USC Department Of Translational Genomics
Keck Medicine of USC

TRGN 599: Applied Data Science and Bioinformatics

UNIT V. Unsupervised Analysis, linear regression, enrichment analysis

Week 11 - Lecture 2

Enrique I. Velazquez Villarreal, M.D., Ph.D., M.P.H., M.S. | **Assistant Professor**

Dept. of Translational Genomics

USC | Keck School of Medicine | Norris Comprehensive Cancer Center

Leader of the USC Bioinformatics Core – *USC CaRE2 Health Equity Center*

David W. Craig, Ph.D. | **Professor and Vice Chair**

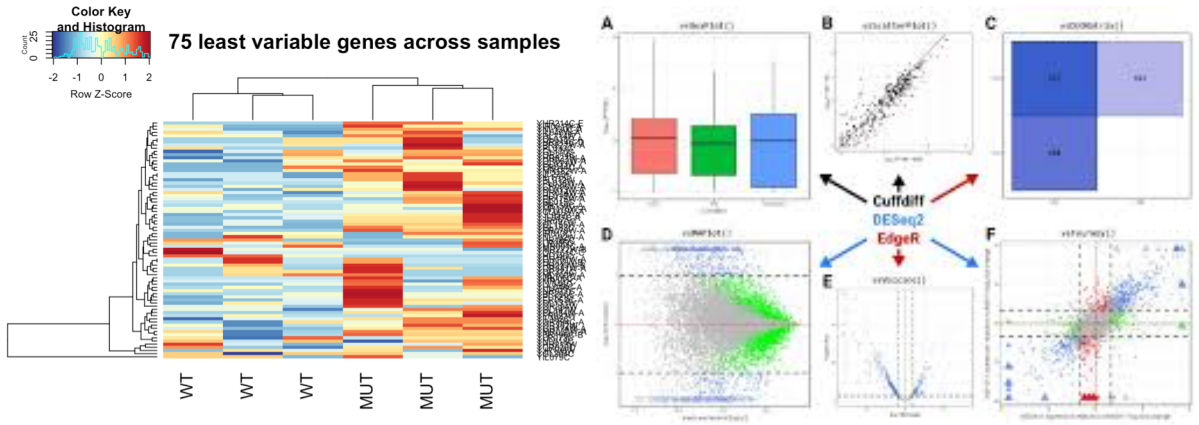
Dept. of Translational Genomics

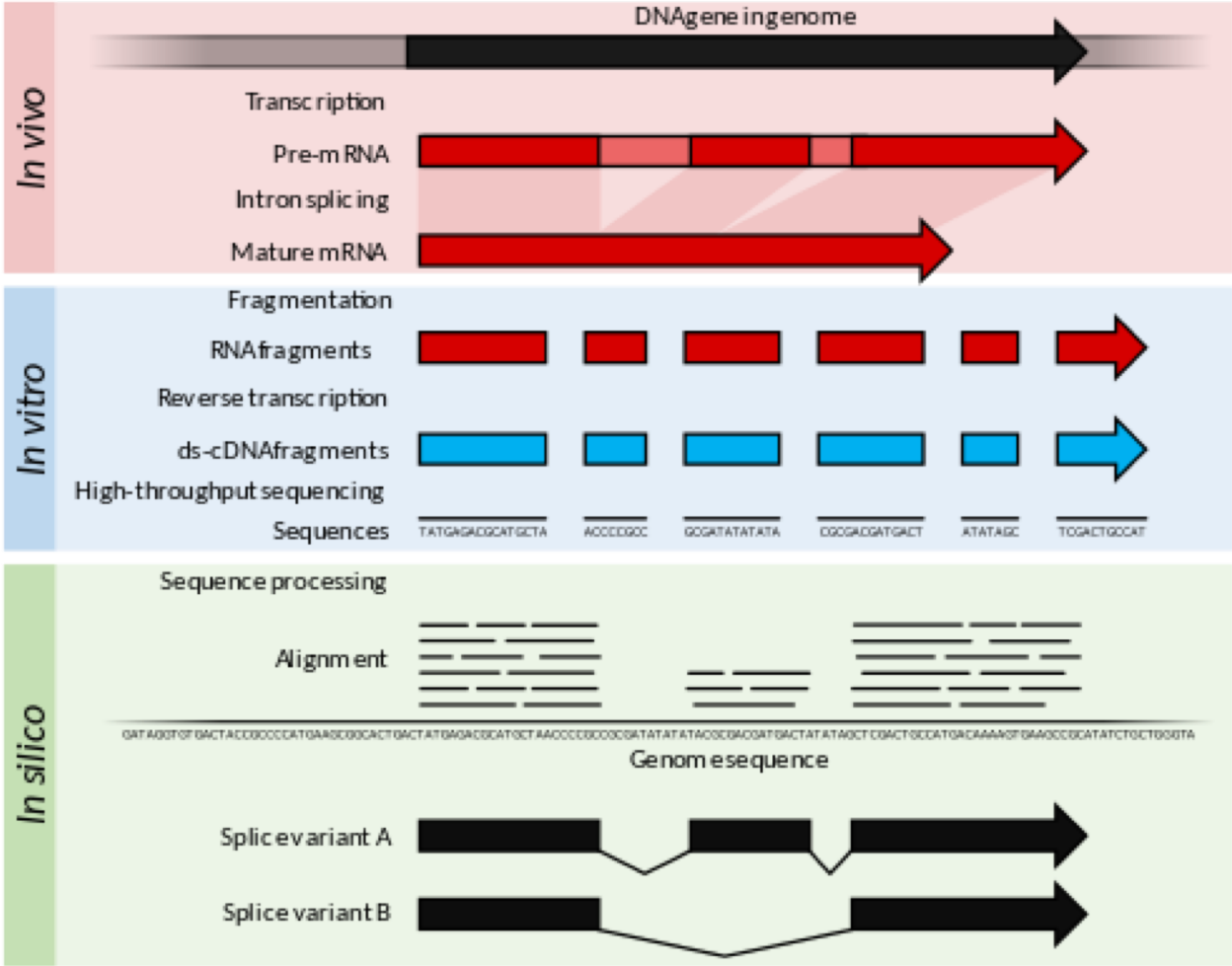
USC | Keck School of Medicine | Norris Comprehensive Cancer Center

Co-Director, Institute of Translational Genomics

Topics

- Next-Generation sequencing transcriptomics





Differential expression with DESeq

- By having different tests from previous class (ttest, edgeR), now the question would be which results should we trust?
- We can use DESeq (Anders and Huber 2010) package to perform the same calculations and investigate how many genes are common on the result lists of the three approaches (ttest, edgeR, DESeq).
- The procedure with DESeq is somewhat similar to that we have done with edgeR.
- It start with the assembling of an appropriately formatted data structure. ttest, edgeR

Differential expression with DESeq

```
# Installing package: https://bioconductor.org/packages/release/bioc/html/DESeq.html

#if (!requireNamespace("BiocManager", quietly = TRUE))
#    install.packages("BiocManager")
#BiocManager::install("DESeq", version = "3.8")

#Installing package lattice
#install.packages("lattice")

library(DESeq)
```

```
## Loading required package: locfit
```

```
## locfit 1.5-9.1    2013-03-22
```

```
## Loading required package: lattice
```

```
##     Welcome to 'DESeq'. For improved performance, usability and
##     functionality, please consider migrating to 'DESeq2'.
```

Differential expression with DESeq

- Unfortunately, the internals of the count.set data structure are masked from us, so the process cannot be followed step by step the same way as with edgeR.

- Still, we have to normalize with the different library sizes.

```
count.set <- newCountDataSet(counts,exp.des)
count.set
```

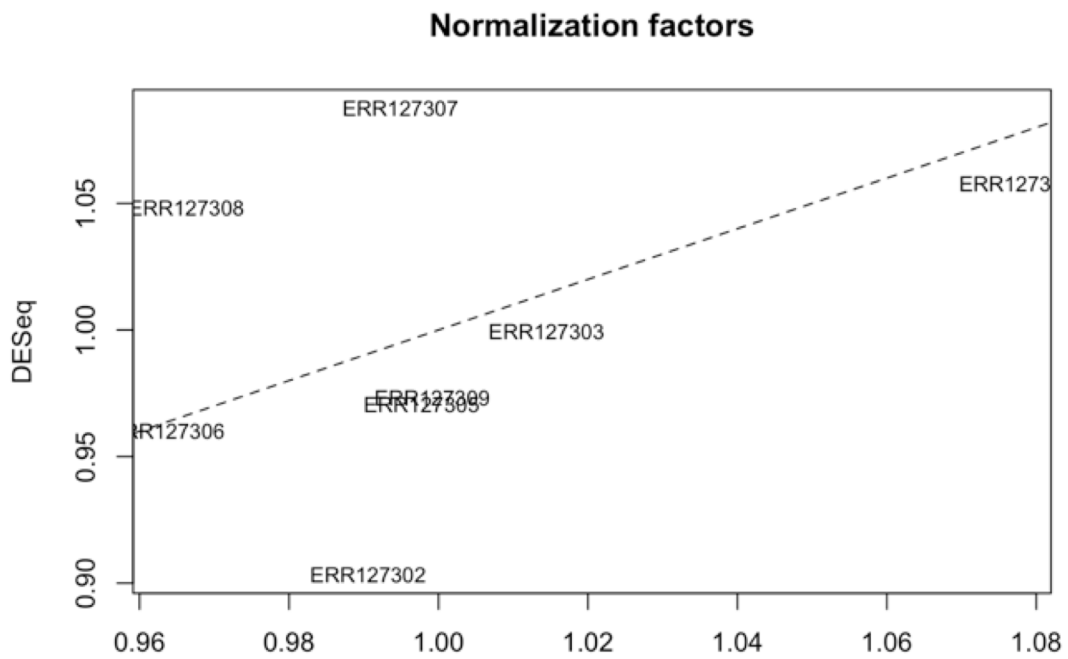
```
## CountDataSet (storageMode: environment)
## assayData: 575 features, 8 samples
##   element names: counts
## protocolData: none
## phenoData
##   sampleNames: ERR127306 ERR127307 ... ERR127305 (8 total)
##   varLabels: sizeFactor condition
##   varMetadata: labelDescription
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation:
```

