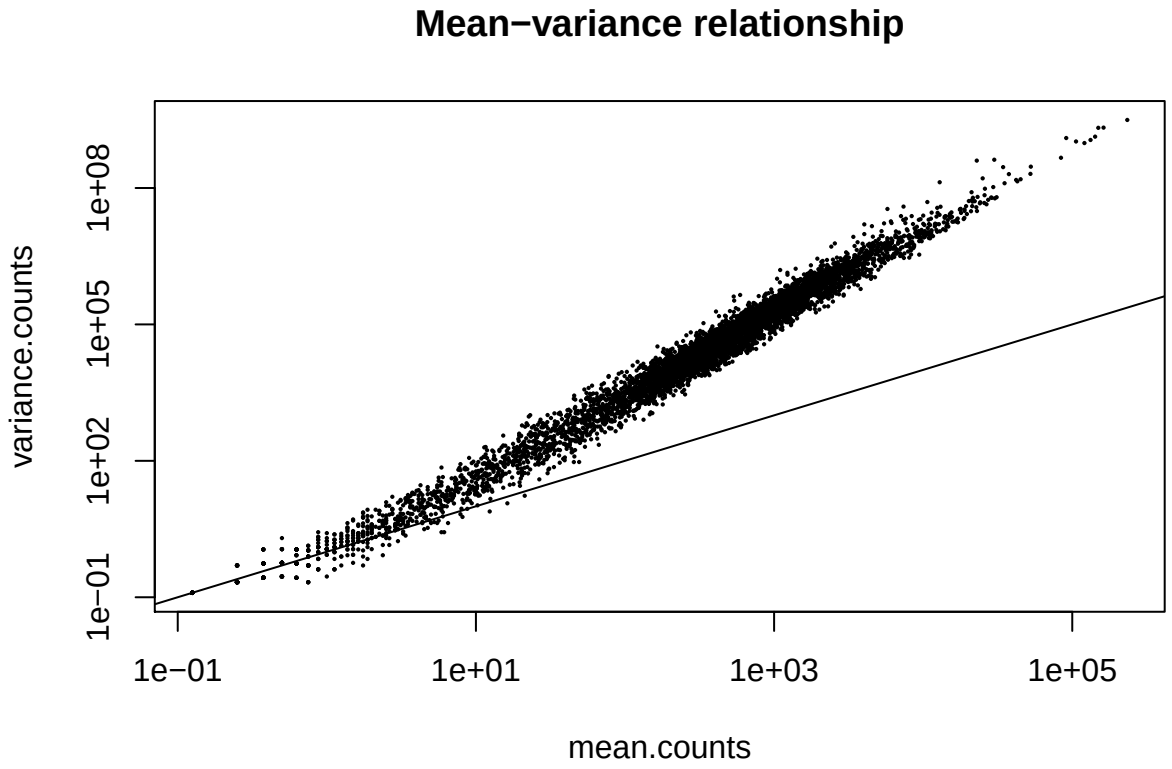
```
##      WT39      WT16      WT11      WT26     Snf42     Snf21     Snf39     Snf20
## 0.7707961 0.5939215 0.6918722 1.0510759 0.9197833 1.2906255 2.1175191 1.2178517
```

```
# effect of the normalization
par(mfrow=c(1,2))
boxplot(log2(counts(dds, normalized=FALSE)+1), main="Raw counts", col=rep(c("lightblue","orange"), each=
boxplot(log2(counts(dds, normalized=TRUE)+1), main="Normalized counts", col=rep(c("lightblue","orange")
```



Mean-variance relationship

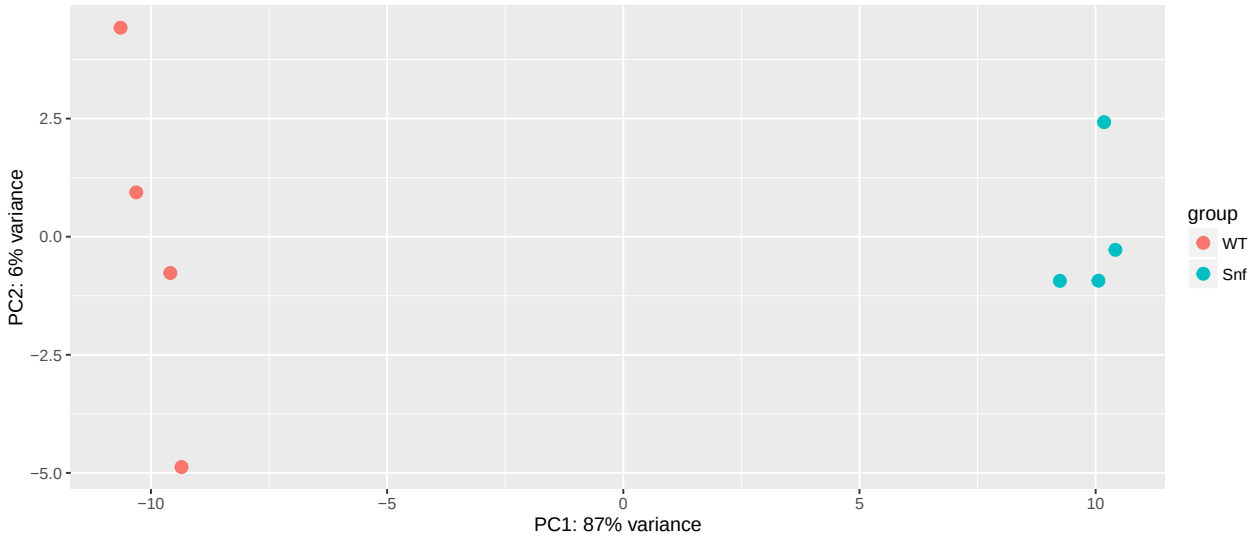
```
mean.counts <- rowMeans(counts(dds))
variance.counts <- apply(counts(dds), 1, var)
plot(x=mean.counts, y=variance.counts, pch=16, cex=0.3, main="Mean-variance relationship", log="xy")
abline(a=0, b=1)
```



