This is a preprint of an article published in:
Methods in Molecular Biology, Volume 1418, pages 391–416 (2016).

It's DE-licious: a recipe for differential expression analyses of RNA-seq experiments using quasi-likelihood methods in edgeR

Aaron T. L. Lun[1,2] Yunshun Chen[1,2] Gordon K. Smyth[1,3]

8 April 2015

(1) Bioinformatics Division, The Walter and Eliza Hall Institute of Medical Research, 1G Royal Parade, Parkville, Victoria 3052, Australia; (2) Department of Medical Biology and (3) Department of Mathematics and Statistics, The University of Melbourne, Parkville, Victoria 3010, Australia.

Abstract

RNA sequencing (RNA-seq) is widely used to profile transcriptional activity in biological systems. Here we present an analysis pipeline for differential expression analysis of RNA-seq experiments using the Rsubread and edgeR software packages. The basic pipeline includes read alignment and counting, filtering and normalization, modelling of biological variability and hypothesis testing. For hypothesis testing, we describe particularly the quasi-likelihood features of edgeR. Some more advanced downstream analysis steps are also covered, including complex comparisons, gene ontology enrichment analyses and gene set testing. The code required to run each step is described, along with an outline of the underlying theory. The chapter includes a case study in which the pipeline is used to study the expression profiles of mammary gland cells in virgin, pregnant and lactating mice.

1 Introduction

RNA sequencing (RNA-seq) is widely used to profile transcriptional activity in biological systems, superseding microarrays as the technique of choice for profiling gene expression [1, 2, 3]. One of the most common aims of RNA-seq profiling is to identify genes or molecular pathways that are differentially expressed (DE) between two or more biological conditions. Changes in expression can then be associated with differences in biology, providing avenues for further investigation into potential mechanisms of action.

This article describes an analysis pipeline for the detection of DE genes and pathways from RNA-seq data using the Rsubread and edgeR software packages [4, 5]. We will assume throughout that RNA samples have been extracted from cells of interest under two or more treatment conditions, and that there are independent biological replicates for at least one of the treatment conditions. We assume that RNA-seq profiling has

been applied to each RNA sample and that the pipeline takes the raw sequence reads as input. The analysis strategy we describe can be applied to any RNA-seq gene expression experiment, but we give give particular attention to experiments with multiple treatment factors and with small numbers of biological replicates.

The pipeline uses the Rsubread package for mapping reads and assigning them to genes, and the edgeR package for statistical analyses. Generalized linear models are used to accommodate complex experimental designs [6]. Quasi-likelihood F-tests are used to conduct hypothesis tests [7]. edgeR provides a range of capabilities. We focus here on the quasi-likelihood features of edgeR rather than exact tests [8, 9] or likelihood ratio tests [6], because the latter have been described previously [10] and because the quasi-likelihood functions provide more robust and reliable error rate control when the number of replicates is small.

The use of the pipeline is demonstrated here with a published RNA-seq data set involving the mouse mammary gland [11]. edgeR, Rsubread and other packages used in this article are publicly available as part of the Bioconductor project [12]. See `http://www.bioconductor.org/install` for installation instructions.

2 Basic theory of the differential expression analysis

2.1 Overview

Before we start on the computational aspects of the analysis, it is useful to give an outline of the underlying theory. The next few sections describe the approach taken to process the reads into gene counts and to normalize the counts for library-specific biases. We also give a brief introduction to the statistical models used to represent complex designs, to estimate technical and biological variability and to conduct hypothesis tests.

2.2 From reads to genewise counts

Massively parallel sequencing generates millions of short read sequences from each RNA sample. We will refer to the set of reads produced from a particular sample as a library. The first step in processing the data is to align each read to the reference genome for the organism being studied. This can be done efficiently with a splice-aware aligner such as TopHat [13] or subread [4]. We use subread in this tutorial because it is especially fast and convenient, as well as being available as an R package.

The next step is to assign reads to genes. The number of reads overlapping the exons of a gene can be used as a measure of the expression level of that gene. featureCounts [14] and HTSeq [15] are popular tools for counting the reads assigned to each gene. We use featureCounts in this tutoral because it is fast and available in the Rsubread package.

This process produces a read count for each gene in each sample. An example of the read counts for a simple RNA-seq experiment is shown in Table 1. Two groups are present in the data set (wild-type and mutant), each of which contains samples from two mice, i.e., two biological replicates. After sequencing, reads for each sample are mapped

Table 1: Table of read counts for a simple RNA-seq experiment with four samples. Each column corresponds to a sample from a mouse with a wild-type or mutant genotype. Each row corresponds to a gene in the mouse genome. Each entry represents the read count for a gene in a sample.

	Wild-type		Mutant	
	Sample 1	Sample 2	Sample 3	Sample 4
Gene 1	24	31	76	59
Gene 2	0	3	7	2
Gene 3	1988	1125	3052	2450
Gene 4	5	0	0	1
...	...	...	...	...

to the mouse genome and summarized into gene-level counts. The final expression profile is represented by a matrix of read counts, with one row for each of the thousands of genes and one column per sample.

Counting reads by genes is far from the only method for RNA-seq data summarization that is available. An alternative would be to obtain a read count for each exon in each sample, in order to investigate isoform levels. Another approach might be to assemble transcripts *de novo* from the read sequences, and then count the number of reads aligned to each new transcript. We recommend the gene counting approach, however, as it is simple and effective when identification of DE genes is desired, especially when the sequencing depths are not very high.

2.3 Modelling variability in a quasi negative binomial framework

edgeR uses the negative binomial (NB) distribution to model the read counts for each gene in each sample [9, 8]. The NB disribution is ideal as it is a discrete count distribution that provides accurate modelling at low counts. It can account for variability between biological replicates, through the NB dispersion parameter [6]. The NB model can also be extended with quasi-likelihood (QL) methods to account for gene-specific variability from both biological and technical sources [7]. Write y_{gi} for the read count for gene g in sample i, and let $E(y_{gi}) = \mu_{gi}$ be the expected count for this gene in this sample given the sequencing depth and treatment conditions applied to sample i. The variance of the count is a quadratic function of the mean,

$$\text{var}(y_{gi}) = \sigma_g^2(\mu_{gi} + \mu_{gi}^2\phi)$$

where ϕ is the NB dispersion parameter and σ_g^2 is the QL dispersion parameter.

Any increase in the observed variance of y_{gi} will be modelled by an increase in the estimates for ϕ and/or σ_g^2. In this model, the NB dispersion ϕ is a global parameter whereas the QL is gene-specific, so the two dispersion parameters have different roles. The NB dispersion describes the overall biological variability across all genes. The square-root of the NB dispersion is known as the biological coefficient of variation [6]. It represents

the observed variation that is attributable to inherent variability in the biological system, in contrast to the Poisson variation from sequencing. The QL dispersion picks up any gene-specific variability above and below the overall level.