Differential expression with DESeq

```
count.set <- estimateSizeFactors(count.set)
sizeFactors(count.set)
```

```
## ERR127306 ERR127307 ERR127308 ERR127309 ERR127302 ERR127303 ERR127304
## 0.9601114 1.0875300 1.0482941 0.9733707 0.9033333 0.9994399 1.0579478
## ERR127305
## 0.9707382
```

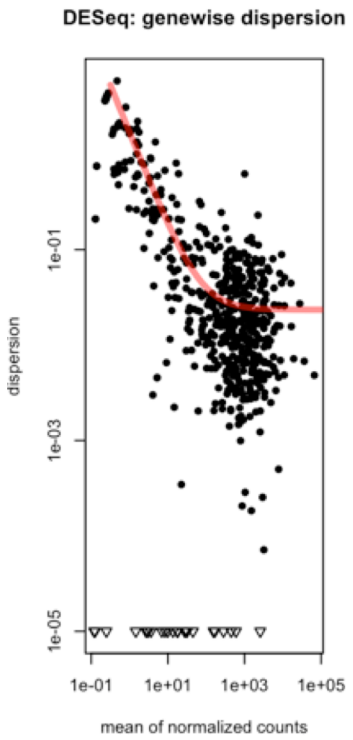
```
plot(genexp$samples$norm.factors,sizeFactors(count.set),xlab="edgeR",ylab="DESeq",main="Normalization factors",ty
pe="n")
abline(0,1,lty=2)
text(genexp$samples$norm.factors,sizeFactors(count.set),labels=names(sizeFactors(count.set)),cex=0.8)
```



Differential expression with DESeq

- The two packages estimate the library size normalization factors rather differently.
- While edgeR uses trimmed means, DESeq employs a relative log expression approach.
- However, the real differences are in their method to calculate common and gene-wide dispersion.

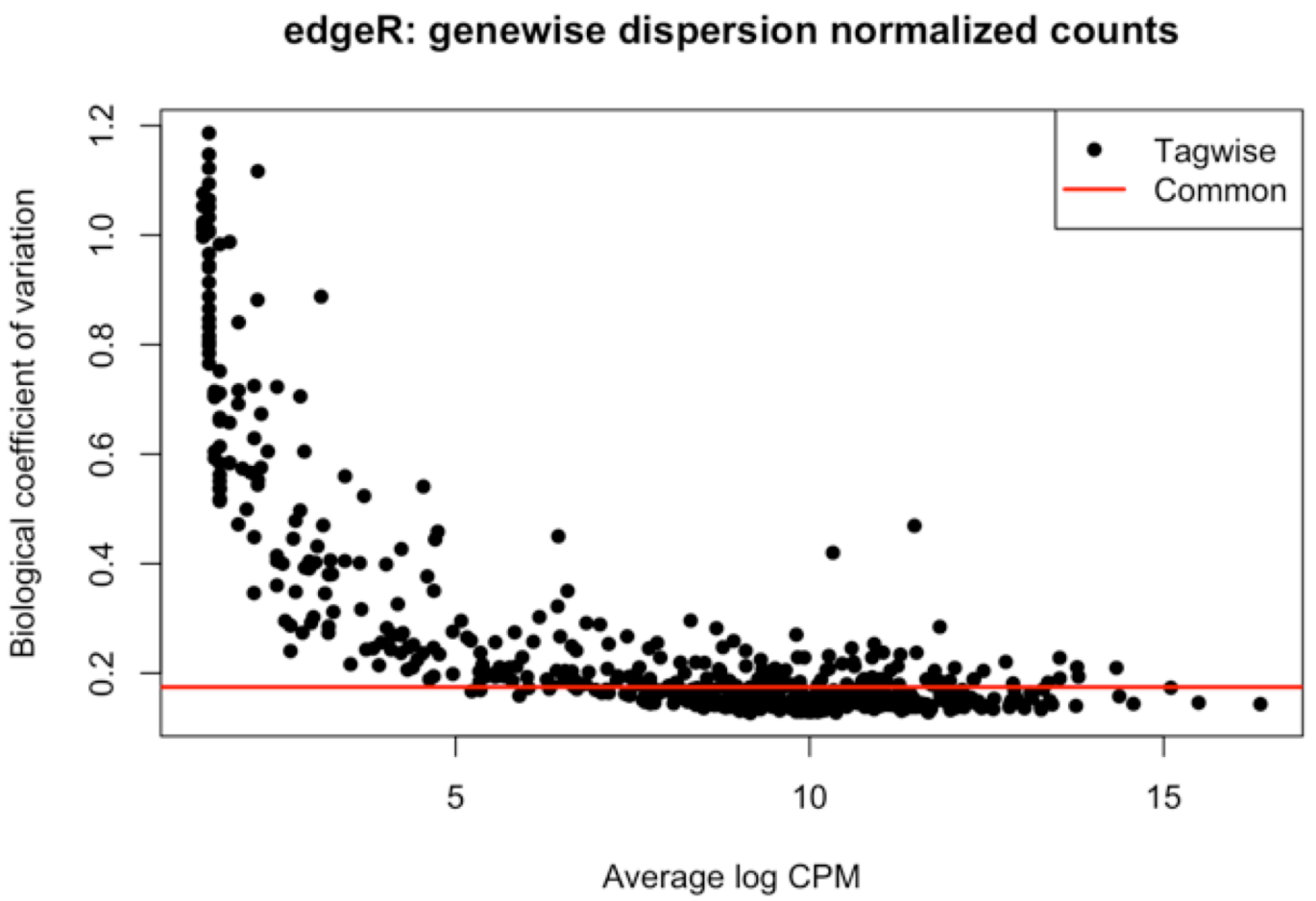
```
# dispersion estimation
count.set <- estimateDispersions(count.set)
par(mfrow=c(1,3))
plotDispEsts(count.set,main="DESeq: genewise dispersion",cex=1)
```



Differential expression with DESeq

- The tag-wide dispersion is called here as gene-wide dispersion:

```
plotBCV(genexp,main="edgeR: genewise dispersion normalized counts",cex=1)
```



Differential expression with DESeq

- While edgeR smooths gene-wide dispersion toward the common dispersion and moderates the differences, DESeq maximizes it if the variability of a gene is larger than the common value.

- In practice, while the edgeR approach is more powerful, it is more sensitive to outliers than DESeq.

- On the general level, both libraries are thought to perform similarly.

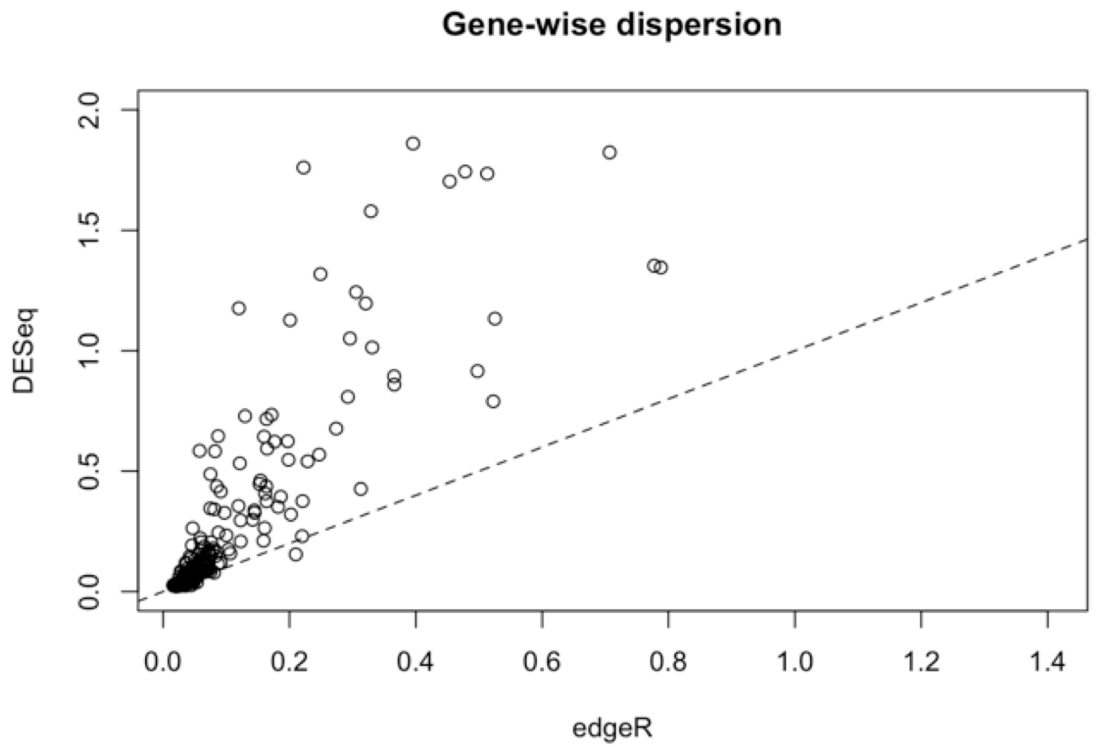
- There is no clear superiority between these approaches.