We observe over-dispersion in the data: the Poisson distribution is not adapted and we prefer the Negative-Binommial distribution.

Principal Component Analysis (PCA)

```
# dispersions estimation
dds <- estimateDispersions(dds)
# make the data homoscedastic
rld <- rlog(dds) # alternative to the "Variance Stabilizing Transformation"
plotPCA(rld, intgroup="strain") # function from the DESeq2 package
```



Statistical test for each gene

```
dds <- nbinomWaldTest(dds)
res.DESeq2 <- results(dds, alpha=0.05, pAdjustMethod="BH")
print(head(res.DESeq2))
```

```
## log2 fold change (MAP): strain Snf vs WT
## Wald test p-value: strain Snf vs WT
## DataFrame with 6 rows and 6 columns
##            baseMean log2FoldChange     lfcSE        stat      pvalue        padj
##           <numeric>      <numeric> <numeric>   <numeric>   <numeric>   <numeric>
## 15s_rrna  22.781440    -0.82933021 0.4235659 -1.95797224 0.050233270 0.102545913
## 21s_rrna 142.090551    -1.35107533 0.4492520 -3.00738832 0.002635029 0.008522267
## hra1       2.202562     0.03609404 0.4401305  0.08200758 0.934640683 0.958309870
## icr1     131.652184     0.32454269 0.1557189  2.08415772 0.037145823 0.080488753
## lsr1     198.397501    -0.70266582 0.4354260 -1.61374338 0.106583100 0.192054548
## nme1      24.847267    -0.26927523 0.3750650 -0.71794286 0.472792519 0.602847226
```

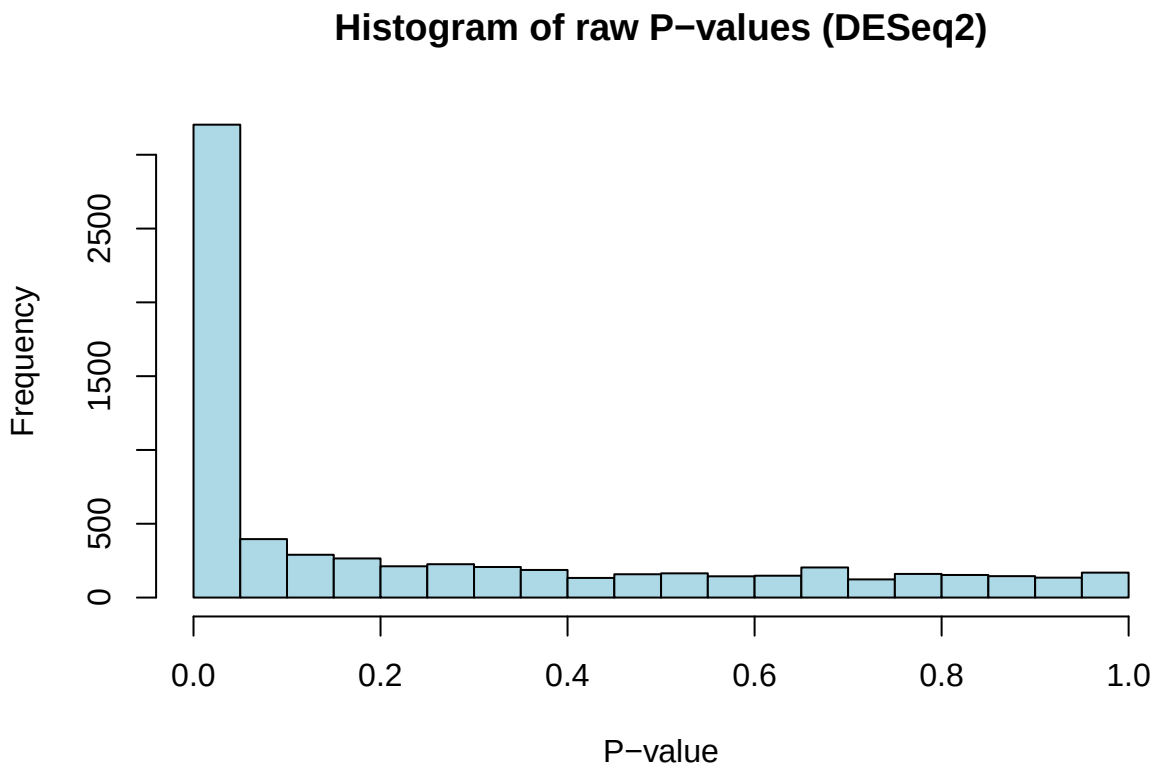
```
summary(res.DESeq2, alpha=0.05)
```

```
##
```

```
## out of 6823 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)      : 1275, 19%
## LFC < 0 (down)    : 1469, 22%
## outliers [1]      : 0, 0%
## low counts [2]    : 264, 3.9%
## (mean count < 1)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

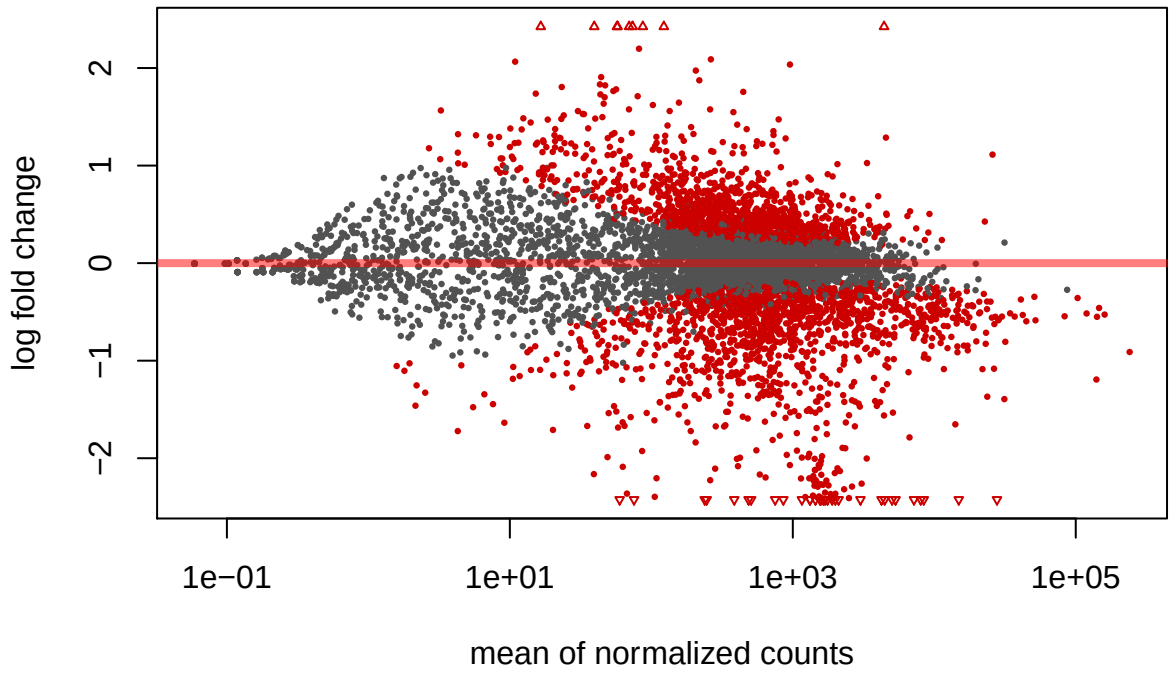
Histogram of raw P-values

