edgeR: Methods for differential expression in digital gene expression datasets

Mark Robinson
mrobinson@wehi.edu.au

Davis McCarthy
dmccarthy@wehi.edu.au

Gordon K. Smyth

12 October 2009

1 Introduction

This document gives a brief introduction and overview of the R Bioconductor package edgeR [Robinson and Smyth, 2007, 2008], which provides statistical routines for determining differential expression in digital gene expression data. The routines can be applied equally to SAGE, CAGE, Illumina/Solexa, 454 or ABI SOLiD experiments. In fact, the methods may be useful in other experiments where counts are observed.

The package can easily be loaded into an R session:

```
> library(edgeR)
```

R packages for the processing of raw data files for digital gene expression (DGE) datasets are still in development stages (e.g. ShortRead) at time of writing. The methods presented here require a simple DGEList object that contains three pieces of information:

1. `counts`: a table of counts where each row represents a gene/exon (or whatever genomic feature is being tracked) and each column is a different sample.
2. `group`: a vector (with length equal to the number of columns of `counts`) denoting the experimental group.
3. `lib.size` (same length as `group`): the total number of reads sequenced for each sample.

2 Reading data

We assume that the data are stored in one of two formats. Either there is a single file containing a table of counts with the first column containing the tag identifiers and the remaining columns containing the tag counts for each library sequenced, or there is an individual file for each library, each with first column for tag identifiers and second column for counts.

If the counts for all libraries are stored in a single file, then an appriopriate in-built R function (such as `read.delim` or `read.csv`) can be used to read the table of counts into R. The library sizes are the column sums from the table of counts, and thus can be easily obtained.

If the counts are stored in separate files, then, given a vector containing the filenames the `edgeR` function `readDGE` will read in the data from the individual files, collate the counts into a table and compute the library sizes and return a `DGEList` object containing the data ready.

3 Moderated negative binomial dispersions

The basic model we use for DGE data is based on the negative binomial distribution. The model is very flexible. For example, if Y is distributed as $NB(\mu, \phi)$, then the expected value of Y is μ and the variance is $\mu + \mu^2 \cdot \phi$, thus giving sufficient flexibility for many scenarios in observing count data.

The observed data can be denoted as Y_{gij} where g is the gene (tag, exon, etc.), i is the experimental group and j is the index of the replicate. We can model the counts as

$$Y_{gij} \sim NB(M_j \cdot p_{gi}, \phi_g)$$

where p_{gi} represents the proportion of the sequenced sample for group i that is tag g and M_j represents the library size.

It is of interest to find genes where, for example, p_{g1} is significantly different from p_{g2}. The parameter ϕ_g is the overdispersion (relative to the Poisson) and represents the biological, or sample-to-sample variability. The methods we developed moderate the dispersion estimates towards a common dispersion, much like how the `limma` package moderates the variances in the analysis of microarray data. It is also possible to analyse DGE data using a common dispersion for each tag using `edgeR`.

4 Case study: SAGE data

4.1 Introduction

This section provides a detailed analysis of data from a SAGE experiment to illustrate the data analysis pipeline for `edgeR`. The data come from a very early study using SAGE technology to analyse gene expression profiles in human cancer cells [Zhang et al., 1997].

4.2 Source of the data

At the time that Zhang et al. [1997] published their paper, no comprehensive study of gene expression in cancer cells had been reported. Zhang et al. [1997] designed a study to address the following issues:

1. How many genes are expressed differentially in tumour versus normal cells?

2. Are the majority of those differences cell-autonomous rather than dependent on the tumour micro-environment?

3. Are most differences cell type-specific or tumour-specific?

They used normal and neoplastic gastro-intestinal tissue as a prototype and analysed global profiles of gene expression in human cancer cells. The researchers derived transcripts from human colorectal (CR) epithelium, CR cancers or pancreatic cancers. The data that we analyse in this case study are Zhang et al. [1997]'s SAGE results for the comparison of expression patterns between normal colon epithelium and primary colon cancer.

They report that the expression profiles revealed that most transcripts were expressed at similar levels, but that 289 transcripts were expressed at significantly different levels [P-value < 0.01] and that 181 of these 289 were decreased in colon tumours as compared with normal colon tissue. Zhang et al. [1997] used Monte Carlo simulation to determine statistical significance. In this case study we will use the `edgeR` package, based around the negative binomial model, to identify genes differentially expressed in the normal and cancer samples.

4.3 Reading in the data and creating a `DGEList` object

Our first task is to load the `edgeR` package, read the data into `R` and organise the data into a `DGEList` object that the functions in the package can recognise. The library size is usually the total sum of all of the counts for a library, and that is how library size is defined in this analysis. The easiest way to construct an appropriate `DGEList` object for these data is described below.