A common NB dispersion for the entire data set can be stably estimated by using information across all genes. In practice, a more flexible approach is to fit a mean-dispersion trend across genes. In this approach, ϕ is replaced by a function $\phi(A)$, where A is a measure of the overall expression level of gene g. Then $\phi(A)$ is referred to as the trended NB dispersion for that gene [6]. This approach accounts for empirical mean-variance relationships, e.g., from distributions other than the NB. Trend fitting is also stabilized by sharing information between genes with similar abundances.

Estimation of the gene-specific QL dispersion is difficult as most RNA-seq data sets have limited numbers of replicates. This means that there is often little information to stably estimate the dispersion for each gene. To overcome this, an empirical Bayes (EB) approach is used whereby information is shared between genes [16, 7, 17]. Briefly, a mean-dependent trend is fitted to the raw QL dispersion estimates. The raw estimates are then squeezed towards this trend to obtain moderated EB estimates, which can be used in place of the raw values for downstream hypothesis testing. This EB strategy reduces the uncertainty of the estimates and improves testing power.

2.4 Introducing the generalized linear model

edgeR uses the generalized linear model (GLM) framework to account for complex experimental designs [6]. The mean count for gene g in sample i is modelled as

$$\log(\mu_{gi}) = x_{i1}\beta_{g1} + x_{i2}\beta_{g2} + \ldots + x_{in}\beta_{gn} + o_i$$

where β_{gj} is the gene-specific value of coefficient j, x_{ij} is the sample-specific predictor for j, and o_i is the sample-specific offset. There are n coefficients in total, where each coefficient should describe some factor of the experimental design, e.g., different biological conditions, batch effects. The predictors can be binary or continuous, though we will focus on binary values for simplicity. Setting $x_{ij} = 1$ indicates that coefficient j contributes to the expression of sample i, i.e., the factor of the design corresponding to j affects this sample. Finally, the value of β_{gj} describes the impact of that design factor on the expression of gene g in affected samples.

For example, consider a design with three groups 1, 2 and 3. One way to choose the predictor variables x_{ij} is as indicator variables for each treatment group (Table 2). With this choice of design matrix, the coefficient β_{gj} represents the log-average expression level of gene g across the libraries in group j.

Another popular way to choose the predictor variables is to make the first coefficient β_{g1} an intercept term (Table 3). With this choice of predictor variables, the coefficients β_{g2} and β_{g3} represent log-fold expression changes for gene g between group 1 and the other two groups. Choosing between the different parametrizations of the predictor variables is a matter of convenience, as they lead to equivalent fitted models and the same downstream differential expression results.

Table 2: Group-mean design matrix. This is displayed as a table of predictor values x_{ij} for each coefficient j in each sample i, for an experimental design containing groups 1, 2 and 3. Each group has two replicate samples. This choice of predictor variables is called the group-mean parametrization. Each of the three estimated coefficients represents the log-expression level (i.e., log-count-per-million) of the gene in the corresponding treatment group.

		Coefficients		
Group	Sample	$j = 1$	$j = 2$	$j = 3$
1	$i = 1$	1	0	0
1	$i = 2$	1	0	0
2	$i = 3$	0	1	0
2	$i = 4$	0	1	0
3	$i = 5$	0	0	1
3	$i = 6$	0	0	1

Table 3: Design matrix with a reference level. This is displayed as a table of predictor values x_{ij} for each coefficient j in each sample i, for an experimental design containing groups 1, 2 and 3. Each group has two replicate samples. This choice of predictor variables treats the first treatment group as a reference. The first coefficient β_{g1} represents log-expression of the gene in the first treatment group, while β_{g2} and β_{g3} represent the log-fold-changes in expression for the second and third treatment groups, respectively, compared to the first.

		Coefficients		
Group	Sample	$j = 1$	$j = 2$	$j = 3$
1	$i = 1$	1	0	0
1	$i = 2$	1	0	0
2	$i = 3$	1	1	0
2	$i = 4$	1	1	0
3	$i = 5$	1	0	1
3	$i = 6$	1	0	1

Finally, the offset term is defined as the log-transformed library size. The library size refers to the total number of reads across all genes in each library, and depends on the amount of sequencing resources spent in characterizing each sample. Consider two samples $i = 1$ and 2, where the library size for $i = 1$ is twice that of 2. This means that the expected count μ_{g1} will be twice as large as μ_{g2} for some gene g with the same expression in both samples. The offsets ensure that the differences in library size do not contribute to spurious differences in expression, i.e.,

$$\log(\mu_{g1}) - o_1 = \log(\mu_{g2}) - o_2 \ .$$

Note that it is possible to construct more complex parametrizations involving gene- and sample-specific offsets. For simplicity, these will not be described here.

2.5 Normalizing for composition biases

Consider two samples A and B that are sequenced to the same "depth", i.e., the resulting libraries have the same number of reads. Assume that all genes are expressed at the same level in the two samples, except for one DE gene that increases in B. The DE gene will use up more sequencing resources in B, such that fewer reads will be available for all other non-DE genes. This results in "composition bias" where the read counts for the non-DE genes are suppressed in library B [18]. Spurious differences will then be observed upon comparison to the counts in library A.

Composition bias can be eliminated with trimmed mean of M-values (TMM) normalization. Briefly, most genes are assumed to be non-DE between libraries A and B. If this is true, any systematic difference in the gene counts between the two libraries must represent composition bias. The log-fold difference of the counts in B over A is estimated from the data and used to compute a scaling factor, also known as the TMM normalization factor. This factor is used to scale the size of library B to obtain an "effective" size, used to compute the GLM offset for B. More intuitively, this is equivalent to scaling the counts in library B upwards by the reciprocal of the estimated factor. This reverses the effect of suppression in B.

3 Tutorial with real data

3.1 Description of the data set

To demonstrate how the differential expression analysis works, we will use RNA-seq data from the Fu *et al.* study [11]. The sequence and count data are publicly available from the Gene Expression Omnibus (GEO) at the series accession number GSE60450. This study examines the expression profiles of basal stem-cell enriched cells (B) and committed luminal cells (L) in the mammary gland of virgin, pregnant and lactating mice. Six groups are present, with one for each combination of cell type and mouse status. Each group contains two biological replicates. This is summarized in the table below, where the basal and luminal cell types are abbreviated with `B` and `L` respectively.

```
> targets
           FileName GEOAccession CellType   Status
1  SRR1552450.fastq   GSM1480297        B   virgin
2  SRR1552451.fastq   GSM1480298        B   virgin
3  SRR1552452.fastq   GSM1480299        B pregnant
4  SRR1552453.fastq   GSM1480300        B pregnant
5  SRR1552454.fastq   GSM1480301        B  lactate
6  SRR1552455.fastq   GSM1480302        B  lactate
7  SRR1552444.fastq   GSM1480291        L   virgin
8  SRR1552445.fastq   GSM1480292        L   virgin
9  SRR1552446.fastq   GSM1480293        L pregnant
10 SRR1552447.fastq   GSM1480294        L pregnant
11 SRR1552448.fastq   GSM1480295        L  lactate
12 SRR1552449.fastq   GSM1480296        L  lactate
```

The name of the file containing the read sequences for each library is also shown. Each file is downloaded from the Sequence Read Archive and has an accession number starting with SRR, e.g., SRR1552450 for the first library in `targets`. All files have been converted to the FASTQ format - see Section 3.2 for more details.

