Differential Expression of Gene Expression Data

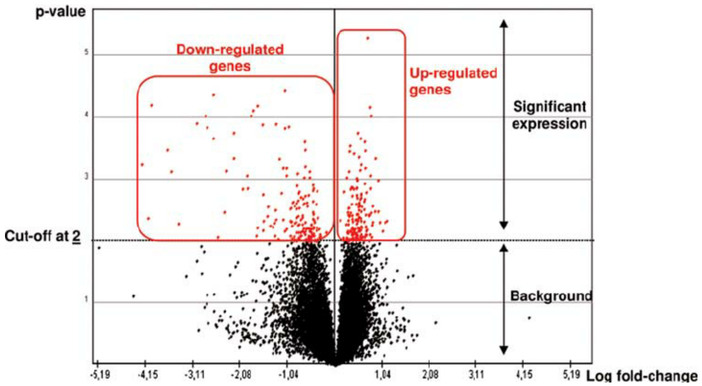
Joseph Powell

Computational and Single Cell Genomics, Institute for Molecular Bioscience

Summer Institute in Statistical Genetics

Brisbane, 13th Feb 2016

Difference in the transcriptome between conditions



Finding genes that are differentially expressed between conditions is an integral part of understanding the molecular basis of phenotypic variation.

Differential Expression

A gene is declared differentially expressed if an observed difference or change in read counts between two experimental conditions is statistically significant

- Stats for microarrays are based on numerical intensity values
- Stats for RNA-Seq instead analyze read-count distributions

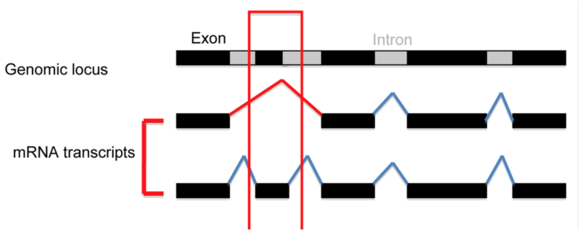
RNA-seq offers several advantages over microarrays, such as an increased dynamic range and a lower background level, and the ability to detect and quantify the expression of previously unknown transcripts and isoforms

Microarrays. One slide and that's all you are getting!

Microarrays have been used routinely for differential expression analysis for over a decade, and there are well-established methods available for this purpose (such as limma). These methods are not immediately transferable to analysis of RNA-seq data.

Ritchie *et al.* Nucleic Acids Research, 2015

RNA-Seq What level of data to choose?



With RNA-Seq data we can choose the following levels of data information

- Exon
- Transcript
- Gene

Your biological questions should probably inform this decision.

Sources of technical variation in RNA-Seq

Gene length

Most RNA-Seq protocols use an mRNA fragmentation approach before sequencing to gain sequence coverage of the whole transcript. Thus, the total number of reads for a given transcript is proportional to the expression level of the transcript multiplied by the length of the transcript.

- Thus a long transcript will have more reads mapping to it compared to a short gene of similar expression
- For this reason, most RNA-seq analysis involves some sort of length normalization.

Other obvious sources of technical variation include sequencing depth, unmatched experimental designs and relative depth of transcripts across the genome.

Length Normalization: Reads Per Kilobase Million (RPKM)

RPKM vs. FPKM

They're almost the same thing. RPKM stands for Reads Per Kilobase of transcript per Million mapped reads. FPKM stands for Fragments Per Kilobase of transcript per Million mapped reads. In RNA-Seq, the relative expression of a transcript is proportional to the number of cDNA fragments that originate from it.

These metrics attempt to normalize for sequencing depth and gene length. Here is how you do it for RPKM:
1. Count up the total reads in a sample and divide that number by 1,000,000
2. Divide the read counts by the per million scaling factor. This normalizes for sequencing depth, giving you reads per million (RPM)
3. Divide the RPM values by the length of the gene, in kilobases. This gives you RPKM

RPKM vs. FPKM

FPKM is very similar to RPKM. RPKM was made for single end RNA seq, where every read corresponded to a single fragment that was sequenced. FPKM was made for paired-end RNA seq. With paired-end RNA seq, two reads can correspond to a single fragment, or, if one read in the pair did not map, one read can correspond to a single fragment. The only difference between RPKM and FPKM is that FPKM takes into account that two reads can map to one fragment.