```
hist(res.DESeq2$pvalue, col="lightblue", main="Histogram of raw P-values (DESeq2)", breaks=20, xlab="P-
```



MA-plot

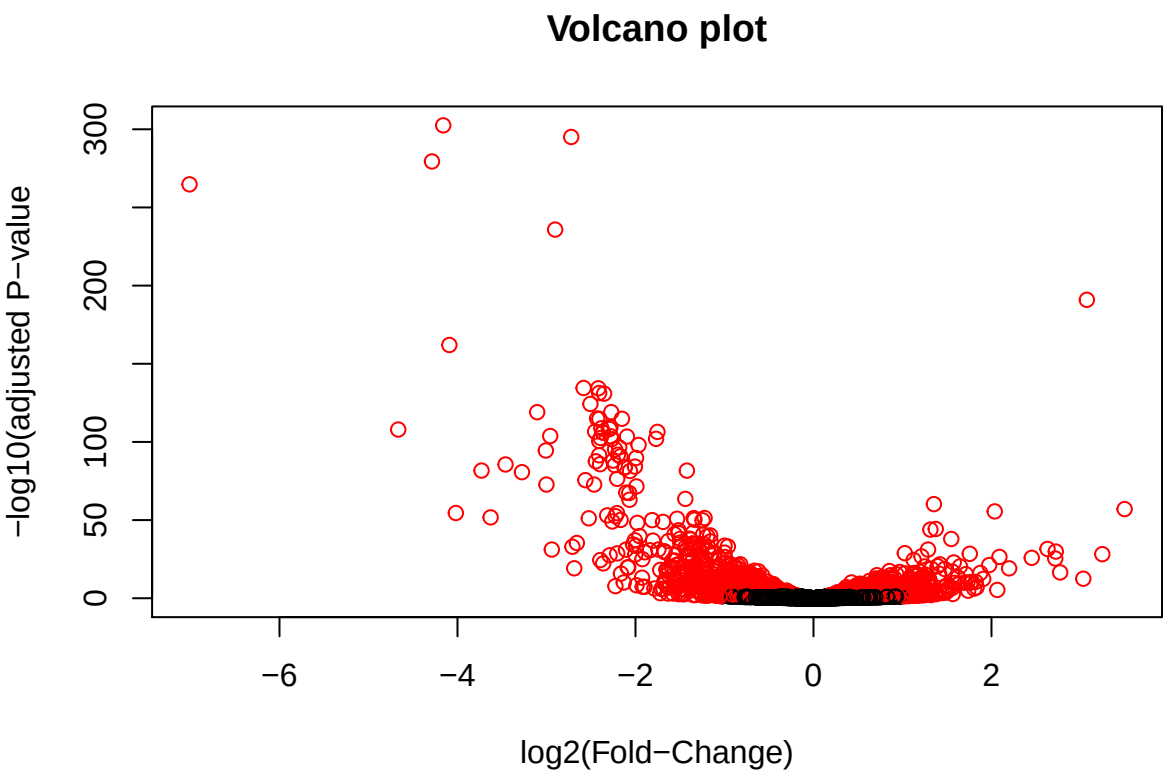
```
plotMA(res.DESeq2, alpha=0.05) # function from the DESeq2 package
```



Volcano-plot

Here we need to build the plot using R base functions:

```
plot(x=res.DESeq2$log2FoldChange, y=-log10(res.DESeq2$padj),
     xlab="log2(Fold-Change)", ylab="-log10(adjusted P-value",
     col=ifelse(res.DESeq2$padj<=0.05, "red", "black"), main="Volcano plot")
```



Differential analysis using edgeR with a few replicates

```
# load the edgeR R package
library(edgeR)
# create the edgeR main object: dge
dge <- DGEList(counts=counts[,c(samples.WT, samples.Snf2)], remove.zeros=FALSE)
dge$design <- model.matrix(~ strain, data=expDesign[c(samples.WT, samples.Snf2),])
print(dge)
```

```
## An object of class "DGEList"
## $counts
##          WT39 WT16 WT11 WT26 Snf42 Snf21 Snf39 Snf20
## 15s_rrna    0   29   27   74     1    20     7     5
## 21s_rrna   10  221  194  352    13    88    64    29
## hra1        5    0    0    2     2     5     5     1
## icr1       85   70   91  111   164   188   286   157
## lsr1       66  314   65  401    57   257   250   144
## 7121 more rows ...
##
## $samples
##       group lib.size norm.factors
## WT39      1  7408720            1
## WT16      1  5569418            1
## WT11      1  6795757            1
## WT26      1  9231334            1
## Snf42     1  6894818            1
```

```
## Snf21     1  9531827            1
## Snf39     1 15182957            1
## Snf20     1  9107888            1
##
## $design
##    (Intercept) strainSnf
## 39           1         0
## 16           1         0
## 11           1         0
## 26           1         0
## 90           1         1
## 69           1         1
## 87           1         1
## 68           1         1
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
## attr(,"contrasts")$strain
## [1] "contr.treatment"
```

Normalization & dispersions estimation with edgeR

```
# normalization
dge <- calcNormFactors(dge)
print(dge$samples$norm.factors)
```

```
## [1] 0.8759759 0.8941748 0.8523457 0.9456989 1.1046557 1.1246171 1.1542386 1.1045606
```

```
# dispersions
dge <- estimateGLMCommonDisp(dge, dge$design)
dge <- estimateGLMTrendedDisp(dge, dge$design)
dge <- estimateGLMTagwiseDisp(dge, dge$design)
```

Modeling and testing with edgeR

```
fit <- glmFit(dge, dge$design)
print(dge$design)
```

```
##    (Intercept) strainSnf
## 39           1         0
## 16           1         0
## 11           1         0
## 26           1         0
## 90           1         1
## 69           1         1
## 87           1         1
## 68           1         1
```

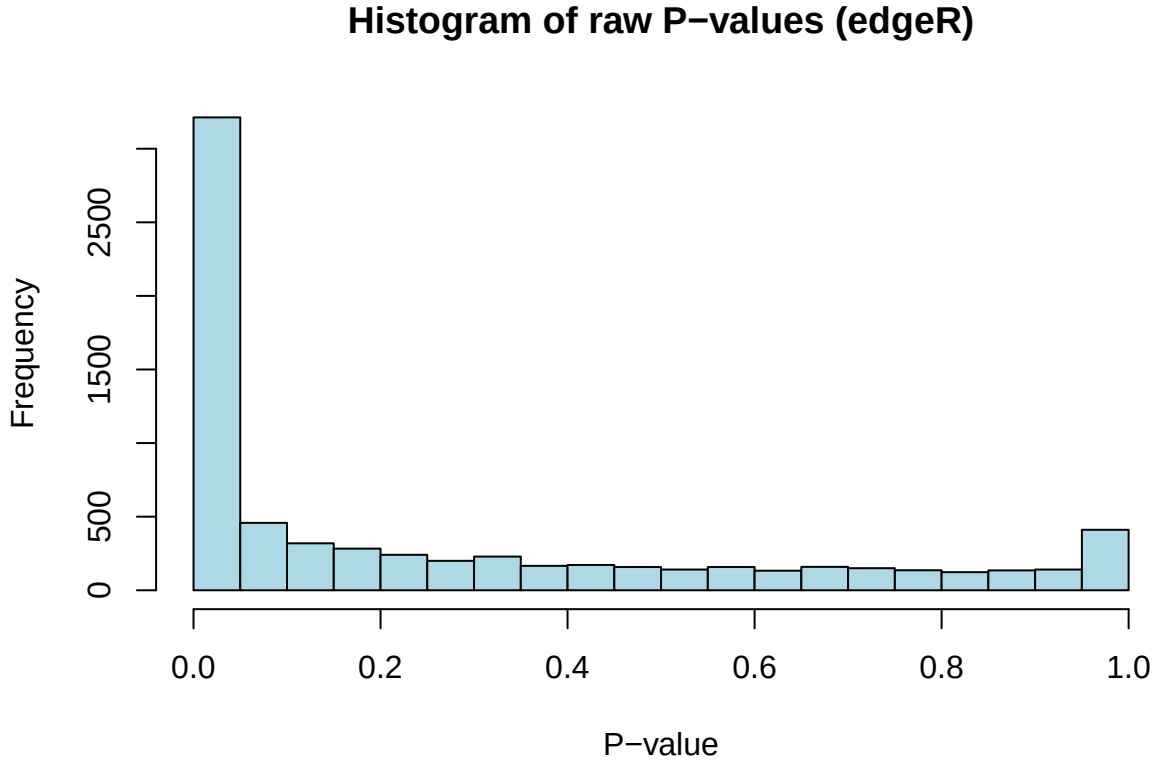
```
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
## attr(,"contrasts")$strain
## [1] "contr.treatment"
```