The experimental design for this study can be parametrized with a one-way layout, whereby one coefficient is assigned to each group. The design matrix contains the predictors for each sample and and is constructed using the code below. Each row of this design matrix corresponds to a sample in `targets`. Each column represents a coefficient that corresponds to the group after which it is named. This is effectively an extension of the three-group example described in Table 2.

```
> group <- factor(paste0(targets$CellType, ".", targets$Status))
> design <- model.matrix(~ 0 + group)
> colnames(design) <- levels(group)
> design
   B.lactate B.pregnant B.virgin L.lactate L.pregnant L.virgin
1          0          0        1         0          0        0
2          0          0        1         0          0        0
3          0          1        0         0          0        0
4          0          1        0         0          0        0
5          1          0        0         0          0        0
6          1          0        0         0          0        0
7          0          0        0         0          0        1
8          0          0        0         0          0        1
9          0          0        0         0          1        0
10         0          0        0         0          1        0
11         0          0        0         1          0        0
12         0          0        0         1          0        0
attr(,"assign")
[1] 1 1 1 1 1 1
attr(,"contrasts")
attr(,"contrasts")$group
[1] "contr.treatment"
```

3.2 Read alignment and processing

Read sequences are stored in FASTQ files. Before the differential expression analysis can proceed, these reads must be aligned to the mouse genome and counted into annotated genes. This can be achieved with functions in the Rsubread package. We assume that an index of the mouse genome is already available - if not, this can be constructed from a FASTA file of the genome sequence with the `buildindex` command. In this example, we assume that the prefix for the index files is `mm10`. The reads in each FASTQ file are then aligned to the mouse genome, as shown below.

```
> library(Rsubread)
> output.files <- sub(".fastq", ".bam", targets$FileName)
> align("mm10", readfile1=targets$FileName, phredOffset=33,
+     input_format="FASTQ", output_file=output.files)
```

This produces a set of BAM files, where each file contains the read alignments for each library. The mapped reads can be counted into mouse genes by using the `featureCounts` function. The code below uses the exon intervals defined in the NCBI annotation of the mm10 genome. Recall that counts for all exons of a gene are added together to obtain the count for each gene. Repeating this for every sample generates a matrix of read counts for each gene in each sample, similar to Table 1.

```
> fc <- featureCounts(output.files, annot.inbuilt="mm10")
> colnames(fc$counts) <- 1:12
> head(fc$counts)
           1   2   3   4   5   6   7   8   9  10  11  12
497097   438 300  65 237 354 287   0   0   0   0   0   0
100503874  1   0   1   1   0   4   0   0   0   0   0   0
100038431  0   0   0   0   0   0   0   0   0   0   0   0
19888      1   1   0   0   0   0  10   3  10   2   0   0
20671    106 182  82 105  43  82  16  25  18   8   3  10
27395    309 234 337 300 290 270 560 464 489 328 307 342
```

The row names of the matrix represent the Entrez gene identifiers for each gene. In the output from `featureCounts`, the column names of `fc$counts` are the output file names from `align`. Here, we simplify them for brevity.

Notes

- Sequence data from GEO is normally obtained in the Sequence Read Archive (SRA) format. Prior to read alignment, these files should be converted into the FASTQ format using the `fastq-dump` utility from the SRA Toolkit. See `http://www.ncbi.nlm.nih.gov/books/NBK158900` for how to download and use the SRA Toolkit.
- By default, alignment is performed with `unique=TRUE`. If a read can be aligned to two or more locations, Rsubread will attempt to select the best location using a number of criteria. Only reads that have a unique best location are reported as being aligned. Keeping this default is recommended, as it avoids spurious signal from non-uniquely mapped reads derived from, e.g., repeat regions.

- The Phred offset determines the encoding for the base-calling quality string in the FASTQ file. For the Illumina 1.8 format onwards, this encoding is set at +33. However, older formats may use a +64 encoding. Users should ensure that the correct encoding is specified during alignment. If unsure, one can examine the first several quality strings in the FASTQ file. A good rule of thumb is to check whether lower-case letters are present (+64 encoding) or absent (+33).

- `featureCounts` requires gene annotation specifying the genomic start and end position of each exon of each gene. Rsubread contains built-in gene annotation for mouse and human. For other species, users will need to read in a data frame in GTF format to define the genes and exons.

3.3 Count loading and annotation

The count matrix is used to construct a `DGEList` class object. This is the main data class in the edgeR package. The `DGEList` object is used to store all the information required to fit a generalized linear model to the data, including library sizes and dispersion estimates as well as counts for each gene.

```
> library(edgeR)
> options(digits=3)
> y <- DGEList(fc$counts, group=group)
> colnames(y) <- targets$GEO
```

Human-readable gene symbols can also be added to complement the Entrez identifiers for each gene, using the annotation in the `org.Mm.eg.db` package.

```
> require(org.Mm.eg.db)
> y$genes <- select(org.Mm.eg.db,keys=rownames(y),columns="SYMBOL")
> head(y$genes)
   ENTREZID  SYMBOL
1    497097    Xkr4
2 100503874 Gm19938
3 100038431 Gm10568
4     19888     Rp1
5     20671   Sox17
6     27395  Mrpl15
```

Notes

- In the GLM framework, specifying `group` is not strictly necessary for the construction of the `DGEList`. It will not be used in the analysis when a design matrix is supplied. Setting `group` is only done here for the sake of completeness.

- Users should pick an organism package corresponding to their biological system. For example, the appropriate package for human data would be `org.Hs.eg.db`. More organism packages can be found on the Bioconductor website.

3.4 Filtering to remove low counts

Genes with very low counts across all libraries provide little evidence for differential expression. In addition, the pronounced discreteness of these counts interferes with some of the statistical approximations that are used later in the pipeline. These genes should be filtered out prior to further analysis. Here, a gene is only retained if it is expressed at a count-per-million (CPM) above 0.5 in at least two samples.

```
> keep <- rowSums(cpm(y) > 0.5) >= 2
> y <- y[keep,]
> summary(keep)
   Mode   FALSE    TRUE    NA's
logical   11375   15804       0
```

Note that the whole `DGEList` object, including annotation as well as counts, subsets by rows as if it was a matrix. This ensures the annotation and counts continue to be aligned correctly in the subsetted object.

A CPM of 0.5 is used as it corresponds to a count of 10–15 for the library sizes in this data set. If the count is any smaller, it is considered to be very low, indicating that the associated gene is not expressed in that sample. A requirement for expression in two or more libraries is used as each group contains two replicates. This ensures that a gene will be retained if it is only expressed in one group.

Notes

- Smaller CPM thresholds are usually appropriate for larger libraries. As a general rule, a good threshold can be chosen by identifying the CPM that corresponds to a count of 10, which in this case is about 0.5:

  ```
  > cpm(10, mean(y$samples$lib.size))
        [,1]
  [1,] 0.446
  ```

  Users should filter with CPMs rather than filtering on the counts directly, as the latter does not account for differences in library sizes between samples.