In this case, the tag counts for the four individual libraries are stored in four separate plain text files, `GSM728.txt, GSM729.txt, GSM755.txt` and `GSM756.txt`. In each file, the tag IDs and counts for each tag are provided in a table. It is best to create a tab-delimited, plain-text 'Targets' file, which, under the headings 'files', 'group' and 'description', gives the filename, the group and a brief description for each sample.

The `targets` object is produced when the 'Targets.txt' file is read into the `R` session. This object makes a convenient argument to the function `readDGE` which reads the tables of counts into our `R` session, calculates the sizes of the count libraries and produces a `DGEList` object for use by subsequent functions.

```
> library(edgeR)
> setwd("/Users/davismcc/Documents/Honours/Data/ZhangData")
> targets <- read.delim(file = "Targets.txt", stringsAsFactors = FALSE)
> targets
```

```
       files group                 description
1 GSM728.txt    NC                Normal colon
2 GSM729.txt    NC                Normal colon
3 GSM755.txt    Tu Primary colonrectal tumour
4 GSM756.txt    Tu Primary colonrectal tumour
```

```
> d <- readDGE(targets, skip = 5, comment.char = "#")
> d

An object of class "DGEList"
$samples
            files group                 description lib.size
GSM728 GSM728.txt    NC                Normal colon    50179
GSM729 GSM729.txt    NC                Normal colon    49593
GSM755 GSM755.txt    Tu Primary colonrectal tumour    57686
GSM756 GSM756.txt    Tu Primary colonrectal tumour    49064

$counts
           GSM728 GSM729 GSM755 GSM756
CCCATCGTCC   1288   1380   1236      0
CCTCCAGCTA    719    458    148    142
CTAAGACTTC    559    558    248    199
GCCCAGGTCA    520    448     22     62
CACCTAATTG    469    472    763    421
57443 more rows ...
```

This `DGEList` is now ready to be passed to the functions that do the calculations to determine differential expression levels for the genes. Note that when we 'see' the `DGEList` object `d`, the counts for just the first five genes in the table are shown, as well as the library sizes and groups for the samples.

4.4 Analysis using common dispersion

4.4.1 Estimating the common dispersion

The first major step in the analysis of DGE data using the NB model is to estimate the dispersion parameter for each tag. The most straight-forward analysis of DGE data uses the common dispersion estimate as the dispersion for all tags. For many applications this will be adequate and it may not be necessary to estimate tagwise dispersions, i.e. estimate the dispersion parameter separately for each tag. Using the common dispersion allows the user to obtain DE results very quickly and in few steps, and so makes a good place to start with any analysis of DGE data.

Estimating the common dispersion is done using the function `estimateCommonDisp`. In order to do this, the function first needs to generate the 'pseudocounts' under the alternative hypothesis (that there really is a difference in expression level between the groups). The conditional maximum likelihood method assumes that the library sizes are equal, which is certainly not true in general for DGE data.

The pseudocounts are calculated using a quantile-to-quantile method for the negative binomial distribution so that the library sizes for the pseudocounts are equal to the geometric mean of the original library sizes. These pseudocounts are then used as the count data for the common conditional negative binomial likelihood function, which is maximised over the dispersion parameter to obtain our estimate of the common dispersion.

```
> d <- estimateCommonDisp(d)
> names(d)
```

```
[1] "samples"          "common.dispersion" "counts"
[4] "pseudo.alt"       "conc"              "common.lib.size"
```

The output of `estimateCommonDisp` is a DGEList object with several new elements. The element `common.dispersion`, as the name suggests, provides the estimate of the common dispersion, and `pseudo.alt` gives the pseudocounts calculated under the alternative hypothesis. The element `conc` gives the estimates of the overall concentration of each tag across all of the original samples (`conc$conc.common`) and the estimate of the concentration of each tag within each group (`conc$conc.group`). The element `common.lib.size` gives the library size to which the original libraries have been adjusted in the pseudocounts.