```
lrt <- glmLRT(fit)
res.edgeR <- topTags(lrt,n=nrow(dge$counts),adjust.method="BH")$table
print(head(res.edgeR))
```

```
##             logFC   logCPM        LR        PValue           FDR
## yor290c -7.411259 6.459658 1546.9969  0.000000e+00  0.000000e+00
## yml123c -4.670965 9.227046 1375.6127 4.186650e-301 1.491703e-297
## yhr215w -4.664998 7.289340 1312.1724 2.558596e-287 6.077518e-284
## yar071w -4.347829 7.701766  977.7045 1.260534e-214 2.245641e-211
## ygr234w -4.209588 8.482723  972.8215 1.452056e-213 2.069471e-210
## ydr033w -3.849749 8.979987  948.0502 3.520406e-208 4.181069e-205
```

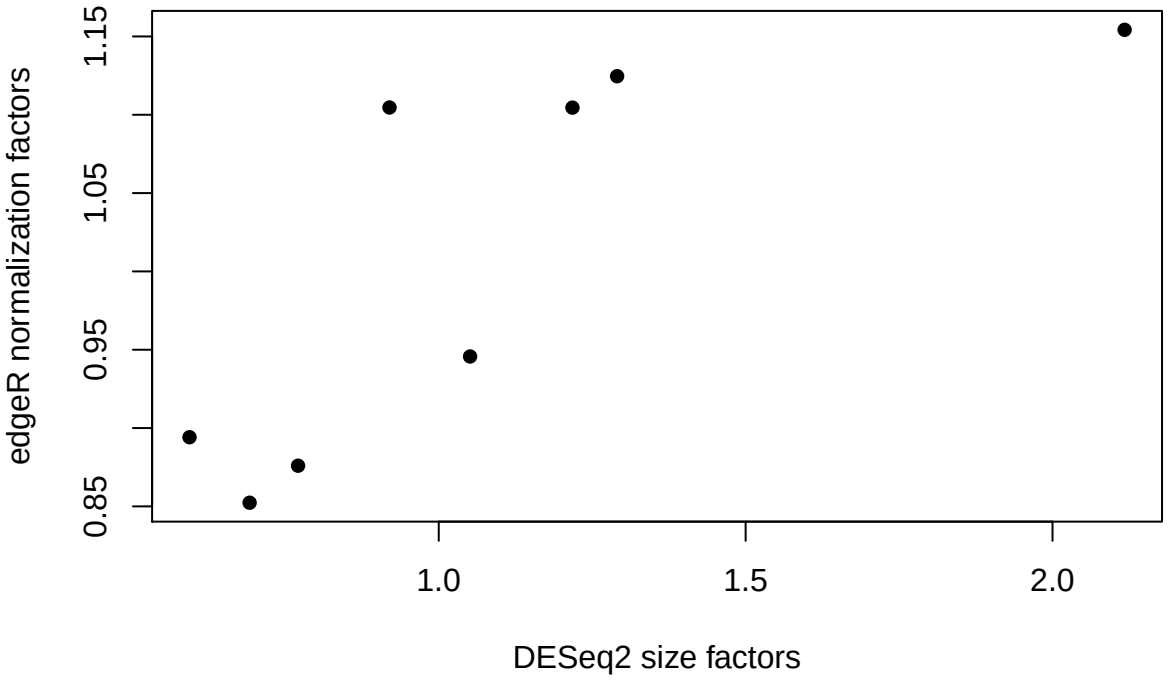
Histogram of raw P-values

```
hist(res.edgeR$PValue, col="lightblue", main="Histogram of raw P-values (edgeR)", breaks=20, xlab="P-val
```



Compare DESeq2 and edgeR results: normalization factors

```
plot(x=sizeFactors(dds), y=dge$samples$norm.factors, xlab="DESeq2 size factors", ylab="edgeR normalizat
```



The normalization/size factors computed by DESeq2 and edgeR are not comparable as they are used in a different manner in the statistical/mathematical models.

Re-order the results according to the gene names