Relative abundance of transcripts

Relative abundance of transcripts

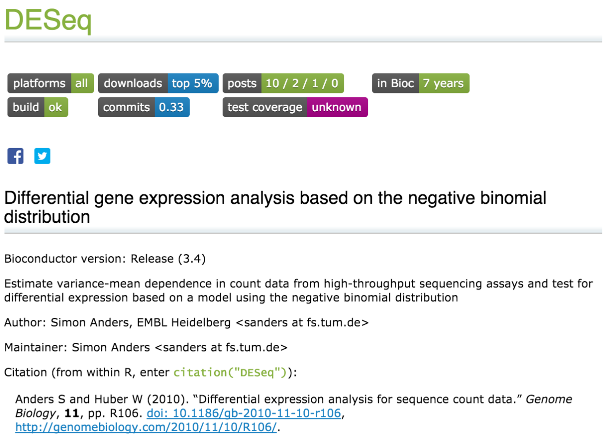
The reason is that even if the library sizes are indeed identical, RNA-seq counts inherently represent relative abundances of the genes. A few highly expressed genes may contribute a very large part of the sequenced reads in an experiment, leaving only a few reads to be distributed among the remaining genes

Considerations in cleaning the data

What other types of non-uniformities are seen between samples in an RNA-seq experiment?

- Sequencing depths or library sizes
- Differences in the conditions or covariates in the cohort.
- Library preparation methods
- Sequencing effects and batchs

DESeq



DESeq

platforms all | downloads top 5% | posts 10 / 2 / 1 / 0 | in Bioc 7 years
build ok | commits 0.33 | test coverage unknown

Differential gene expression analysis based on the negative binomial distribution

Bioconductor version: Release (3.4)

Estimate variance-mean dependence in count data from high-throughput sequencing assays and test for differential expression based on a model using the negative binomial distribution

Author: Simon Anders, EMBL Heidelberg <sanders at fs.tum.de>

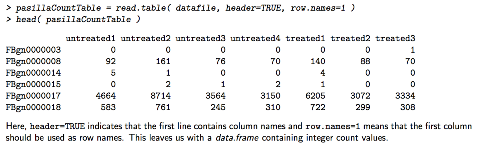
Maintainer: Simon Anders <sanders at fs.tum.de>

Citation (from within R, enter `citation("DESeq")`):

Anders S and Huber W (2010). “Differential expression analysis for sequence count data.” *Genome Biology*, **11**, pp. R106. doi: 10.1186/gb-2010-11-10-r106, http://genomebiology.com/2010/11/10/R106/.

DESeq - Input data

Count table



```
> pasillaCountTable = read.table( datafile, header=TRUE, row.names=1 )
> head( pasillaCountTable )
```

```
            untreated1 untreated2 untreated3 untreated4 treated1 treated2 treated3
FBgn0000003          0          0          0          0        0        0        1
FBgn0000008         92        161         76         70      140       88       70
FBgn0000014          5          1          0          0        4        0        0
FBgn0000015          0          2          1          2        1        0        0
FBgn0000017       4664       8714       3564       3150     6205     3072     3334
FBgn0000018        583        761        245        310      722      299      308
```

Here, `header=TRUE` indicates that the first line contains column names and `row.names=1` means that the first column should be used as row names. This leaves us with a *data.frame* containing integer count values.

Meta data

```
           condition    libType
untreated1 untreated single-end
untreated2 untreated single-end
untreated3 untreated paired-end
untreated4 untreated paired-end
treated1     treated single-end
treated2     treated paired-end
treated3     treated paired-end
```

DESeq - Normalisation

As a first processing step, we need to estimate the effective library size. This step is sometimes also called *normalisation*, even though there is no relation to normality or a normal distribution. The effective library size information is called the *size factors* vector, since the package only needs to know the relative library sizes. If the counts of non-differentially expressed genes in one sample are, on average, twice as high as in another (because the library was sequenced twice as deeply), the size factor for the first sample should be twice that of the other sample [1, 4]. The function `estimateSizeFactors` estimates the size factors from the count data. (See the man page of `estimateSizeFactorsForMatrix` for technical details on the calculation.)

```
> cds = estimateSizeFactors( cds )
> sizeFactors( cds )
```