3.5 Normalization for composition bias

TMM normalization is performed to eliminate composition biases between libraries. This generates a set of normalization factors, where the product of these factors and the library sizes defines the effective library size. The `calcNormFactors` function returns the `DGEList` argument with only the `norm.factors` changed.

```
> y <- calcNormFactors(y)
> y$samples
                group lib.size norm.factors
GSM1480297   B.virgin 23227641        1.237
GSM1480298   B.virgin 21777891        1.214
```

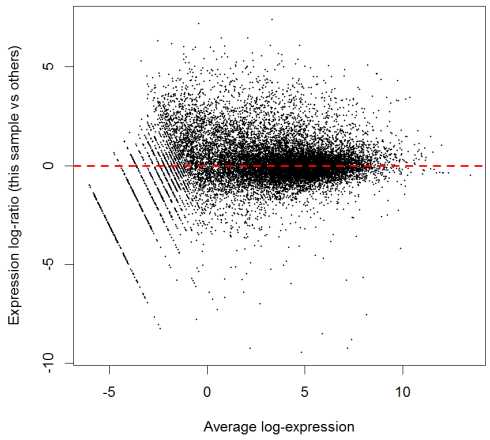

Figure 1: A MA plot of expression in sample 1, against the average expression across all other samples. Each point represents a gene, and the red line lies at a M-value of zero.

```
GSM1480299 B.pregnant 24100765       1.126
GSM1480300 B.pregnant 22665371       1.070
GSM1480301  B.lactate 21529331       1.036
GSM1480302  B.lactate 20015386       1.087
GSM1480291   L.virgin 20392113       1.368
GSM1480292   L.virgin 21708152       1.365
GSM1480293 L.pregnant 22241607       1.005
GSM1480294 L.pregnant 21988240       0.923
GSM1480295  L.lactate 24723827       0.529
GSM1480296  L.lactate 24657293       0.536
```

The normalization factors multiply to unity across all libraries. A normalization factor below unity indicates that the library size will be scaled down, as there is more suppression (i.e., composition bias) in that library relative to the other libraries. This is also equivalent to scaling the counts upwards in that sample. Conversely, a factor above unity scales up the library size and is equivalent to downscaling the counts.

The performance of the TMM normalization procedure can be examined using mean-difference (MD) plots. This visualizes the library size-adjusted log-fold change between two libraries (the difference) against the average log-expression across those libraries (the mean). In Figure 1, an MD plot is generated by comparing sample 1 against an artificial library constructed from the average of all other samples.

```
> plotMD(cpm(y, log=TRUE), column=1)
> abline(h=0, col="red", lty=2, lwd=2)
```

Ideally, the bulk of genes should be centred at a log-fold change of zero. This indicates that any composition bias between libraries has been successfully removed. This quality check should be repeated by constructing a MD plot for each sample.

Notes

- The MD plot shown above compares each library to all others. An alternative is to compare each library to a single reference library. The code below uses sample 6 as a reference, and compares sample 1 to this reference.

  ```
  > plotMD(cpm(y[,c(1,6)], log=TRUE))
  ```

  This may be easier to interpret as any problems with other libraries will not affect the MD plot for this pair. As a rule of thumb, the reference library should not be very large or small, e.g., use the sample with the median library size.

- In extreme cases, trends may be observed in the MD plot. These trended biases cannot be removed with scaling normalization methods like TMM. Rather, non-linear methods are required. These were traditionally developed for normally-distributed microarray intensities, but can be applied here on log-counts.

  ```
  > logy <- log(y$counts + 0.5)
  > normy <- normalizeBetweenArrays(logy, method="cyclicloess")
  > y$offset <- normy - logy
  ```

3.6 Exploring differences between libraries

The data can be explored by generating multi-dimensional scaling (MDS) plots. This visualizes the differences between the expression profiles of different samples in two dimensions. Figure 2 shows the MDS plot for the mammary gland data.

```
> points <- c(0,1,2,15,16,17)
> colors <- rep(c("blue", "darkgreen", "red"), 2)
> plotMDS(y, col=colors[group], pch=points[group])
> legend("topleft", legend=levels(group),
+     pch=points, col=colors, ncol=2)
```

The distance between each pair of samples in the MDS plot is calculated as the *leading fold change*, defined as the root-mean-square of the largest 500 $\log_2$-fold changes between that pair of samples. Replicate samples from the same group cluster together in the plot, while samples from different groups form separate clusters. This indicates that the differences between groups are larger than those within groups, i.e., differential expression is greater than the variance and can be detected. In Figure 2, the distance between basal samples on the left and luminal cells on the right is about 6 units, corresponding to a leading fold change of about 64-fold ($2^6 = 64$) between basal and luminal. The expression differences between virgin, pregnant and lactating are greater for luminal cells than for basal.

Notes

- The MDS plot can be simply generated with `plotMDS(y)`. The additional code is purely for aesthetics, to improve the visualization of the groups.

- Clustering in the MDS plot can be used to motivate changes to the design matrix in light of potential batch effects. For example, imagine that the first replicate of each

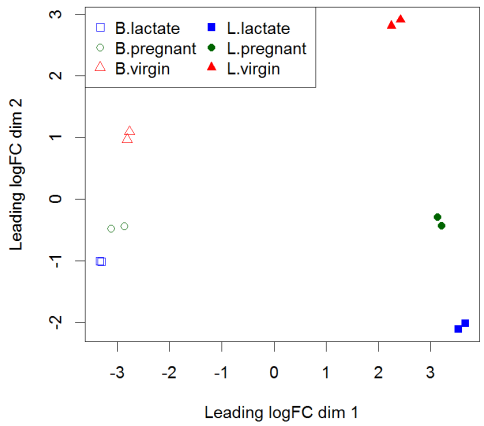

Figure 2: The MDS plot of the mammary gland RNA-seq data set. Samples are separated by the cell type in the first dimension, and by the mouse status in the second dimension.

group was prepared at a separate time from the second replicate. If the MDS plot showed separation of samples by time, it might be worthwhile adding an additional column to `design` to represent this time-based effect.

3.7 Dispersion estimation

The trended NB dispersion is estimated using the `estimateDisp` function. This returns the `DGEList` object with additional entries for the estimated NB dispersions for all gene. These estimates can be visualized with `plotBCV`, which shows the root-estimate, i.e., the biological coefficient of variation for each gene (Figure 3).

```
> y <- estimateDisp(y, design, robust=TRUE)
> plotBCV(y)
```

In general, the trend in the NB dispersions should decrease smoothly with increasing abundance. This is because the expression of high-abundance genes is expected to be more stable than that of low-abundance genes. Any substantial increase at high abundances may be indicative of batch effects or trended biases. The value of the trended NB dispersions should range between 0.005 to 0.05 for laboratory-controlled biological systems like mice or cell lines, though larger values will be observed for patient-derived data (> 0.1) or single-cell data (> 1). Note that tagwise and common estimates are also shown here but will not be used further.