We see in the output below that the total number of counts in each library of the pseudocounts agrees well with the common library size, as desired.

```
> d$samples$lib.size
```

```
[1] 50179 49593 57686 49064
```

```
> d$common.lib.size
```

```
[1] 51516
```

```
> colSums(d$pseudo.alt)
```

```
GSM728 GSM729 GSM755 GSM756
 51512  51513  51674  51483
```

Under the negative binomial model, the square root of the common dispersion gives the coefficient of variation of biological variation. Here, as seen in the code below, the coefficient of variation of biological variation is found to be 0.45. We also note that a common dispersion estimate of 0.2 means that there is a lot more variability in the data that can be accounted for by the Poisson model—if a tag has just 200 counts, then the estimate of the tag's variance under the NB model is over 40 times greater than it would be under the Poisson model.

```
> d$common.dispersion
```

```
[1] 0.1988
```

```
> sqrt(d$common.dispersion)
```

```
[1] 0.4458
```

4.4.2 Testing

Once we have an estimate of the common dispersion, we can proceed with testing procedures for determining differential expression. The `edgeR` package uses an exact test for the negative binomial distribution, which has strong parallels with Fisher's exact test, to compute exact p-values that can be used to assess differential expression. The function `exactTest` allows the user to conduct the NB exact test for pairwise comparisons of groups. Here there are only two groups, so the pair need not be specified—the function by default compares the two groups present.

```
> de.com <- exactTest(d)
```

```
Comparison of groups:  Tu - NC
```

```
> names(de.com)
```

```
[1] "table"      "comparison"
```

```
> names(de.com$table)
```

```
[1] "logConc" "logFC"   "p.value"
```

The object produced by `exactTest` contains two elements: `table` and `comparison`. The element `de.com$comparison` contains a vector giving the names of the two groups compared. The table`de.com$table` contains the elements `logConc`, which gives the overall concentration for a tag across the two groups being compared, `logFC`, which gives the log-fold change difference for the counts between the groups and `p.value` gives the exact p-values computed.

The results of the NB exact test can be accessed conveniently using the `topTags` function applied to the object produced by `exactTest`. The user can specify the number, `n`, of tags for which they would like to see the differential expression information, ranked by p-value (default) or fold change. As the same test is conducted for many thousands of tags, adjusting the p-values for multiple testing is recommended. The desired adjustment method can be supplied by the user, with the default method being Benjamini and Hochberg's approach for controlling the false discovery rate (FDR) [Benjamini and Hochberg, 1995]. The table below shows the top 10 DE genes ranked by p-value.

The output below shows that the `edgeR` package identifies a good deal of differential expression between the normal colon cell group and the primary CR cancer cell group. The top DE genes are given very small p-values, even after adjusting for multiple testing. Furthermore, all of the top genes have a large fold change, indicating that these genes are more likely to be biologically meaningful. A Gene Ontology analysis could be carried out using the list of top genes and p-values provided by `topTags` in order to obtain more systematic and functional information about the differentially expressed genes.

```
> options(digits = 4)
> topTags(de.com)
```

```
Comparison of groups:  Tu - NC
           logConc   logFC    PValue       FDR
AGCTGTTCCC  -28.24  43.552 1.017e-19 5.841e-15
CTTGGGTTTT  -29.88  40.271 2.967e-10 8.522e-06
TACAAAATCG  -30.26  39.510 2.712e-08 4.926e-04
CCCAACGCGC  -12.80  -5.849 3.430e-08 4.926e-04
GCCACCCCCT  -30.33  39.380 6.341e-08 7.285e-04
CCAGTCCGCC  -30.43  39.177 1.844e-07 1.513e-03
GTCATCACCA  -30.42 -39.189 1.844e-07 1.513e-03
TCACCGGTCA  -11.13  -4.229 4.428e-07 3.180e-03
TAAATTGCAA  -11.39  -4.242 6.090e-07 3.887e-03
CGCGTCACTA  -12.30   4.611 7.300e-07 4.194e-03
```

The table below shows the raw counts for the genes that `edgeR` has identified as the most differentially expressed. For these genes there seems to be very large differences between the groups, suggesting that the DE genes identified are truly differentially expressed, and not false positives.

```
> detags.com <- rownames(topTags(de.com)$table)
> d$counts[detags.com, ]
```

```
           GSM728 GSM729 GSM755 GSM756
AGCTGTTCCC      0      0    119   1011
CTTGGGTTTT      0      0     21     97
TACAAAATCG      0      0     14     56
CCCAACGCGC    106      1      2      0
GCCACCCCCT      0      0      5     58
CCAGTCCGCC      0      0      6     49
GTCATCACCA     35     20      0      0
TCACCGGTCA    118     75      6      5
TAAATTGCAA    103     59      3      6
CGCGTCACTA      1      3     88     21
```