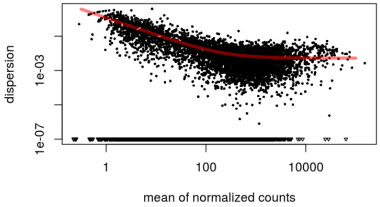
```
untreated3 untreated4   treated2   treated3
     0.873      1.011      1.022      1.115
```

If we divide each column of the count table by the size factor for this column, the count values are brought to a common scale, making them comparable. When called with `normalized=TRUE`, the `counts` accessor function performs this calculation. This is useful, e.g., for visualization.

```
> head( counts( cds, normalized=TRUE ) )
```

```
            untreated3 untreated4 treated2 treated3
FBgn0000003       0.00       0.00      0.0    0.897
FBgn0000008      87.05      69.27     86.1   62.803
FBgn0000014       0.00       0.00      0.0    0.000
FBgn0000015       1.15       1.98      0.0    0.000
FBgn0000017    4082.02    3116.93   3004.5 2991.238
FBgn0000018     280.61     306.75    292.4  276.335
```

DESeq - dispersion



The *dispersion* can be understood as the square of the coefficient of biological variation. So, if a gene's expression typically differs from replicate to replicate sample by 20%, this gene's dispersion is $0.2^2 = .04$. Note that the variance seen between counts is the sum of two components: the sample-to-sample variation just mentioned, and the uncertainty in measuring a concentration by counting reads. The latter, known as shot noise or Poisson noise, is the dominating noise source for lowly expressed genes. The former dominates for highly expressed genes. The sum of both, shot noise and dispersion, is considered in the differential expression inference.

Hence, the variance v of count values is modelled as

$$v = s\mu + \alpha s^2 \mu^2,$$

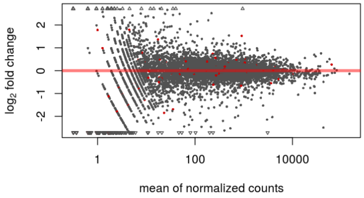
where μ is the expected normalized count value (estimated by the average normalized count value), s is the size factor for the sample under consideration, and α is the dispersion value for the gene under consideration.

To estimate the dispersions, use this command.

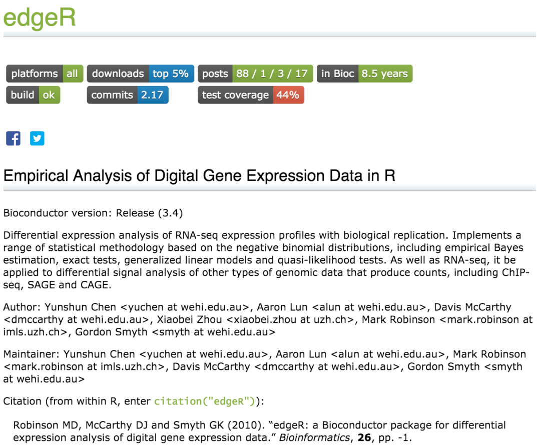
```
> cds = estimateDispersions( cds )
```

DESeq - differential expression

```
> res = nbinomTest( cds, "untreated", "treated" )

> head(res)

           id baseMean baseMeanA baseMeanB foldChange log2FoldChange  pval  padj
1 FBgn0000003    0.224      0.00     0.449        Inf            Inf 1.000 1.000
2 FBgn0000008   76.296     78.16    74.436      0.952        -0.0704 0.835 1.000
3 FBgn0000014    0.000      0.00     0.000        NaN            NaN    NA    NA
4 FBgn0000015    0.781      1.56     0.000      0.000           -Inf 0.416 1.000
5 FBgn0000017 3298.682   3599.47  2997.890      0.833        -0.2638 0.241 0.881
6 FBgn0000018  289.031    293.68   284.385      0.968        -0.0464 0.757 1.000
```



EdgeR



edgeR

platforms all | downloads top 5% | posts 88 / 1 / 3 / 17 | in Bioc 8.5 years
build ok | commits 2.17 | test coverage 44%

Empirical Analysis of Digital Gene Expression Data in R

Bioconductor version: Release (3.4)

Differential expression analysis of RNA-seq expression profiles with biological replication. Implements a range of statistical methodology based on the negative binomial distributions, including empirical Bayes estimation, exact tests, generalized linear models and quasi-likelihood tests. As well as RNA-seq, it be applied to differential signal analysis of other types of genomic data that produce counts, including ChIP-seq, SAGE and CAGE.