For the QL dispersions, estimation can be performed using the `glmQLFit` function. This returns a `DGEGLM` object containing the estimated values of the GLM coefficients for each gene. It also contains a number of EB statistics, e.g., the fitted mean-QL dispersion trend, the squeezed QL estimates and the prior degrees of freedom (df). These can be visualized with the `plotQLDisp` function (Figure 4).

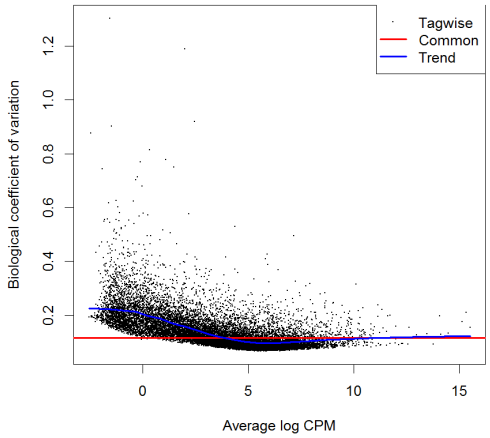

Figure 3: A plot of the biological coefficient of variation against the average abundance of each gene. Coefficients are shown corresponding to the estimates for the common, trended and tagwise NB dispersions.

```
> fit <- glmQLFit(y, design, robust=TRUE)
> head(fit$coefficients)
       B.lactate B.pregnant B.virgin L.lactate L.pregnant L.virgin
497097    -11.14     -12.02   -11.23     -19.0     -19.03    -19.0
20671     -12.77     -12.51   -12.15     -14.5     -14.31    -14.1
27395     -11.27     -11.30   -11.53     -10.6     -10.87    -10.9
18777     -10.15     -10.21   -10.77     -10.1     -10.39    -10.4
21399      -9.89      -9.74    -9.79     -10.2      -9.97    -10.0
58175     -16.16     -14.86   -15.99     -13.3     -12.29    -12.1
> plotQLDisp(fit)
```

EB squeezing of the raw dispersion estimators towards the trend reduces the uncertainty of the final estimators. The extent of this moderation is determined by the value of the prior df, as estimated from the data. Large estimates for the prior df indicate that the QL dispersions are less variable between genes, meaning that stronger EB moderation can be performed. Small values for the prior df indicate that the dispersions are highly variable, meaning that strong moderation would be inappropriate.

```
> summary(fit$df.prior)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   3.15    6.76    6.76    6.63    6.76    6.76
```

Notes

- Setting `robust=TRUE` in `glmQLFit` is strongly recommended [17]. This causes `glmQLFit` to estimate a vector of `df.prior` values, with lower values for outlier genes and

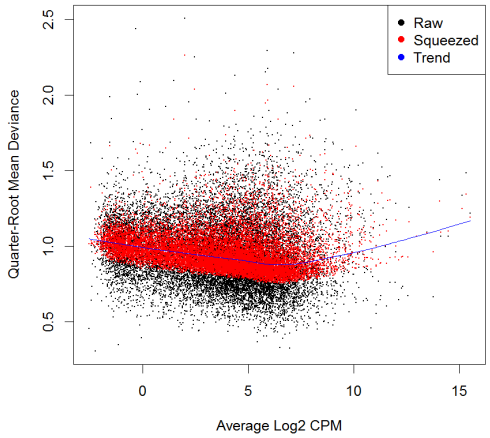

Figure 4: A plot of the quarter-root QL dispersion against the average abundance of each gene. Estimates are shown for the raw (before EB moderation), trended and squeezed dispersions.

larger values for the main body of genes. This has two advantages. First, it means that outlier genes with unusually large or small QL dispersions will be assigned lower prior df values, meaning that they will be less strongly squeezed towards the mean-dependent trend. This prevents genes with extremely low dispersions from being inappropriately called as significant. Second, and most importantly, it allows a higher prior df to be estimated for the main body of non-outlier genes. This increases the total df and increases statistical power to detect differential expression for most genes.

- Setting `robust=TRUE` in `estimateDisp` has no effect on the downstream analysis, but is nevertheless very useful as it identifies genes that are outliers from the mean-NB dispersion trend. Outliers are marked by small `prior.df` values:

```
> o <- order(y$prior.df)
> y$genes[o[1:6],]
      ENTREZID    SYMBOL
615      12835    Col6a3
1616    215866 LOC215866
2745    140703     Emid1
7070     20390     Sftpd
7789     21943   Tnfsf11
16048    11828      Aqp3
```

In mouse data sets, this set of outliers may be enriched for sex-linked genes, when replicates are from different sexes; ribosomal genes, when there are technical issues with cDNA preparation; or immunoglobulins, when the cell population is contaminated with plasma cells. If an obvious pattern can be identified among the outlier

genes, the associated set of genes can be removed beforehand in order to avoid distorting the trended NB dispersion estimates. In this case, there is no apparent enrichment of genes from any particular group, so no action is required.

- Estimating the QL dispersion requires special care for genes that have all counts exactly zero within a treatment group. `glmQLFit` handles this situation by reducing the effective residual degrees of freedom for such genes. The output vector `df.residual.zeros` contains the effective residual degrees of freedom used to estimate the raw QL dispersions.

3.8 Testing for differential expression

The final step is to actually test for significant differential expression in each gene, using the QL F-test. The contrast of interest can be specified using the `makeContrasts` function. Here, genes are detected that are DE between the basal pregnant and lactating groups. This is done by defining the null hypothesis as `B.pregnant - B.lactate = 0`.

```
> con <- makeContrasts(B.pregnant - B.lactate, levels=design)
> res <- glmQLFTest(fit, contrast=con)
```

The top set of most significant genes can be examined with `topTags`. Here, a positive log-fold change represents genes that are up in `B.pregnant` over `B.lactate`. Multiplicity correction is performed by applying the Benjamini-Hochberg method on the p-values, to control the false discovery rate (FDR). The total number of DE genes in each direction at a FDR of 5% can be examined with `decideTestsDGE`.

```
> topTags(res)
Coefficient:  -1*B.lactate 1*B.pregnant
      ENTREZID  SYMBOL logFC logCPM   F   PValue      FDR
18071    12992 Csn1s2b -6.09  10.18 421 4.71e-11 7.45e-07
22881   211577  Mrgprf -5.15   2.74 345 1.30e-10 8.06e-07
12177   226101    Myof -2.32   6.44 322 1.97e-10 8.06e-07
851     381290  Atp2b4 -2.14   6.14 320 2.04e-10 8.06e-07
9279    140474    Muc4  7.17   6.05 308 2.63e-10 8.31e-07
18829   231830 Micall2  2.25   5.18 282 4.49e-10 1.18e-06
2491     24117    Wif1  1.82   6.76 260 7.28e-10 1.58e-06
18684    12740   Cldn4  5.32   9.87 299 8.35e-10 1.58e-06
22829    21953   Tnni2 -5.75   3.86 314 9.02e-10 1.58e-06
19483   231991   Creb5 -2.57   4.86 241 1.17e-09 1.85e-06
```

The top gene Csn1s2b has a large negative log2-fold-change, showing that it is far more highly expressed in lactating than pregnant mice. This gene is known to be a major source of protein in milk.