Differential expression with DESeq

```
# Dispersion values per gene
head(cbind(genexp$tagwise.dispersion,fData(count.set)))
```

```
##          genexp$tagwise.dispersion disp_pooled
## MED6                    0.02161196  0.02858431
## SNORD127                0.18620898  0.39400033
## SNORD126                0.78806405  1.34554840
## SNORA11B                0.08505375  0.43688173
## ZBTB42                  0.02024114  0.02698785
## KTN1-AS1                0.04156770  0.04692908
```

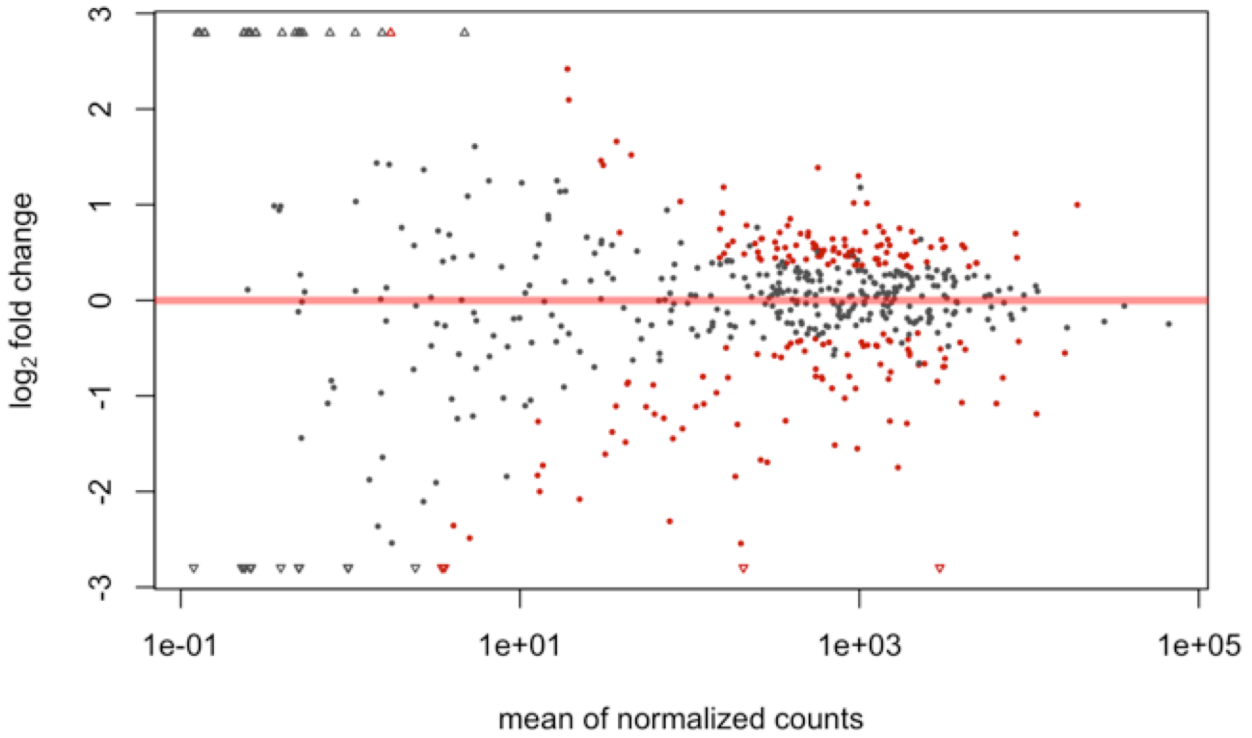
```
plot(cbind(genexp$tagwise.dispersion,fData(count.set)),ylim=c(0,2),main="Gene-wise dispersion",xlab="edgeR",ylab="DESeq")
abline(0,1,lty=2)
```



Differential expression with DESeq

- Finally, let us investigate which are the genes with significantly different expression levels according to DESeq.
- The package provides binomial test-based statistics:

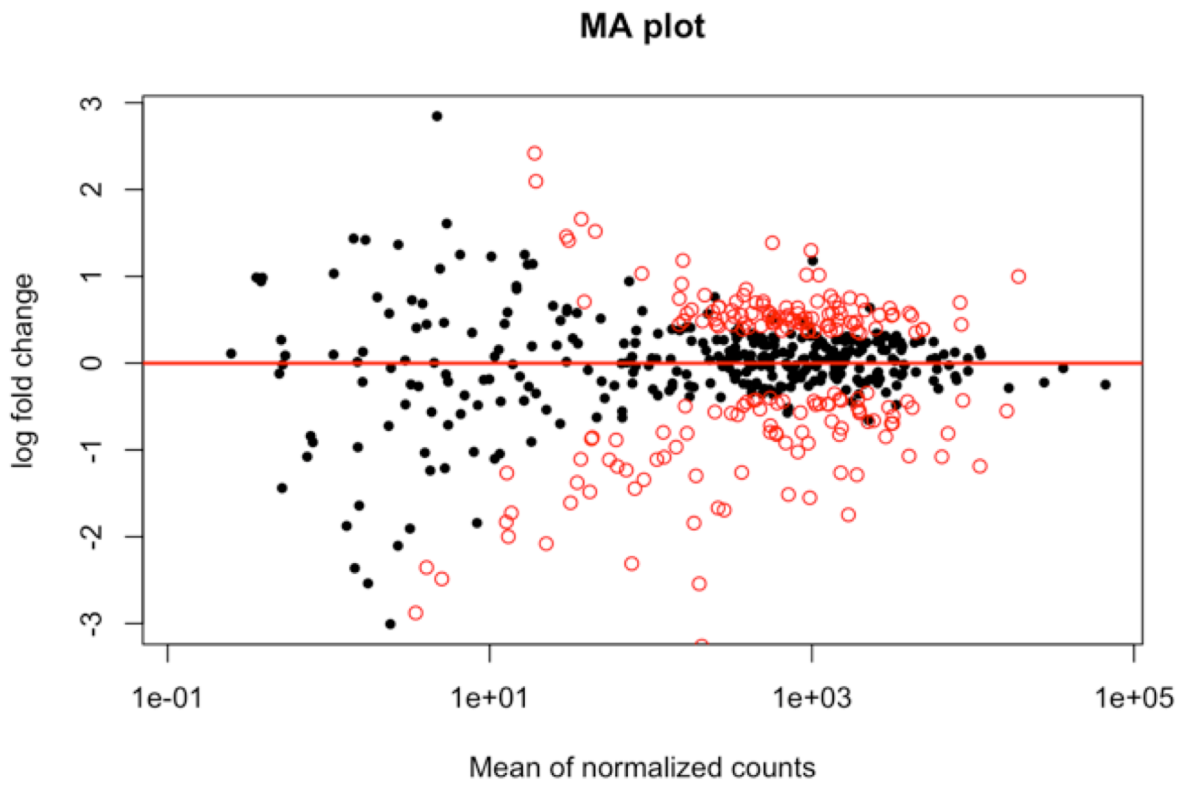
```
# simple - differential expression
genediff.ds <- nbinomTest(count.set,"Control","HNRNPC knockdown")
plotMA(genediff.ds)
```



Differential expression with DESeq

- Finally, let us investigate which are the genes with significantly different expression levels according to DESeq.
- The package provides binomial test-based statistics:

```
plot(log2FoldChange ~ baseMean,data=genediff.ds[genediff.ds$padj>0.1,], log="x",xlab="Mean of normalized counts",
ylab="log fold change", main="MA plot", pch=20)
abline(h=0,lwd=2,col="red")
points(genediff.ds[genediff.ds$padj<0.1, c("baseMean","log2FoldChange")], col="red")
```



Differential expression with DESeq

- The plot in the last slide shows the fold change (y-axis) as a function of mean counts.
- The more left are the points on the plot, the lower are the counts in the calculations, meaning less reliable results.
- Genes with significant differences are highlighted by starts.
- On the right part of the plot, a smaller difference is enough for a gene to be highlighted as significant, while the calculation is tricter for genes with low reads counts (left part of the plot).