```
print(head(res.DESeq2))
```

```
## log2 fold change (MAP): strain Snf vs WT
## Wald test p-value: strain Snf vs WT
## DataFrame with 6 rows and 6 columns
##            baseMean log2FoldChange     lfcSE        stat      pvalue        padj
##           <numeric>      <numeric> <numeric>   <numeric>   <numeric>   <numeric>
## 15s_rrna  22.781440    -0.82933021 0.4235659 -1.95797224 0.050233270 0.102545913
## 21s_rrna 142.090551    -1.35107533 0.4492520 -3.00738832 0.002635029 0.008522267
## hra1       2.202562     0.03609404 0.4401305  0.08200758 0.934640683 0.958309870
## icr1     131.652184     0.32454269 0.1557189  2.08415772 0.037145823 0.080488753
## lsr1     198.397501    -0.70266582 0.4354260 -1.61374338 0.106583100 0.192054548
## nme1      24.847267    -0.26927523 0.3750650 -0.71794286 0.472792519 0.602847226
```

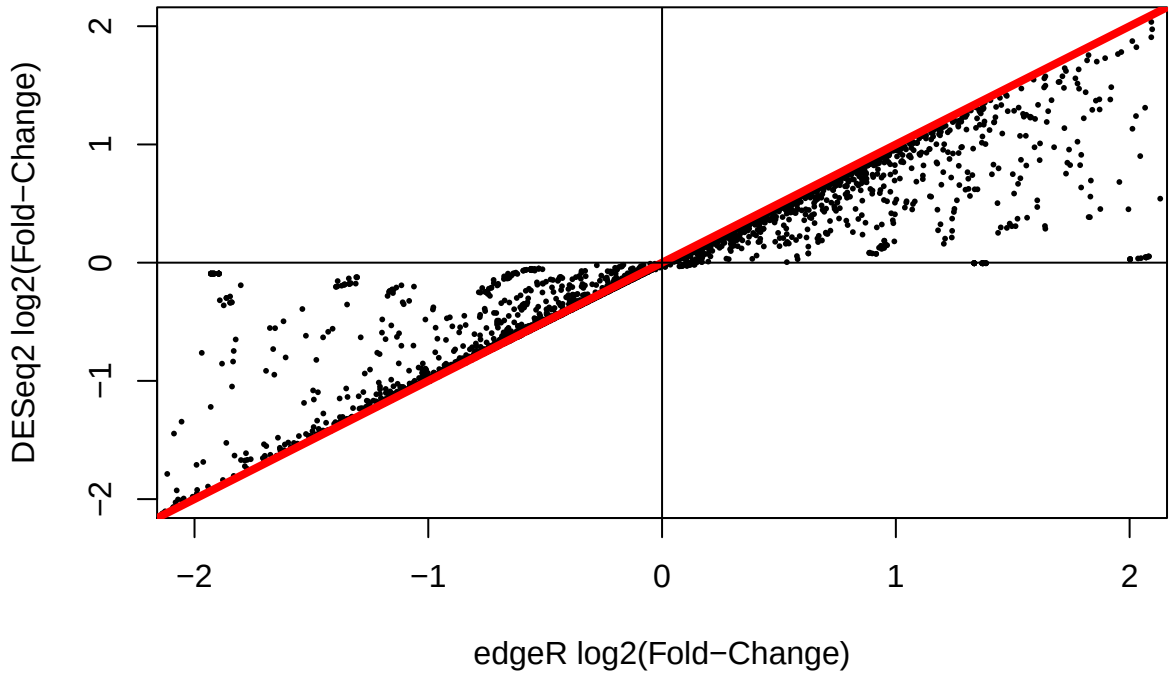
```
print(head(res.edgeR))
```

```
##             logFC   logCPM        LR        PValue           FDR
## yor290c -7.411259 6.459658 1546.9969  0.000000e+00  0.000000e+00
## yml123c -4.670965 9.227046 1375.6127 4.186650e-301 1.491703e-297
## yhr215w -4.664998 7.289340 1312.1724 2.558596e-287 6.077518e-284
## yar071w -4.347829 7.701766  977.7045 1.260534e-214 2.245641e-211
## ygr234w -4.209588 8.482723  972.8215 1.452056e-213 2.069471e-210
## ydr033w -3.849749 8.979987  948.0502 3.520406e-208 4.181069e-205
```