Author: Yunshun Chen <yuchen at wehi.edu.au>, Aaron Lun <alun at wehi.edu.au>, Davis McCarthy <dmccarthy at wehi.edu.au>, Xiaobei Zhou <xiaobei.zhou at uzh.ch>, Mark Robinson <mark.robinson at imls.uzh.ch>, Gordon Smyth <smyth at wehi.edu.au>

Maintainer: Yunshun Chen <yuchen at wehi.edu.au>, Aaron Lun <alun at wehi.edu.au>, Mark Robinson <mark.robinson at imls.uzh.ch>, Davis McCarthy <dmccarthy at wehi.edu.au>, Gordon Smyth <smyth at wehi.edu.au>

Citation (from within R, enter `citation("edgeR")`):

Robinson MD, McCarthy DJ and Smyth GK (2010). "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data." *Bioinformatics*, **26**, pp. -1.

EdgeR - Reading in data

We first need to load the required library and data required for this practical. You may use the file previously generated, or the set of read counts in Day3/Counts.RData.

Note that the genes in this file are identified by their Entrez gene ids.

```
library(edgeR)
load("Day3/Counts.RData")
Counts <- tmp$counts
colnames(Counts) <- c("16N", "16T", "18N", "18T", "19N", "19T")

dim(Counts)
head(Counts)
```

EdgeR - making an object

We will now create a DGEList object to hold our read counts. This object is a container for the counts themsleves, and also for all the associated metadata - these include, for example, sample names, gene names and normalisation factors once these are computed. The DGEList is an example of the custom task-specific structures that are frequently used in Bioconductor to make analyses easier.

```
dgList <- DGEList(counts=Counts, genes=rownames(Counts))
```

```
dgList
dgList$samples
head(dgList$counts)      #Many rows!
head(dgList$genes)       #Likewise!
```

EdgeR - Filtering data

There are approximately 26000 genes in this dataset. However, many of them will not be expressed, or will not be represented by enough reads to contribute to the analysis. Removing these genes means that we have ultimately have fewer tests to perform, thereby reducing the problems associated with multiple testing. Here, we retain only those genes that are represented at least 1cpm reads in at least two samples (cpm=counts per million).

```r
countsPerMillion <- cpm(dgList)
summary(countsPerMillion)
#'summary' is a useful function for exploring numeric data; eg. summary(1:100)
```

```r
countCheck <- countsPerMillion > 1
head(countCheck)

keep <- which(rowSums(countCheck) >= 2)
dgList <- dgList[keep,]
summary(cpm(dgList))    #compare this to the original summary
```

EdgeR - Filtering data

We are now ready to set up the model! We first need to specify our design matrix, which describes the setup of the experiment.

```
sampleType<- rep("N", ncol(dgList))      #N=normal; T=tumour
sampleType[grep("T", colnames(dgList))] <- "T"
#'grep' is a string matching function.

sampleReplicate <- paste("S", rep(1:3, each=2), sep="")
```

```
designMat <- model.matrix(~sampleReplicate + sampleType)
designMat
```

EdgeR - Estimating the dispersions

As disucssed, we need to estimate the dispersion parameter for our negative binomial model. As there are only a few samples, it is difficult to estimate the dispersion accurately for each gene, and so we need a way of 'sharing' information between genes. Possible solutions include:

- Using a common estimate across all genes.
- Fitting an estimate based on the mean-variance trend across the dataset, such that genes similar abundances have similar variance estimates (trended dispersion).
- Computing a genewise dispersion (tagwise dispersion)

In *edgeR*, we use an empirical Bayes method to 'shrink' the genewise dispersion estimates towards the common dispersion (tagwise dispersion).