Differential expression with DESeq

- It is useful to compare edgeR and DESeq outputs.

```
gsign.ds <- genediff.ds[genediff.ds$padj<0.03,]
gsign.ds <- gsign.ds[order(gsign.ds$padj),]
row.names(gsign.ds)<-gsign.ds$id
```

```
head(gsign.ds,10)
```

```
##                        id   baseMean baseMeanA  baseMeanB foldChange
## HNRNPC             HNRNPC 2971.09165 5698.2563  243.92695  0.0428073
## EXOC3L4           EXOC3L4  207.22965  375.2915   39.16781  0.1043664
## NFATC4             NFATC4  199.94877  341.3155   58.58205  0.1716361
## IRF2BPL           IRF2BPL 1685.14677 2597.0984  773.19510  0.2977150
## ZFHX2               ZFHX2  185.36585  289.8950   80.83675  0.2788484
## ARHGEF40         ARHGEF40  715.09827 1059.3518  370.84470  0.3500675
## LINC00641       LINC00641   76.15007  126.7438   25.55633  0.2016377
## LOC100289511 LOC100289511  261.48481  397.8763  125.09334  0.3144026
## HSPA2               HSPA2 1901.85206 2698.5250 1105.17914  0.4095493
## FOXN3               FOXN3  986.98313  570.1860 1403.78024  2.4619689
##              log2FoldChange          pval           padj
## HNRNPC           -4.545999 1.189734e-129 6.840972e-127
## EXOC3L4          -3.260271  6.497263e-49  1.867963e-46
## NFATC4           -2.542575  1.288882e-32  2.470357e-30
## IRF2BPL          -1.747996  1.288926e-25  1.852831e-23
## ZFHX2            -1.842447  7.329379e-19  8.428786e-17
## ARHGEF40         -1.514295  1.186756e-18  1.137308e-16
## LINC00641        -2.310163  1.025326e-17  8.422319e-16
## LOC100289511     -1.669315  2.655391e-15  1.739799e-13
## HSPA2            -1.287891  2.723164e-15  1.739799e-13
## FOXN3             1.299813  8.219555e-15  4.726244e-13
```

Differential expression with DESeq

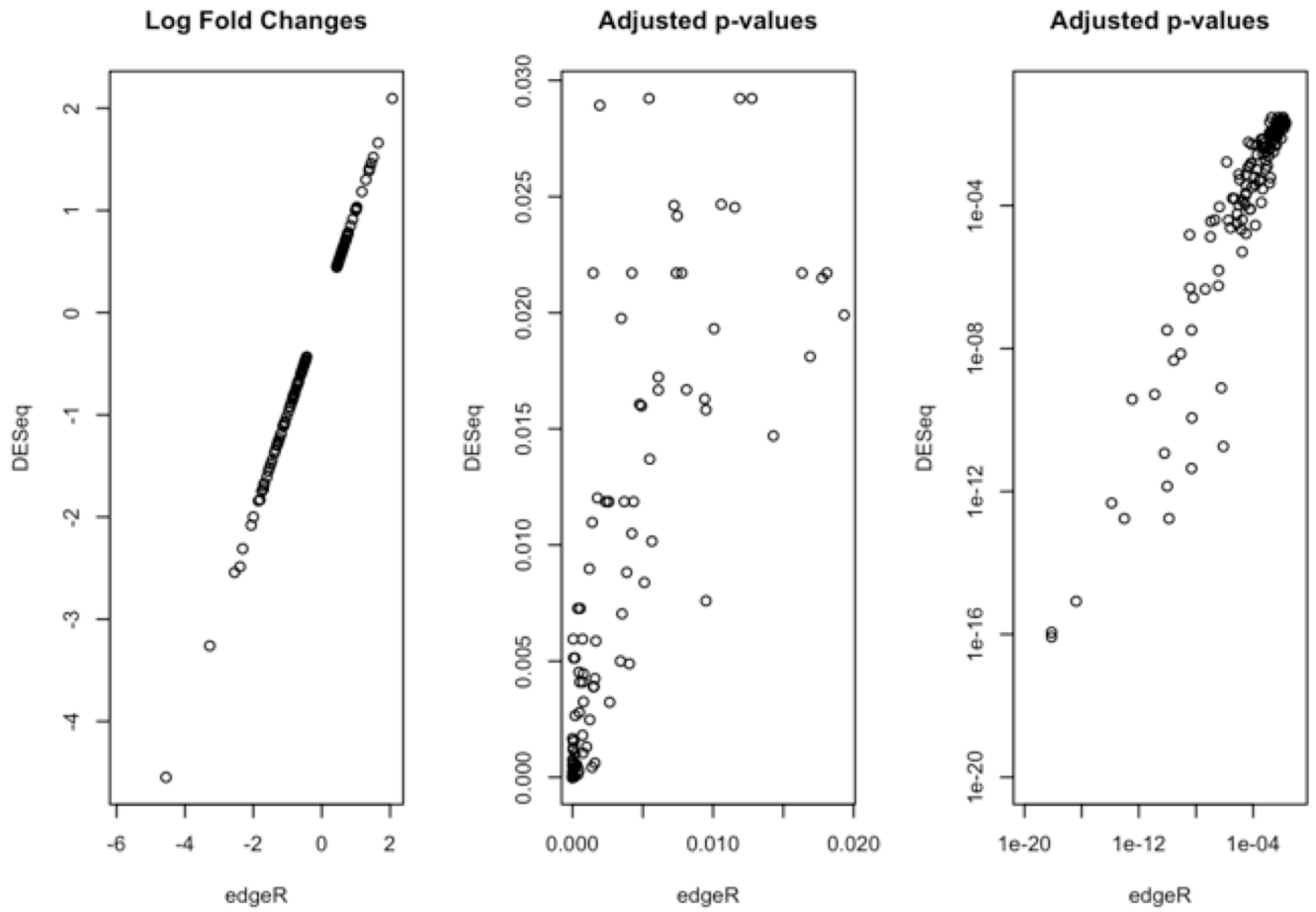
- There are only minimal differences in the order of the 10 most significant genes from the two methods.

- We can see that the slightly different statistic calculation methods of these packages agree nicely with each other.

- Meaning that those genes coming from the t-test calculations can be neglected.

Differential expression with DESeq

```
par(mfrow=c(1,3))
plot(gsign[intersect(row.names(gsign),gsign.ds$id),"logFC"],gsign.ds[intersect(row.names(gsign),gsign.ds$id),"log
2FoldChange"],main="Log Fold Changes",xlab="edgeR",ylab="DESeq")
plot(gsign[intersect(row.names(gsign),gsign.ds$id),"FDR"],gsign.ds[intersect(row.names(gsign),gsign.ds$id),"padj"
],main="Adjusted p-values",xlab="edgeR",ylab="DESeq")
plot(gsign[intersect(row.names(gsign),gsign.ds$id),"FDR"],gsign.ds[intersect(row.names(gsign),gsign.ds$id),"padj"
],main="Adjusted p-values",xlab="edgeR",ylab="DESeq",log="xy",xlim=c(1e-20,1e-1),ylim=c(1e-20,1e-1))
```

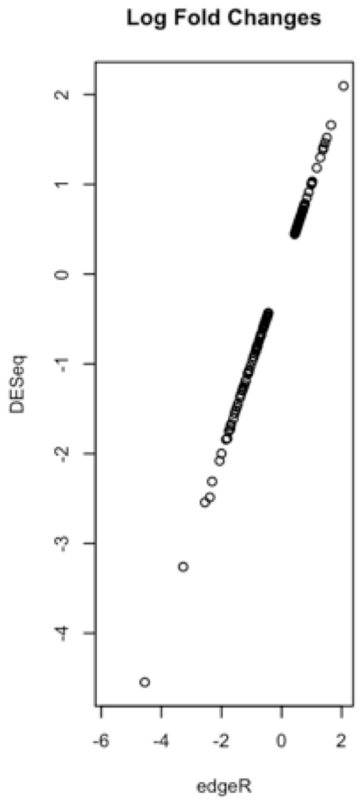


Differential expression with DESeq

- Comparing the common genes from the two calculations reveals that the fold change values correlate greatly.
- The highly significant adjusted p-values (FDR) also agree nicely.
- Only some adjusted p-values (middle panel on the plot) show a degree of difference.
- These observations lead to the conclusion that the edgeR and DESeq packages perform similarly in this example analysis.