```
res.edgeR <- res.edgeR[order(rownames(res.edgeR)),]
res.DESeq2 <- res.DESeq2[order(rownames(res.DESeq2)),]
```

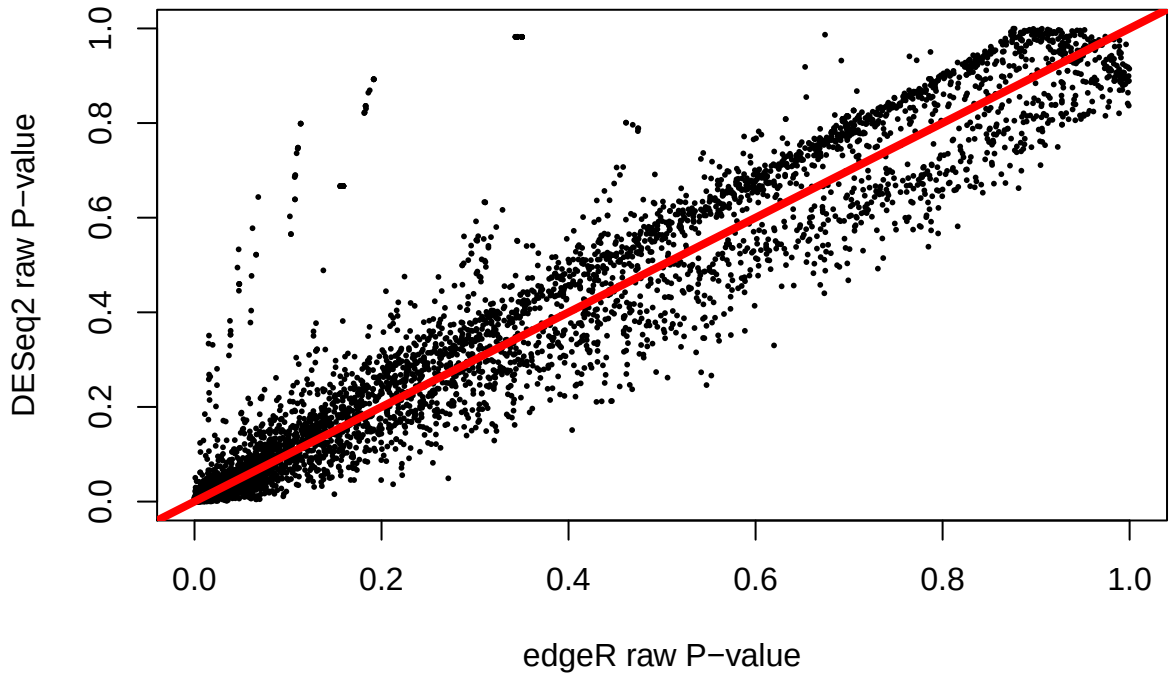
Comparing log2(Fold-Change) estimations

```
plot(x=res.edgeR$logFC, y=res.DESeq2$log2FoldChange,
     pch=16, cex=0.4, xlim=c(-2,2), ylim=c(-2,2),
     xlab="edgeR log2(Fold-Change)", ylab="DESeq2 log2(Fold-Change)")
abline(a=0, b=1, col="red", lwd=4) # draw the y=x curve (y=a+b*x with a=0 and b=1)
abline(h=0, v=0) # horizontal and vertical line
```



Comparing raw P-values

```
plot(x=res.edgeR$PValue, y=res.DESeq2$pvalue,
     pch=16, cex=0.4, xlab="edgeR raw P-value", ylab="DESeq2 raw P-value")
abline(a=0, b=1, col="red", lwd=4) # draw the y=x curve (y=a+b*x with a=0 and b=1)
```



Number of differentially expressed genes

```
# remember the number of replicates
print(nb.replicates)
```

```
## [1] 4
```

```
# DESeq2
sum(res.DESeq2$padj <= 0.05, na.rm=TRUE)
```

```
## [1] 2744
```

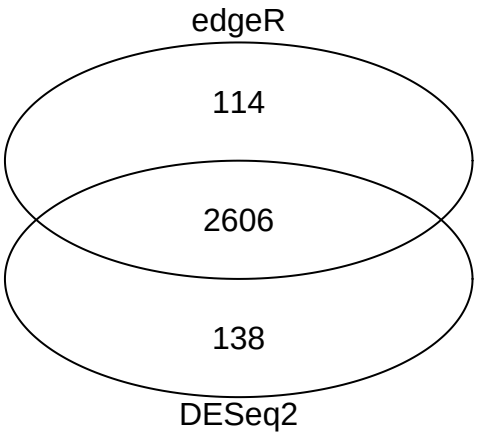
```
# edgeR
sum(res.edgeR$FDR <= 0.05, na.rm=TRUE)
```

```
## [1] 2720
```

What's the behaviour of the number of differentially expressed genes according to the number of samples?

Venn diagram

```
library(gplots)
venn(list(DESeq2=rownames(res.DESeq2[which(res.DESeq2$padj <= 0.05),]),
          edgeR=rownames(res.edgeR[which(res.edgeR$FDR <= 0.05),])))
```



Supplementary exercise: do the same plot for up- and down-regulated genes separately.

What's edgeR or DESeq2-specific?