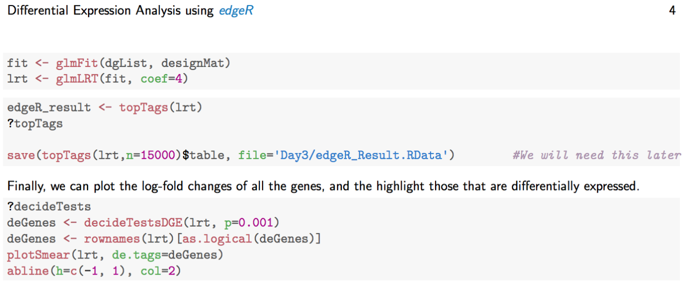
Note that either the common or trended dispersion needs to be estimated before we can estimate the tagwise dispersion.

```
dgList <- estimateGLMCommonDisp(dgList, design=designMat)
dgList <- estimateGLMTrendedDisp(dgList, design=designMat)
dgList <- estimateGLMTagwiseDisp(dgList, design=designMat)
```

We can plot the estimates and see how they differ. The biological coefficient of variation (BCV) is the square root of the dispersion parameter in the negative binomial model.

```
plotBCV(dgList)
```

EdgeR - Differential expression



```
fit <- glmFit(dgList, designMat)
lrt <- glmLRT(fit, coef=4)
```

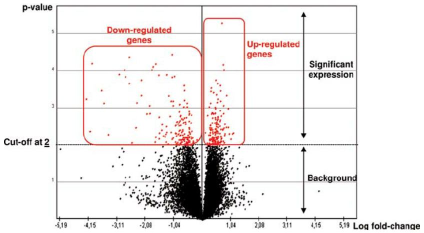
```
edgeR_result <- topTags(lrt)
?topTags

save(topTags(lrt,n=15000)$table, file='Day3/edgeR_Result.RData')        #We will need this later
```

Finally, we can plot the log-fold changes of all the genes, and the highlight those that are differentially expressed.

```
?decideTests
deGenes <- decideTestsDGE(lrt, p=0.001)
deGenes <- rownames(lrt)[as.logical(deGenes)]
plotSmear(lrt, de.tags=deGenes)
abline(h=c(-1, 1), col=2)
```

Controlling type 1 error rate

Which genes are significantly differentially expressed?



What is a *p*-value?

What is the literal meaning of $p < 0.05$?

What is a *p*-value?

Definition

The p-value is the probability of obtaining a test statistic at least as extreme as the one that was observed, assuming that the null hypothesis is true. (Wikipedia)

What is a p-value?

Definition

The p-value is the probability of obtaining a test statistic at least as extreme as the one that was observed, assuming that the null hypothesis is true. (Wikipedia)

What is the literal meaning of $p < 0.05$?

$p < 0.05$

This means that if we performed 100 random tests where we knew the null hypothesis was true, we'd see a test statistic at least this extreme five times.

We just performed 50,000 tests

- If we set our threshold at $p < 0.05$ and we perform 50,000 tests, we would expect to get 2,500 ‘significant’ results

We just performed 50,000 tests

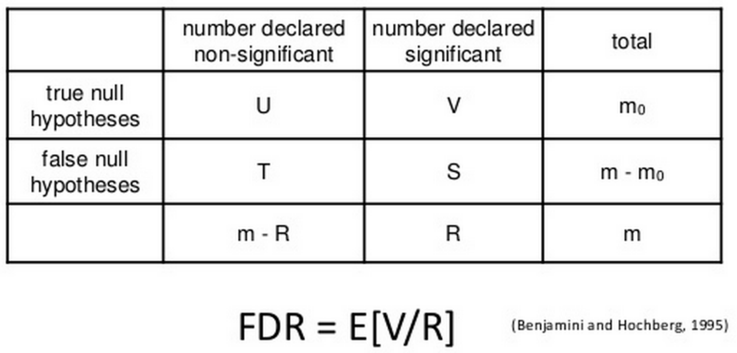
- If we set our threshold at $p < 0.05$ and we perform 50,000 tests, we would expect to get 2,500 ‘significant’ results
- To be sure that there is only a 5% chance of a false positive we must adjust our threshold

We just performed 50,000 tests

- If we set our threshold at $p < 0.05$ and we perform 50,000 tests, we would expect to get 2,500 'significant' results
- To be sure that there is only a 5% chance of a false positive we must adjust our threshold
- Bonferroni correction for multiple testing: set the threshold to:

$$p < 0.05/50000$$
$$p < 1 \times 10^{-6}$$

Controlling error rates with FDR



	number declared non-significant	number declared significant	total
true null hypotheses	U	V	m_0
false null hypotheses	T	S	$m - m_0$
	m - R	R	m

$$FDR = E[V/R]$$

(Benjamini and Hochberg, 1995)