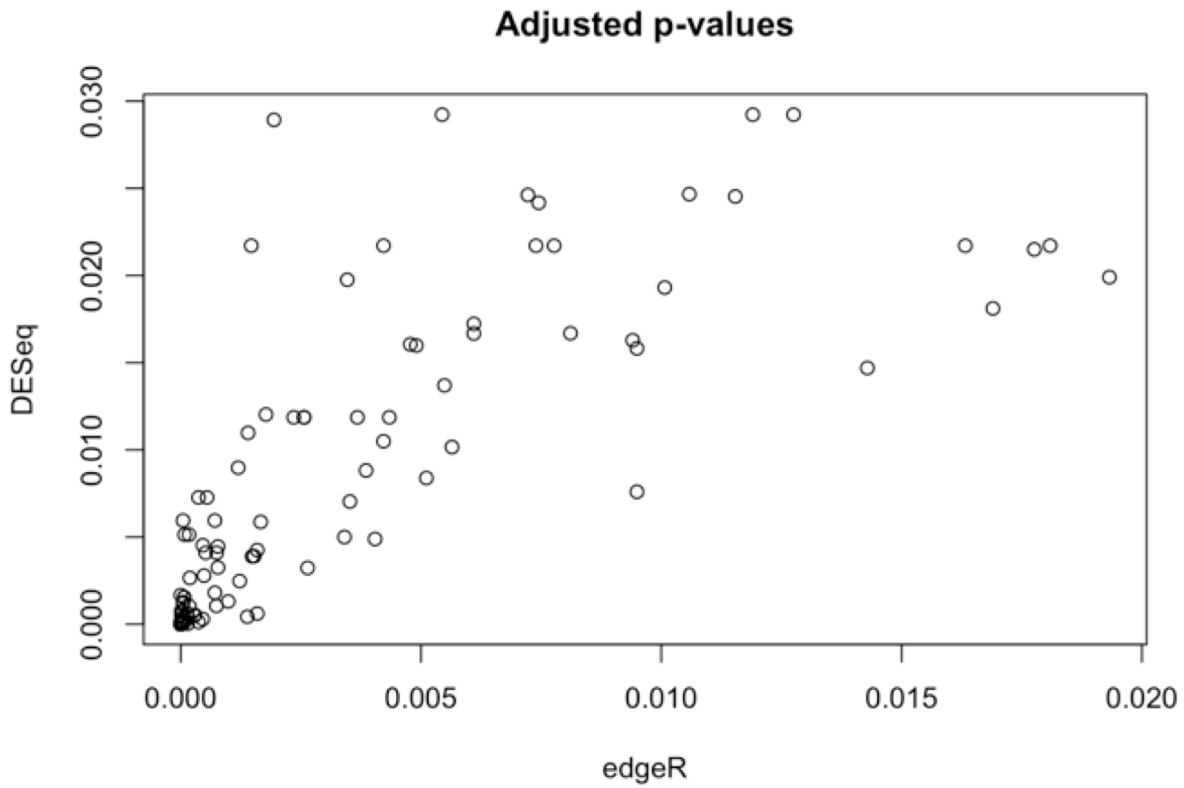
Differential expression with DESeq

```
par(mfrow=c(1,3))
plot(gsign[intersect(row.names(gsign),gsign.ds$id),"logFC"],gsign.ds[intersect(row.names(gsign),gsign.ds$id),"log
2FoldChange"],main="Log Fold Changes",xlab="edgeR",ylab="DESeq")
```



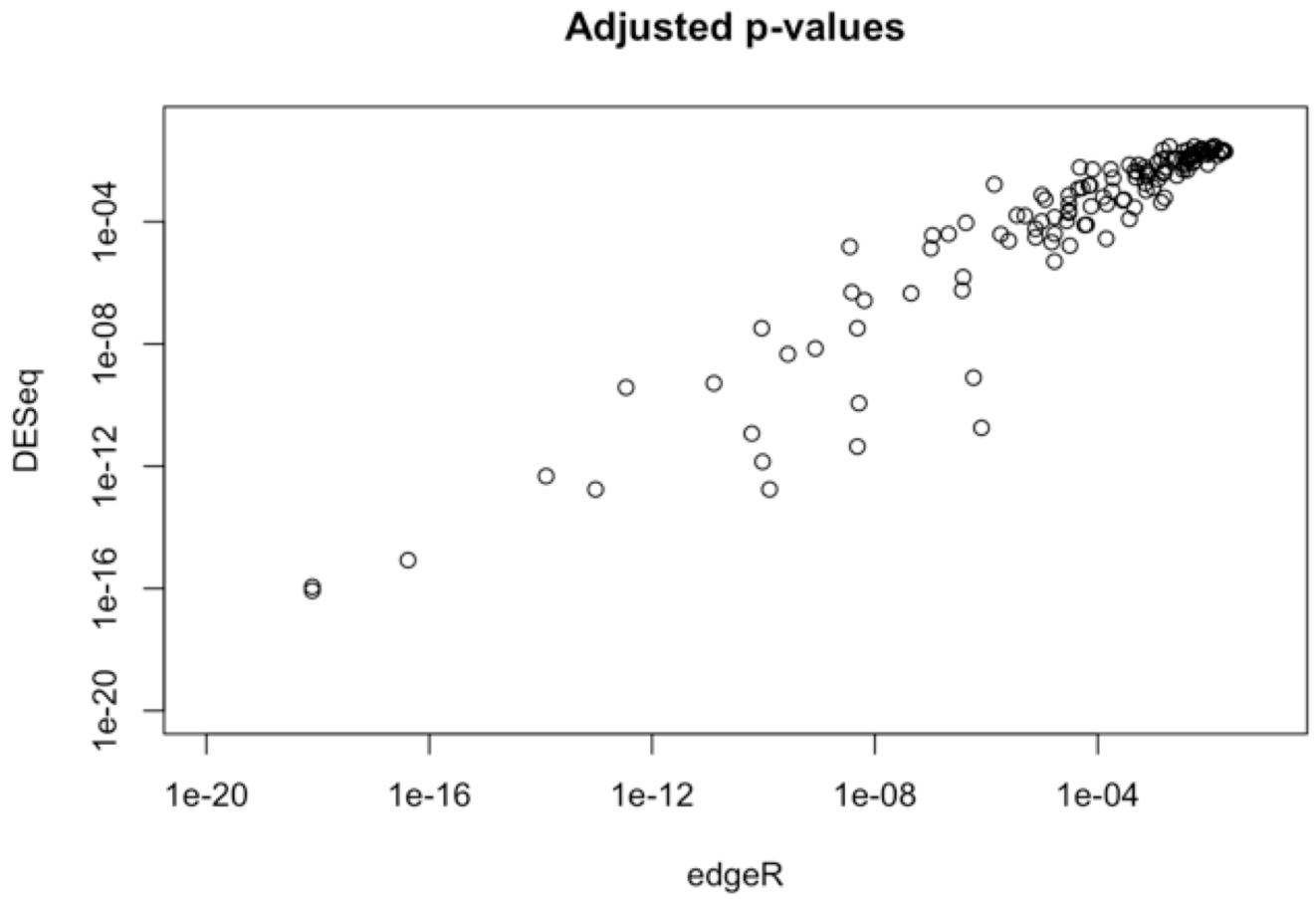
Differential expression with DESeq

```
plot(gsign[intersect(row.names(gsign),gsign.ds$id),"FDR"],gsign.ds[intersect(row.names(gsign),gsign.ds$id),"padj"
],main="Adjusted p-values",xlab="edgeR",ylab="DESeq")
```



Differential expression with DESeq

```
plot(gsign[intersect(row.names(gsign),gsign.ds$id),"FDR"],gsign.ds[intersect(row.names(gsign),gsign.ds$id),"padj"
],main="Adjusted p-values",xlab="edgeR",ylab="DESeq",log="xy",xlim=c(1e-20,1e-1),ylim=c(1e-20,1e-1))
```



Complex experimental arrangements

- Pair-wise comparisons are common in experimental biology, but most experiments have samples in more than two groups.

- These kinds of experiments are typically especially in preliminary or exploratory analyses when it is not yet clear which of the factors will be carried on for later stages of a study.

- For example, multiple concentrations of a drug compound can be tested the time of acting can be different and many muytated genes can be investigated in the same experiment.

Complex experimental arrangements

- One way to incorporate all these different aspects and groups into an analysis is to do several pair-wise comparisons.

- This is a direct extension of the arlier demonstrated two-way method, and it can be applied to all or selected condition pairs.

- For example, to study the dynamics of gene expression after a particular treatment, samples might be taken after several time intervals.

Complex experimental arrangements

- It is sensible to compare all those time points to a before treatment control state, but usually the comparison of individual time points to each other is not required.

- In this case these are as many comparisons as many comparisons as many time points are examined.

- This solution is entirely feasible and often practiced.

Complex experimental arrangements

- On the other hand, if there are several equally important conditions, the number of comparison pairs would be so high that this approach would be really impractical.

- In this case, researchers usually use analysis of variance or related approaches to find out if there are differences.

- The problem with this approach is that if there are real significant differences, we still have to apply pair-wise comparisons to find condition pairs where these differences are detectable.