There are in fact nearly 2500 DE genes in this comparison:

```
> is.de <- decideTestsDGE(res, p.value=0.05)
> summary(is.de)
   [,1]
-1  2094
0  11307
1   2403
```

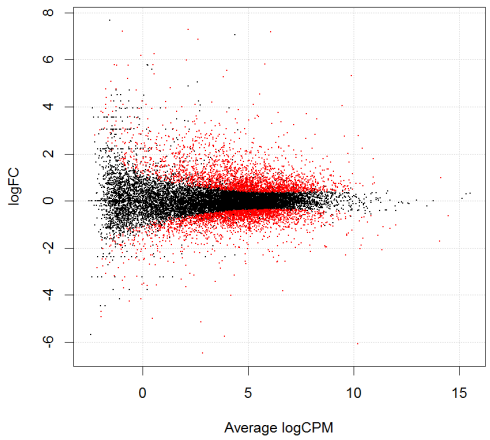

Figure 5: A smear plot showing the log-fold change and average abundance of each gene. DE genes are marked in red.

The differential expression test results can be visualized using a smear plot (Figure 5). The log-fold change for each gene is plotted against the average abundance, i.e., `logCPM` in the result table above. Significantly DE genes at a FDR of 5% are highlighted in red.

```
> plotSmear(res, de.tags=rownames(res)[is.de!=0])
```

Notes

- While the likelihood ratio test (LRT) is a more obvious choice for inferences with GLMs, the QL F-test is preferred as it reflects the uncertainty in estimating the dispersion for each gene.
- The expression supplied to `makeContrasts` is assumed to equate to zero, in order to define the null hypothesis for the contrast. The signs of the terms in the expression determine how the log-fold change is to be interpreted. For example, setting `B.lactate - B.pregnant` in `makeContrasts` would return positive log-fold changes for genes that are upregulated in the basal lactating group.

4 Advanced usage

4.1 Analysis of variance

The differential expression analysis of two-group comparison can be easily extended to comparisons between three or more groups. This is done by creating a matrix of contrasts, where which each column represents a contrast between two groups of interest. In this manner, users can perform a one-way analysis of variance (ANOVA) for each gene.

As an example, suppose we want to compare the three groups in the luminal population, i.e., virgin, pregnant and lactating. An appropriate contrast matrix can be created as shown below, to make pairwise comparisons between all three groups.

```
> con <- makeContrasts(
+     L.PvsL = L.pregnant - L.lactate,
+     L.VvsL = L.virgin - L.lactate,
+     L.VvsP = L.virgin - L.pregnant, levels=design)
```

The QL F-test is then applied to identify genes that are DE among the three groups. This combines the three pairwise comparisons into a single F-statistic and p-value. The top set of significant genes can be displayed with `topTags`.

```
> res <- glmQLFTest(fit, contrast=con)
> topTags(res)
Coefficient:  LR test of 2 contrasts
      ENTREZID  SYMBOL logFC.L.PvsL logFC.L.VvsL logCPM    F
19230    19242     Ptn        -1.54         7.26   7.96 2390
15679    13645     Egf        -5.36        -7.22   3.79 1164
21207    52150   Kcnk6        -2.42        -7.00   5.94 1021
18071    12992 Csn1s2b        -8.55       -11.36  10.18 1051
23907    15439      Hp         1.08         5.42   4.90  988
22626    14183   Fgfr2        -1.15         3.95   7.37  953
20062    11941  Atp2b2        -7.37       -10.56   6.62 1135
2901     20856    Stc2        -1.81         3.19   6.09  918
9083     13358 Slc25a1        -4.13        -4.91   7.50  888
8278     17068    Ly6d         3.42         9.24   4.65  884
        PValue      FDR
19230 3.78e-17 5.97e-13
15679 3.66e-15 2.89e-11
21207 8.39e-15 3.25e-11
18071 9.87e-15 3.25e-11
23907 1.03e-14 3.25e-11
22626 1.30e-14 3.25e-11
20062 1.61e-14 3.25e-11
2901  1.65e-14 3.25e-11
9083  2.03e-14 3.30e-11
8278  2.09e-14 3.30e-11
```

Notes

- Note that the three contrasts of pairwise comparisons are linearly dependent. Constructing the contrast matrix with any two of the contrasts would be sufficient to specify an ANOVA test. For instance, the contrast matrix shown below produces the same test results but with a different column of log-fold changes.

  ```
  > con <- makeContrasts(
  +     L.PvsL = L.pregnant - L.lactate,
  +     L.VvsP = L.virgin - L.pregnant, levels=design)
  ```

  If all three contrasts are present in the contrast matrix, then only the log-fold changes of the first two contrasts are shown in the output of `topTags`.

4.2 Complicated contrasts

The GLM framework is highly flexible in that arbitrary contrasts can be specified by the user. Suppose we are interested in the differences in the time effect from late pregnancy to early lactation between the two cell populations, i.e., whether the change in expression between pregnant and lactating groups for basal cells is the same as that for luminal cells. An appropriate contrast can be made as shown below.

```
> con <- makeContrasts(
+     (L.pregnant-L.lactate)-(B.pregnant-B.lactate),
+     levels=design)
```

The contrast is passed to `glmQLFTest` to test for genes that are DE under this comparison. The top set of DE genes can be viewed with `topTags`. A positive log-fold change represents a stronger time effect in the luminal over the basal population.

```
> res <- glmQLFTest(fit, contrast=con)
> topTags(res)
Coefficient:  1*B.lactate -1*B.pregnant -1*L.lactate 1*L.pregnant
      ENTREZID   SYMBOL logFC logCPM   F   PValue      FDR
8947     19041      Ppl  4.62   6.96 525 9.60e-12 1.52e-07
19483   231991    Creb5  5.61   4.86 439 2.93e-11 2.07e-07
7967     20512   Slc1a3 -5.03   3.69 418 3.93e-11 2.07e-07
4354    217294 BC006965  3.88   4.67 372 8.14e-11 2.92e-07
1929     14598     Ggt1 -3.17   6.38 357 1.04e-10 2.92e-07
9083     13358  Slc25a1 -3.47   7.50 354 1.11e-10 2.92e-07
12763   192166    Sardh -2.92   5.11 342 1.36e-10 3.01e-07
25207    19659     Rbp1  4.40   6.81 336 1.65e-10 3.01e-07
15727    67547  Slc39a8 -6.19   5.11 378 1.72e-10 3.01e-07
6536     14063    F2rl1  3.92   5.60 302 2.95e-10 4.34e-07
```

4.3 Gene Ontology enrichment analysis

Further analyses are required to interpret the differential expression results in a biological context. One common downstream procedure is a gene ontology (GO) enrichment analysis. Genes are grouped into GO categories, or GO terms, by some common biological property. Then, given a set of genes that are up- or down-regulated under a certain contrast of interest, a GO enrichment analysis will find which GO terms are over-represented (or under-represented) using annotations for the genes in that set.