```
DESeq2.genes <- rownames(res.DESeq2[which(res.DESeq2$padj <= 0.05),])
edgeR.genes <- rownames(res.edgeR[which(res.edgeR$FDR <= 0.05),])
# select a DESeq2 specific gene
spe.DESeq2 <- setdiff(DESeq2.genes, edgeR.genes)
summary(res.edgeR[spe.DESeq2,"FDR"])
```

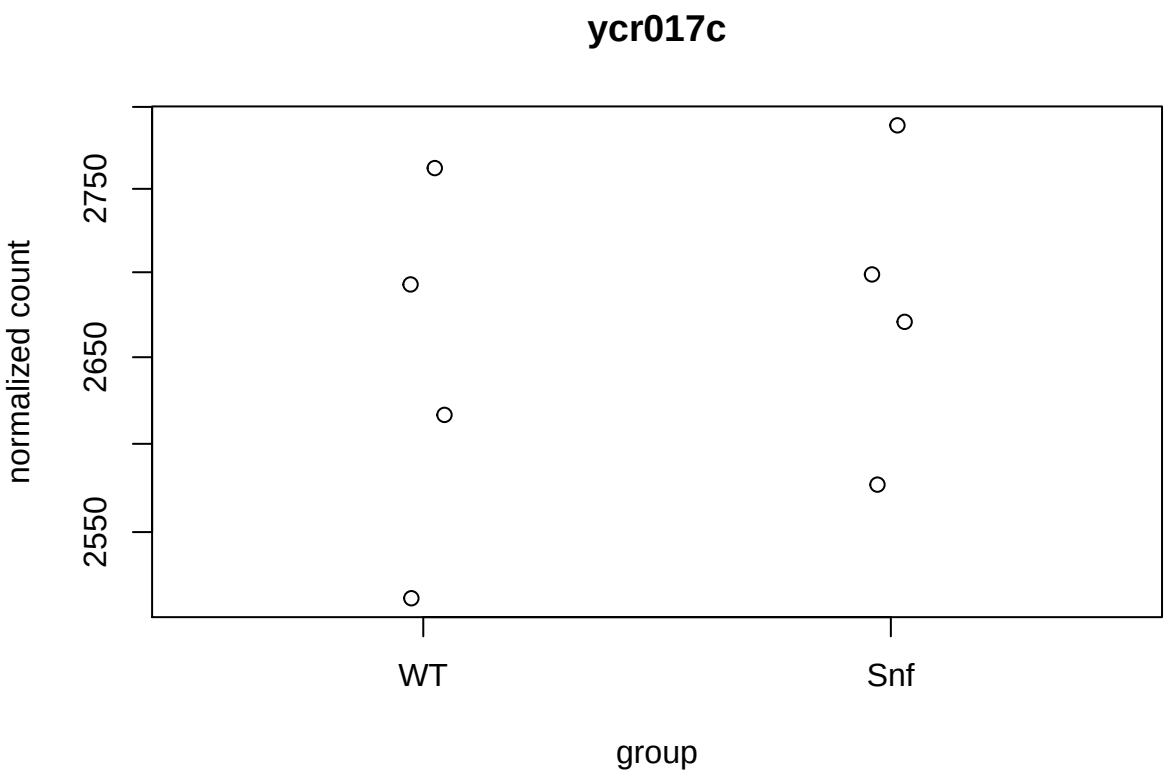
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.05001 0.05968 0.07061 0.08316 0.09569 0.26720
```

```
# select a edgeR specific gene
spe.edgeR <- setdiff(edgeR.genes, DESeq2.genes)
summary(res.DESeq2[spe.edgeR,"padj"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
## 0.05005 0.05548 0.06814 0.07795 0.09077 0.19210      11
```

DESeq2 results for one gene

```
# plotCounts is a function from the DESeq2 R package
plotCounts(dds, gene="ycr017c", intgroup="strain", normalized=TRUE)
```



Differential analysis under H_0

Here we perform a differential analysis in which we compare N WT samples vs N other WT samples.

```
nb.replicates <- 10
samples.WT <- sample(1:48, size=2*nb.replicates, replace=FALSE)
print(samples.WT)
```

```
##  [1] 20  1 42 37 10  2 15  7 26 31 39 11  3 13 24 38 45 46  8 36
```

```
counts.H0 <- counts[,samples.WT]
expDesign.H0 <- expDesign[samples.WT,]
# add a fictive condition factor
expDesign.H0$condition <- factor(rep(c("A","B"), each=nb.replicates))
print(expDesign.H0)
```

```
##    label strain condition
## 20  WT20     WT         A
## 1    WT1     WT         A
## 42  WT42     WT         A
## 37  WT37     WT         A
## 10  WT10     WT         A
## 2    WT2     WT         A
## 15  WT15     WT         A
## 7    WT7     WT         A
## 26  WT26     WT         A
```

```
## 31  WT31     WT         A
## 39  WT39     WT         B
## 11  WT11     WT         B
## 3    WT3     WT         B
## 13  WT13     WT         B
## 24  WT24     WT         B
## 38  WT38     WT         B
## 45  WT45     WT         B
## 46  WT46     WT         B
## 8    WT8     WT         B
## 36  WT36     WT         B
```

```
dds.H0 <- DESeqDataSetFromMatrix(countData = counts.H0, colData = expDesign.H0, design = ~ condition)
```

Differential analysis with DESeq2 under H_0

```
dds.H0 <- DESeq(dds.H0)
res.H0 <- results(dds.H0)
summary(res.H0)
```

```
##
## out of 6862 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)     : 0, 0%
## LFC < 0 (down)   : 0, 0%
## outliers [1]     : 0, 0%
## low counts [2]   : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

sessionInfo

Here are the details of the R packages used to generate this document:

```
sessionInfo()
```

```
## R version 3.3.1 (2016-06-21)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.12.1 (Sierra)
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
```

```
##  [1] gplots_3.0.1              edgeR_3.14.0              limma_3.28.21             DESeq2_1.12.4
## [11] BiocGenerics_0.18.0
##
## loaded via a namespace (and not attached):
##  [1] genefilter_1.54.2    gtools_3.5.0         locfit_1.5-9.1       splines_3.3.1        lattice_0.20
## [15] RColorBrewer_1.1-2   plyr_1.8.4           stringr_1.1.0        zlibbioc_1.18.0      munsell_0.4
## [29] Rcpp_0.12.7          KernSmooth_2.23-15   acepack_1.4.1        xtable_1.8-2         scales_0.4.0
## [43] grid_3.3.1           tools_3.3.1          bitops_1.0-6         magrittr_1.5         RCurl_1.95-4
## [57] nnet_7.3-12
```