Complex experimental arrangements

- An entirely different approach is originating from regression analysis and the application of general linear models (GLMs).
- Here, the goal is to figurate out which is the experimental factor that contributes significantly to the output of an experiment.
- GLM are often employed in high-throughput data analysis.
- Let us investigate how to apply them to find experimental factors affecting gene expression and differentially expressed genes in the earlier produced read count table.

Experimental factors and design matrix

- The firs step in fitting linear models to the measured data is to create an appropriate design matrix describing the relationship between the experimental groups.

- R provides the model.matrix() function to assist this process.

- This function accepts a data frame with named columns describing which samples belong to which category.

Experimental factors and design matrix

- In the example dataset individual samples are coming either from the control or the knowck-down cell cultures.
- This is the first factor of the experiment.
- Let us suppose that the measurements were performed on two consecutive days for practical reasons.
- So cells measured on day 2 were older, and they were cultured in higher densities in the flasks.

Experimental factors and design matrix

- The question now to be investigated id that if this affect gene expression? By introducing the previous commented information to the analysis.

Complex experimental arrangements

```
# Experimental factors and design matrix
repls <- rep(rep(1:2,each=2),2)
design <- model.matrix(~exp.des+repls)
design
```

```
##   (Intercept) exp.desHNRNPC knockdown repls
## 1           1                        0     1
## 2           1                        0     1
## 3           1                        0     2
## 4           1                        0     2
## 5           1                        1     1
## 6           1                        1     1
## 7           1                        1     2
## 8           1                        1     2
## attr(,"assign")
## [1] 0 1 2
## attr(,"contrasts")
## attr(,"contrasts")$exp.des
## [1] "contr.treatment"
```

Experimental factors and design matrix

- Here, the repls vector records on which day an individual sample was measured.
- The design variable holds a design matrix that is suitable for regression-type analyses.
- Since fitting a GLM to the data is a regression-type problem, this design matrix is employed in the next slides.

GLM with edgeR

- The first part of fitting a GLM with edgeR package is similar to the workflow of a two-category analysis.
- A specific data structure holding the count table is created, and common and gene-wise dispersions are estimated.
- The next step is the fitting of a negative binomial generalized log-linear model to the data

GLM with edgeR

```
genexp.glm <- DGEList(counts=counts)
genexp.glm <- estimateGLMCommonDisp(genexp.glm,design)
#genexp.glm <- estimateGLMTrendedDisp(genexp.glm,design)
genexp.glm <- estimateGLMTagwiseDisp(genexp.glm,design)
gene.exp.fit <- glmFit(genexp.glm,design)
gene.exp.fit
```

```
## An object of class "DGEGLM"
## $coefficients
##          (Intercept) exp.desHNRNPC knockdown       repls
## MED6       -7.977293               0.40778370 -0.01836644
## SNORD127  -12.056152               0.06614378 -0.01339499
## SNORD126  -11.692635               1.90575481 -1.18643792
## SNORA11B  -11.388224              -1.57500374 -0.17600434
## ZBTB42     -7.283256              -0.33648893  0.01457439
## 570 more rows ...
```

GLM with edgeR

```
## $fitted.values
##           ERR127306  ERR127307   ERR127308   ERR127309  ERR127302
## MED6     263.249239 289.031436 282.4783680 252.2169142 367.335319
## SNORD127   4.355733   4.782326   4.6959116   4.1928461   4.326423
## SNORD126   1.821629   2.000037   0.5830553   0.5205935  12.401800
## SNORA11B   7.309971   8.025898   6.6627165   5.9489504   1.304291
## ZBTB42   544.747878 598.099590 604.1224737 539.4038036 361.034648
##           ERR127303  ERR127304  ERR127305
## MED6     388.173049 387.780703 381.027521
## SNORD127   4.571847   4.588736   4.508824
## SNORD126  13.105314   3.905169   3.837161
## SNORA11B   1.378279   1.169541   1.149174
## ZBTB42   381.514961 393.897987 387.038273
## 570 more rows ...
```

GLM with edgeR

```
## $deviance
## [1]  3.832116 16.020896 25.269075  6.652989  3.103388
## 570 more elements ...
##
## $iter
## [1] 4 3 5 5 4
## 570 more elements ...
##
## $failed
## [1] FALSE FALSE FALSE FALSE FALSE
## 570 more elements ...
##
## $method
## [1] "levenberg"
```

GLM with edgeR

```
## $counts
##           ERR127306 ERR127307 ERR127308 ERR127309 ERR127302 ERR127303
## MED6            228       312       258       288       350       426
## SNORD127          7         1         6         4         5         5
## SNORD126          0         5         0         0         5        19
## SNORA11B          5        10         9         4         3         0
## ZBTB42          574       630       599       486       380       319
##           ERR127304 ERR127305
## MED6            420       330
## SNORD127          8         0
## SNORD126          9         0
## SNORA11B          1         1
## ZBTB42          388       436
## 570 more rows ...
##
```

GLM with edgeR

```
## $unshrunk.coefficients
##          (Intercept) exp.desHNRNPC knockdown       repls
## MED6       -7.977762                0.40794312 -0.01837088
## SNORD127  -12.084069                0.06801754 -0.01367226
## SNORD126  -11.741426                1.99287954 -1.22807625
## SNORA11B  -11.398411               -1.64881057 -0.18158362
## ZBTB42     -7.283493               -0.33657988  0.01458227
## 570 more rows ...
##
## $df.residual
## [1] 5 5 5 5 5
## 570 more elements ...
```

GLM with edgeR

```
## $design
##   (Intercept) exp.desHNRNPC knockdown repls
## 1           1                        0     1
## 2           1                        0     1
## 3           1                        0     2
## 4           1                        0     2
## 5           1                        1     1
## 6           1                        1     1
## 7           1                        1     2
## 8           1                        1     2
## attr(,"assign")
## [1] 0 1 2
## attr(,"contrasts")
## attr(,"contrasts")$exp.des
## [1] "contr.treatment"
```

GLM with edgeR

```
## $offset
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]   [,7]     [,8]
## [1,] 13.56923 13.66267 13.65811 13.54479 13.49446 13.54964 13.567 13.54943
## attr(,"class")
## [1] "CompressedMatrix"
## attr(,"Dims")
## [1] 5 8
## attr(,"repeat.row")
## [1] TRUE
## attr(,"repeat.col")
## [1] FALSE
## 570 more rows ...
```

GLM with edgeR

```
## $dispersion
## [1] 0.02006942 0.02657994 0.04005595 0.02379895 0.01814103
## 570 more elements ...
##
## $prior.count
## [1] 0.125
##
## $samples
##           group lib.size norm.factors
## ERR127306     1   781706            1
## ERR127307     1   858265            1
## ERR127308     1   854358            1
## ERR127309     1   762832            1
## ERR127302     1   725390            1
## ERR127303     1   766539            1
## ERR127304     1   779962            1
## ERR127305     1   766379            1
##
## $prior.df
## [1] 10
##
## $AveLogCPM
## [1] 8.714176 3.047092 3.102853 2.956143 9.239511
## 570 more elements ...
```

GLM with edgeR

- The values produced by the glmFit() function are rather complicated.

- The most important parts are that design variable holding the design matrix for the experiment and the coefficients variable that contains the values of the design matrix parameters for the individual genes.

- With those parameters, it is possible to estimate the read counts (fitted.values variable) and to check how far the observed read counts are from those fitted values (matrix).

- This excellent and certainly very complex model can be used to find out that if we compare the different sample groups (control vs knock-down samples; day 1 vs day 2 samples), which genes are differentially expressed, and which grouping has stronger effect on the gene expression.

GLM with edgeR

```
genediff.lrt <- glmLRT(gene.exp.fit,coef=2)
topTags(genediff.lrt)
```

```
## Coefficient:  exp.desHNRNPC knockdown
##                logFC    logCPM        LR        PValue           FDR
## HNRNPC     -4.520703 11.862357 625.59050 4.548326e-138 2.615288e-135
## EXOC3L4    -3.224721  8.040816 294.43787  5.365496e-66  1.542580e-63
## NFATC4     -2.504977  7.990495 169.43592  9.825831e-39  1.883284e-36
## IRF2BPL    -1.714868 11.052581 136.81960  1.320532e-31  1.898265e-29
## ZFHX2      -1.795919  7.884623 112.68449  2.529937e-26  2.909428e-24
## ARHGEF40   -1.477685  9.819598 104.24370  1.789005e-24  1.714463e-22
## LINC00641 -2.253656  6.634149  99.26007  2.214277e-23  1.818870e-21
## FOXN3       1.342795 10.308462  92.67046  6.176854e-22  4.439614e-20
## HSPA2      -1.244195 11.228285  76.32375  2.407702e-18  1.538254e-16
## LTBP2      -1.681349  8.498707  67.68559  1.917596e-16  1.102618e-14
```