Suppose we want to identify GO terms that are over-represented in the basal lactating group compared to the basal pregnancy group. This can be achieved by applying the `goana` function to the differential expression results of that comparison. The top set of most enriched GO terms can be viewed with the `topGO` function.

```
> con <- makeContrasts(B.lactate - B.pregnant, levels=design)
> res <- glmQLFTest(fit, contrast=con)
> go <- goana(res, species = "Mm")
> topGO(go, n=10)
                                        Term Ont    N  Up Down
GO:0044822                poly(A) RNA binding  MF 1077  84  324
```

```
GO:0003723                          RNA binding  MF 1391 118  383
GO:0042254                  ribosome biogenesis  BP  190   5  102
GO:0022613 ribonucleoprotein complex biogenesis  BP  285  18  127
GO:0022626                  cytosolic ribosome  CC   85   0   61
GO:0005730                           nucleolus  CC  667  74  213
GO:0030529           ribonucleoprotein complex  CC  592  34  195
GO:0006364                     rRNA processing  BP  136   2   72
GO:0016072               rRNA metabolic process  BP  139   4   72
GO:0003676                 nucleic acid binding  MF 2748 329  593
            P.Up   P.Down
GO:0044822 1.000 2.85e-38
GO:0003723 1.000 6.49e-36
GO:0042254 1.000 2.69e-35
GO:0022613 1.000 6.31e-33
GO:0022626 1.000 1.23e-31
GO:0005730 0.989 7.50e-29
GO:0030529 1.000 2.35e-28
GO:0006364 1.000 8.95e-25
GO:0016072 1.000 5.09e-24
GO:0003676 1.000 1.52e-23
```

The row names of the output are the universal identifiers of the GO terms, with one term per row. The `Term` column gives the names of the GO terms. These terms cover three domains - biological process (BP), cellular component (CC) and molecular function (MF), as shown in the `Ont` column. The `N` column represents the total number of genes that are annotated with each GO term. The `Up` and `Down` columns represent the number of genes with the GO term that are significantly up- and down-regulated in this differential expression comparison, respectively. The `P.Up` and `P.Down` columns contain the p-values for over-representation of the GO term across the set of up- and down-regulated genes, respectively. The output table is sorted by the minimum of `P.Up` and `P.Down` by default.

Notes

- Users can specify the domain of the enriched GO terms in `topGO`. For instance, `topGO(go,ontology="BP")` lists the top set of most enriched GO terms that are related to a biological process. This avoids other domains that are not of interest.
- The `goana` function uses the NCBI RefSeq annotation. Therefore, the Entrez Gene identifier (ID) should be supplied for each gene as the row names of `res`.
- Obviously, users should set `species` according to the organism being studied.

4.4 Rotation gene set tests

Another downstream step uses the rotation gene set test (ROAST) [19]. Given a set of genes, the aim of this procedure is to test whether the majority of the genes in the set are DE across the contrast of interest. It is useful when the specified set contains all genes involved in some pathway or process, such that systematic differential expression across the set indicates a change in the activity of the entire pathway or process.

In our case study, suppose we are interested in three GO terms related to cytokinesis. Each term will be used to define a set containing all genes that are annotated with that term. The names of these terms can be viewed as shown below.

```
> cyt.go <- c("GO:0032465", "GO:0000281", "GO:0000920")
> term <- select(GO.db, keys=cyt.go, columns="TERM")
> term
       GOID                        TERM
1 GO:0032465   regulation of cytokinesis
2 GO:0000281         mitotic cytokinesis
3 GO:0000920 cytokinetic cell separation
```

The first step is to extract the set of genes for each GO term from the GO database. We construct a list of three components, each of which is a vector of Entrez Gene IDs for all genes annotated with one of the GO terms. We then convert the Gene IDs into row indices of the `fit` object using the function `ids2indices`.

```
> Rkeys(org.Mm.egGO2ALLEGS) <- cyt.go
> ind <- ids2indices(as.list(org.Mm.egGO2ALLEGS),
+     fit$genes$ENTREZID)
```

Finally, we proceed to run ROAST on the defined gene sets for the contrast of interest. Suppose the comparison of interest is between the virgin and lactating groups in the basal population. We use `mroast` to test for multiple gene sets.

```
> con <- makeContrasts(B.virgin-B.lactate, levels=design)
> rst <- mroast(y, index=ind, design=design, nrot=9999,
+     contrast=con)
> rst
           NGenes PropDown PropUp Direction PValue     FDR
GO:0032465     42    0.167  0.476        Up 0.0004 0.00105
GO:0000920     16    0.688  0.188      Down 0.0014 0.00202
GO:0000281     26    0.385  0.423        Up 0.0058 0.00580
           PValue.Mixed FDR.Mixed
GO:0032465        2e-04     2e-04
GO:0000920        1e-04     1e-04
GO:0000281        1e-04     1e-04
```

Each row corresponds to a single gene set, i.e., GO term. The `NGenes` column gives the number of genes in each set. The `PropDown` and `PropUp` columns contain the proportions of genes in the set that are down- and up-regulated, respectively, with absolute fold changes greater than $\sqrt{2}$. The net direction of change is determined from the significance of changes in each direction, and is shown in the `Direction` column. The `PValue` provides evidence for whether the majority of genes in the set are DE in the specified direction, whereas the `PValue.Mixed` tests for differential expression in any direction. FDRs are computed from the corresponding p-values across all sets.

A barcode plot can be produced with the `barcodeplot` function to visualize the results for any particular set. In this case, visualization is performed for the gene set defined by GO:0032465 (Figure 6). Here, genes are represented by bars and are ranked from left to right by decreasing log-fold change. This forms the barcode-like pattern. The line above

the barcode shows the relative local enrichment of the vertical bars in each part of the plot. This particular plot suggests that most genes in this set are up-regulated in the virgin group compared to the lactating group.

```
> res <- glmQLFTest(fit, contrast=con)
> barcodeplot(res$table$logFC, ind[[1]], main=names(ind)[1])
```

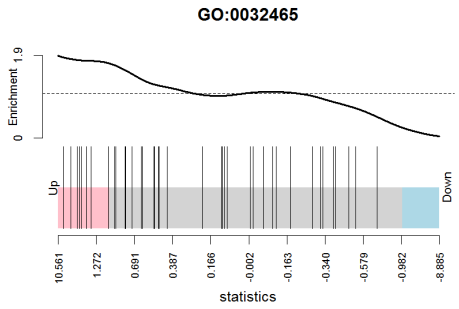

Figure 6: A barcode plot of genes in one of the GO terms, for the comparison between virgin and lactating groups in the basal population.

Notes

- Unlike `goana`, ROAST is not limited to GO terms. Any pre-defined gene set can be used, for example KEGG pathways or MSigDB gene sets. A common application is to use a set of DE genes that was defined from an analysis of an independent data set. ROAST can then determine whether similar changes are observed in the contrast of interest for the current data set.

- Even for GO-defined gene sets, `goana` and ROAST have different behaviours. In `goana`, the significance of differential expression for a GO term is determined relative to other DE genes that are not annotated with that term. In ROAST, only differential expression for the genes in the set are relevant to the significance of that set and its corresponding term. `goana` depends on a significance cutoff to choose DE genes, whereas ROAST does not require a cutoff and evaluates all genes in the set.

- The `roast` function can be used to test each gene set individually.

- ROAST estimates p-values by simulation, so the results may change slightly between runs. More precise p-values can be obtained by increasing the number of rotations, albeit at the cost of increased computational time.

- The smallest p-value that can be reported is $1/(2 \cdot \text{nrot} + 1)$ where nrot is the number of rotations. This lower bound can be decreased by increasing `nrot`.

4.5 Session information

The session information describes the versions of R and of the packages that were used in the analysis. This is useful for record-keeping purposes, and ensures that an analysis can be reproduced even when the software is updated over time.

```
> sessionInfo()
R version 3.2.0 (2015-04-16)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 7 x64 (build 7601) Service Pack 1

locale:
[1] LC_COLLATE=English_Australia.1252
[2] LC_CTYPE=English_Australia.1252
[3] LC_MONETARY=English_Australia.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_Australia.1252

attached base packages:
[1] parallel  stats4    stats     graphics  grDevices utils
[7] datasets  methods   base

other attached packages:
 [1] GO.db_3.1.2          org.Mm.eg.db_3.1.2   RSQLite_1.0.0
 [4] DBI_0.3.1            AnnotationDbi_1.30.1 GenomeInfoDb_1.4.0
 [7] IRanges_2.2.1        S4Vectors_0.6.0      Biobase_2.28.0
[10] BiocGenerics_0.14.0  edgeR_3.10.0         limma_3.24.5

loaded via a namespace (and not attached):
[1] locfit_1.5-9.1  lattice_0.20-31 grid_3.2.0      splines_3.2.0
[5] statmod_1.4.21
```

Acknowledgements

This worked was funded by the University of Melbourne (Elizabeth and Vernon Puzey Scholarship to ATLL), by the National Health and Medical Research Council (NHMRC) (Fellowship 1058892 and Program 1054618 to GKS), by the NHMRC Independent Research Institutes Infrastructure Support (IRIIS) Scheme, and by a Victorian State Government Operational Infrastructure Support (OIS) Grant.

References

[1] Mortazavi, A., Williams, B.A., McCue, K., Schaeffer, L., Wold, B.: Mapping and quantifying mammalian transcriptomes by RNA-Seq. Nature Methods **5**(7), 621–628 (2008)

[2] Wang, Z., Gerstein, M., Snyder, M.: RNA-Seq: a revolutionary tool for transcriptomics. Nat. Rev. Genet. **10**(1), 57–63 (2009)

[3] Shendure, J., Aiden, E.L.: The expanding scope of DNA sequencing. Nature Biotechnology **30**(11), 1084–1094 (2012)

[4] Liao, Y., Smyth, G.K., Shi, W.: The Subread aligner: fast, accurate and scalable read mapping by seed-and-vote. Nucleic Acids Research **41**(10), e108 (2013)

[5] Robinson, M., McCarthy, D., Smyth, G.: edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics **26**(1), 139–140 (2010)

[6] McCarthy, D.J., Chen, Y., Smyth, G.K.: Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. Nucleic Acids Research **40**(10), 4288–4297 (2012)

[7] Lund, S., Nettleton, D., McCarthy, D., Smyth, G.: Detecting differential expression in RNA-sequence data using quasi-likelihood with shrunken dispersion estimates. Statistical Applications in Genetics and Molecular Biology **11**(5), Article 8 (2012)

[8] Robinson, M.D., Smyth, G.K.: Small-sample estimation of negative binomial dispersion, with applications to SAGE data. Biostatistics **9**(2), 321–332 (2008)

[9] Robinson, M.D., Smyth, G.K.: Moderated statistical tests for assessing differences in tag abundance. Bioinformatics **23**(21), 2881–2887 (2007)

[10] Anders, S., McCarthy, D.J., Chen, Y., Okoniewski, M., Smyth, G.K., Huber, W., Robinson, M.D.: Count-based differential expression analysis of RNA sequencing data using R and Bioconductor. Nature Protocols **8**, 1765–1786 (2013)

[11] Fu, N.Y., Rios, A., Pal, B., Soetanto, R., Lun, A.T.L., Liu, K., Beck, T., Best, S., Vaillant, F., Bouillet, P., Strasser, A., Preiss, T., Smyth, G.K., Lindeman, G., , Visvader, J.: EGF-mediated induction of Mcl-1 at the switch to lactation is essential for alveolar cell survival. Nature Cell Biology **17**(4), 365–375 (2015)

[12] Huber, W., Carey, V.J., Gentleman, R., Anders, S., Carlson, M., Carvalho, B.S., Bravo, H.C., Davis, S., Gatto, L., Girke, T., Gottardo, R., Hahne, F., Hansen, K.D., Irizarry, R.A., Lawrence, M., Love, M.I., MacDonald, J., Obenchain, V., Oles, A.K., Pagès, H., Reyes, A., Shannon, P., Smyth, G.K., Tenenbaum, D., Waldron, L., Morgan, M.: Orchestrating high-throughput genomic analysis with Bioconductor. Nature Methods **12**(2), 115–121 (2015)

[13] Trapnell, C., Pachter, L., Salzberg, S.L.: TopHat: discovering splice junctions with RNA-seq. Bioinformatics **25**(9), 1105–1111 (2009)

[14] Liao, Y., Smyth, G.K., Shi, W.: featureCounts: an efficient general-purpose read summarization program. Bioinformatics **30**(7), 923–930 (2014)

[15] Anders, S., Pyl, P.T., Huber, W.: HTSeq—a Python framework to work with high-throughput sequencing data. Bioinformatics **31**(2), 166–169 (2015)

[16] Smyth, G.: Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. Statistical Applications in Genetics and Molecular Biology **3**(1), Article 3 (2004)

[17] Phipson, B., Lee, S., Majewski, I.J., Alexander, W.S., Smyth, G.K.: Empirical Bayes in the presence of exceptional cases, with application to microarray data. Tech. rep., Bioinformatics Division, Walter and Eliza Hall Institute of Medical Research, Melbourne, Australia (2013). URL `http://www.statsci.org/smyth/pubs/RobustEBayesPreprint.pdf`

[18] Robinson, M.D., Oshlack, A.: A scaling normalization method for differential expression analysis of RNA-seq data. Genome Biology **11**(3), R25 (2010)

[19] Wu, D., Lim, E., Vaillant, F., Asselin-Labat, M., Visvader, J., Smyth, G.: ROAST: rotation gene set tests for complex microarray experiments. Bioinformatics **26**(17), 2176–2182 (2010)