GLM with edgeR

```
genediff.lrt.2 <- glmLRT(gene.exp.fit,coef=3)
topTags(genediff.lrt.2)
```

```
## Coefficient:  repls
##                logFC    logCPM        LR       PValue       FDR
## G2E3      -0.8189484  8.321130 12.901540 0.0003283118 0.1564708
## GPHN      -0.7228571  9.801758 10.403995 0.0012574300 0.1564708
## ADSSL1     1.2694980 10.348547 10.268702 0.0013530597 0.1564708
## RDH12     -1.4364102  3.435385 10.235844 0.0013773759 0.1564708
## DEGS2     -1.0659606  4.710039  9.873632 0.0016766421 0.1564708
## PGF       -0.7408800  6.172249  9.814197 0.0017316983 0.1564708
## SNORD126 -1.7116681  3.102853  9.639026 0.0019048619 0.1564708
## SGPP1     -1.3704191 11.467154  8.957307 0.0027636197 0.1902113
## SEC23A    -0.7313020 11.827112  8.569036 0.0034192818 0.1902113
## AP5M1     -0.6468231  8.922520  8.447649 0.0036551573 0.1902113
```

GLM with edgeR

- The glmLRT() function conducts likelihood ratio test for one of more coeficients in the linear model.
- The coef parameter refers to the columns in the design matrix: coef=2 refers to control versus knock-down aspect, while coef=3 to the day 1 vs. day 2 aspect.
- The FDR values of the second comparison reveal that none of the genes show significant differential expression between measurements performed on day 1 and day 2 (which is excellent since this factor is entirely artificial).
- In the next slide there is list of differentially expressed genes according to the control versus knock-down comparison and compare it to the table gained from the earlier single factor analysis.

GLM with edgeR

```
genediff.lrt$table <- cbind(genediff.lrt$table,FDR=p.adjust(genediff.lrt$table$PValue,method="fdr"))
gsign.lrt <- genediff.lrt$table[genediff.lrt$table$FDR<0.03,]
gsign.lrt <- gsign.lrt[order(gsign.lrt$FDR),]

dim(gsign.lrt)
```

```
## [1] 227   5
```

```
head(gsign.lrt)
```

```
##              logFC    logCPM       LR        PValue           FDR
## HNRNPC   -4.520703 11.862357 625.5905 4.548326e-138 2.615288e-135
## EXOC3L4  -3.224721  8.040816 294.4379  5.365496e-66  1.542580e-63
## NFATC4   -2.504977  7.990495 169.4359  9.825831e-39  1.883284e-36
## IRF2BPL  -1.714868 11.052581 136.8196  1.320532e-31  1.898265e-29
## ZFHX2    -1.795919  7.884623 112.6845  2.529937e-26  2.909428e-24
## ARHGEF40 -1.477685  9.819598 104.2437  1.789005e-24  1.714463e-22
```

GLM with edgeR

```
head(gsign.lrt, 10)
```

```
##                logFC    logCPM        LR        PValue           FDR
## HNRNPC     -4.520703 11.862357 625.59050 4.548326e-138 2.615288e-135
## EXOC3L4    -3.224721  8.040816 294.43787  5.365496e-66  1.542580e-63
## NFATC4     -2.504977  7.990495 169.43592  9.825831e-39  1.883284e-36
## IRF2BPL    -1.714868 11.052581 136.81960  1.320532e-31  1.898265e-29
## ZFHX2      -1.795919  7.884623 112.68449  2.529937e-26  2.909428e-24
## ARHGEF40   -1.477685  9.819598 104.24370  1.789005e-24  1.714463e-22
## LINC00641 -2.253656  6.634149  99.26007  2.214277e-23  1.818870e-21
## FOXN3       1.342795 10.308462  92.67046  6.176854e-22  4.439614e-20
## HSPA2      -1.244195 11.228285  76.32375  2.407702e-18  1.538254e-16
## LTBP2      -1.681349  8.498707  67.68559  1.917596e-16  1.102618e-14
```

GLM with edgeR

```
head(gsign)
```

```
##                logFC    logCPM        PValue           FDR
## HNRNPC    -4.551355 11.886747 4.632204e-127 2.663517e-124
## EXOC3L4   -3.272266  8.059754  4.854015e-55  1.395529e-52
## NFATC4    -2.548098  8.007588  2.936992e-33  5.629235e-31
## IRF2BPL   -1.755284 11.067865  5.182513e-25  7.449862e-23
## ARHGEF40  -1.523177  9.833366  8.308641e-21  7.962447e-19
## ZFHX2     -1.849398  7.899158  8.008260e-21  7.962447e-19
```

```
head(gsign, 10)
```

```
##                logFC    logCPM        PValue           FDR
## HNRNPC    -4.551355 11.886747 4.632204e-127 2.663517e-124
## EXOC3L4   -3.272266  8.059754  4.854015e-55  1.395529e-52
## NFATC4    -2.548098  8.007588  2.936992e-33  5.629235e-31
## IRF2BPL   -1.755284 11.067865  5.182513e-25  7.449862e-23
## ARHGEF40  -1.523177  9.833366  8.308641e-21  7.962447e-19
## ZFHX2     -1.849398  7.899158  8.008260e-21  7.962447e-19
## LINC00641 -2.308960  6.642652  5.078366e-19  4.171515e-17
## FOXN3      1.292207 10.293310  1.746678e-16  1.255425e-14
## HSPA2     -1.295521 11.241673  1.516631e-15  9.689586e-14
## MIS18BP1   1.380263  9.499867  5.944031e-15  3.417818e-13
```

GLM with edgeR

- In the table above, there are only minor differences in the order of the ten best genes, but there are around 50 more significant genes using this GLM-based approach.

GLM with DESeq

- Certainly, GLM-based differentially expression identification can also be done with DESeq if one prefers the dispersion calculation of this package.
- The procedure is similar to that one shown already, but it needs the model matrix in a slightly different format.

GLM with DESeq

```
# multifactor fifferential expression with glm
exp.des.cmplx<-data.frame(Type=as.character(exp.des),Repl=repls)
row.names(exp.des.cmplx)<-names(exp.des)
exp.des.cmplx
```

```
##                          Type Repl
## ERR127306             Control    1
## ERR127307             Control    1
## ERR127308             Control    2
## ERR127309             Control    2
## ERR127302 HNRNPC knockdown    1
## ERR127303 HNRNPC knockdown    1
## ERR127304 HNRNPC knockdown    2
## ERR127305 HNRNPC knockdown    2
```

GLM with DESeq

- This is a simple dataframe with the group belongings.
- Usually, this notation format is more understandable at first sight, but less used elsewhere
- The next steps are exactly the same as in edgeR.

```
count.set.glm <- newCountDataSet(counts,exp.des.cmplx)
count.set.glm <- estimateSizeFactors(count.set.glm)
count.set.glm <- estimateDispersions(count.set.glm)
```

GLM with DESeq

- However, fitting the linear model is a bit different.
- It should be explicitly specified which experimental factor has to be taken into account.
- In the first model, we have taken into account both froupings, shile in the second model only the sample types.
- Next, the significant of the differences is calculated by performing chi-squared test for each genes, and the calculated p-values are adjusted with the FDR method.
- Now you can pick up significant genes.

```
genediff.ds.glm.pvals<-nbinomGLMTest(genediff.ds.mod1,genediff.ds.mod2)
genediff.ds.glm.fdr<-p.adjust(genediff.ds.glm.pvals,method="fdr")

genediff.ds.glm <- cbind(genediff.ds.mod1[,2:3],PVal=genediff.ds.glm.pvals,FDR=genediff.ds.glm.fdr)

gsign.ds.glm <- genediff.ds.glm[genediff.ds.glm$FDR<0.03,]
gsign.ds.glm <- gsign.ds.glm[order(gsign.ds.glm$FDR),]
```

GLM with DESeq

- Comparing results to the GLM results from edge R.
- There are lot of fewer significant genes, and only a tiny part of them is common with those identified by edgeR.

```
# compare to edgeR

common.genes <- intersect(row.names(gsign.lrt),row.names(gsign.ds.glm))
dim(gsign.lrt)
```

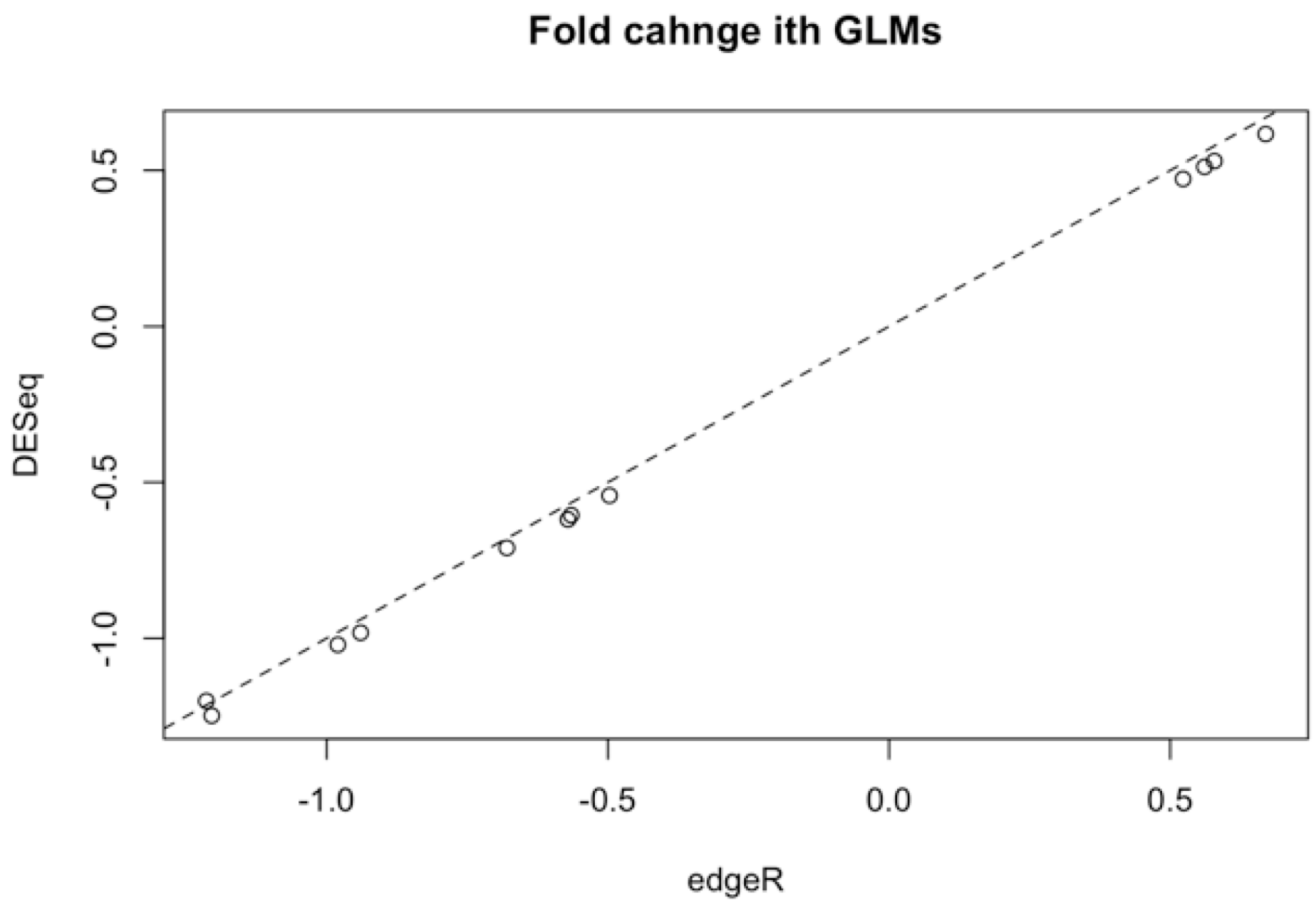
```
## [1] 227   5
```

```
dim(gsign.ds.glm)
```

```
## [1] 45  4
```

GLM with DESeq

```
plot(gsign.lrt[common.genes,"logFC"],gsign.ds.glm[common.genes,"TypeHNRNPC knockdown"],main="Fold cahnge ith GLMs
",xlab="edgeR",ylab="DESeq")
abline(0,1,lty=2)
```



GLM with DESeq

- The model that takes both types of grouping (sample types and experiment days) into account is not generating results that agree with the oder approaches.

- It is a good idea to drop the days factor from the analysis and step back to either the single factor model r the GLM with the sample types only.

Heatmap visualization

- The final task is to generate a heatmap of the differential gene expressions from different methods.
- There are many common genes between the results of the simple analyses with edgeR and DESeq.
- These results from various calculations can be visualized by employing the gplots package.
- This package offers the heatmap.2() function and an excellent color palette called redgreen() that is highly adequate for gene expression visualization.

Heatmap visualization

- Let us start with the assembling of a dataframe holding fold change values from the different analyses.

Heatmap visualization

```
# heatmap visualization
diff.exp <- data.frame(edgeR=gsign[intersect(row.names(gsign),gsign.ds$id),"logFC"],DESeq=gsign.ds[intersect(row.
names(gsign),gsign.ds$id),"log2FoldChange"])
row.names(diff.exp) <- intersect(row.names(gsign),gsign.ds$id)

diff.exp[is.infinite(diff.exp$DESeq),]<--2
```

```
library(gplots)
```

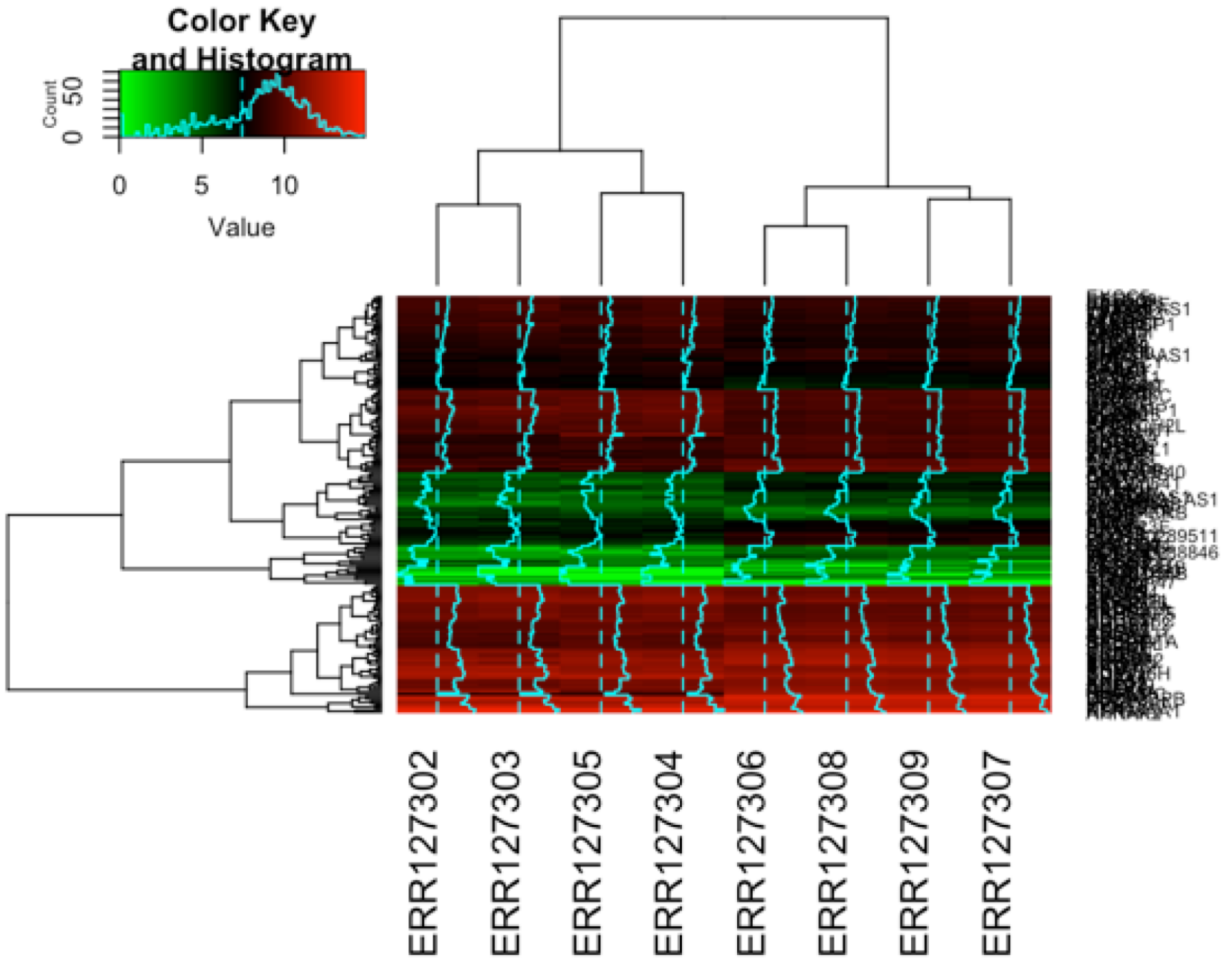
Heatmap visualization

- With this last correction, the infinite values are eliminated that are coming from some zero read counts.
- Here, the behavior of the significant genes is shown in a dataset holding measurements from the edgeR package.
- The similarity of the individual samples is well visible on the horizontal axis. Let us also compare the results from the two packages with each other.
- In conclusion the finding of the two different calculations are very similar.

GLM with DESeq

```
my.palette <-rev(redgreen(75))

# full dataset with edgeR results
heatmap.2(log2(as.matrix(counts[rownames(gsign),]+1)),col=my.palette,margins=c(10,12))
```



GLM with DESeq

```
# Behaviour of differentially expressed genes
heatmap.2(as.matrix(diff.exp),Colv=F,col=my.palette,margins = c(12, 10))
```

```
## Warning in heatmap.2(as.matrix(diff.exp), Colv = F, col = my.palette,
## margins = c(12, : Discrepancy: Colv is FALSE, while dendrogram is `both'.
## Omitting column dendogram.
```