If we order the genes by fold change instead of p-value, as in the table below, we see that the genes with the largest fold changes have very small concentrations. This ranking is dominated by genes that have zero total counts in one group and is less informative than ranking by p-value.

```
> topTags(de.com, sort.by = "logFC")
```

```
Comparison of groups:  Tu - NC
           logConc  logFC    PValue       FDR
AGCTGTTCCC  -28.24  43.55 1.017e-19 5.841e-15
CTTGGGTTTT  -29.88  40.27 2.967e-10 8.522e-06
```

```
TACAAAATCG  -30.26  39.51 2.712e-08 4.926e-04
GCCACCCCCT  -30.33  39.38 6.341e-08 7.285e-04
GTCATCACCA  -30.42 -39.19 1.844e-07 1.513e-03
CCAGTCCGCC  -30.43  39.18 1.844e-07 1.513e-03
GTGCGCTGAG  -30.67  38.69 2.362e-06 9.048e-03
CTTGACATAC  -30.69 -38.66 2.837e-06 9.588e-03
GTGTGTTTGT  -30.72  38.59 4.135e-06 1.320e-02
GGGGGGGGGG  -30.74  38.56 5.019e-06 1.518e-02
```

Zhang et al. [1997] identified 289 genes as significantly differentially expressed with p-values less than 0.01. We can look at the genes that are given an exact p-value less than 0.01 by `edgeR` before adjusting for multiple testing, and less than 0.05 after adjustment.

We see in the output below that 264 genes are significantly differentially expressed according to `edgeR` when using the common dispersion estimate. Of those genes, 100 are up-regulated in the cancer cells compared with the normal cells and 164 are down-regulated in the cancer cells compared with normal cells. These proportions of up- and down-regulated tags are very similar to those found by Zhang et al. [1997].

```
> sum(de.com$table$p.value < 0.01)

[1] 264

> top264 <- topTags(de.com, n = 264)
> sum(top264$table$logFC > 0)

[1] 100

> sum(top264$table$logFC < 0)

[1] 164
```

Furthermore, we see below that 33 tags (0.06% of the total number) have a p-value of less than 0.05 after adjusting for multiple testing using the Benjamini and Hochberg [1995] method for controlling the FDR, which is strong evidence for differential expression.

```
> sum(p.adjust(de.com$table$p.value, method = "BH") < 0.05)

[1] 33

> mean(p.adjust(de.com$table$p.value, method = "BH") < 0.05) *
+     100

[1] 0.05744
```

4.4.3 Visualising DGE results

The function `plotSmear` can be used to generate a plot of the log-fold change against the log-concentration for each tag (analogous to an MA-plot in the microarray context). We can easily identify the top DE tags and highlight them on the plot. The code for producing the default fold-change plot is shown below, and the result of this code is shown in Figure 5.

```
> detags264 <- rownames(topTags(de.com, n = 264)$table)

> plotSmear(d, de.tags = detags264, main = "FC plot using common dispersion")
> abline(h = c(-2, 2), col = "dodgerblue")
```

Figure 5 shows the default fold change-plot for these data—the 'smear plot'. Plotting DGE data poses some challenges, as when the total counts in one group are zero, the log-fold change is technically infinite, and the log-concentration is negative infinity. With the algorithm used by topTags, we see very high log-fold changes and very small values for log-concentration for such tags, but plotting these values directly causes problems with the scale of the graph. To get around this problem, `edgeR` produces a 'smear' of points at the left-most edge of the plot for tags which have zero counts in one of the groups. Although this is still slightly artificial, it has the advantage that the expression level of all tags can be visualised and interpreted simultaneously.

The 'lines' of points we see at smaller log-concentration values arise from the discrete nature of the count data. When the sum all of the counts in one of the groups is one, we see the lines of points furthest away from the main body of points, and other lines of points correspond to when the total sum of counts in one of the groups is 2, 3, 4 and so on.

In Figure 5, the 264 tags identified as differentially expressed (i.e. those identified as significant (p-value less than 0.01) by `edgeR` using the common dispersion) are outlined in red.

4.5 Analysing the data using moderated tagwise dispersions

4.5.1 Estimating the tagwise dispersion

An extension to simply using the common dispersion for each tag is to estimate the dispersion separately for each tag, while 'squeezing' these estimates towards the common dispersion estimate in order to improve inference by sharing information between tags. This type of analysis can also be carried out in few steps using the `edgeR` package.

As noted earlier, the dispersion parameter is the overdispersion relative to the Poisson, and represents the biological, or sample-to-sample variability. The methods we developed moderate the dispersion estimates towards a common dispersion, much like how the `limma` package moderates the variances in the analysis of microarray data.

To run the moderated analysis, we need to determine how much moderation is necessary. For this, we can use an empirical Bayes rule that involves calculating a weight parameter prior.n. However, for many applications (especially if the estimated smoothing parameter is large), using the common dispersion for all tags will give excellent results.

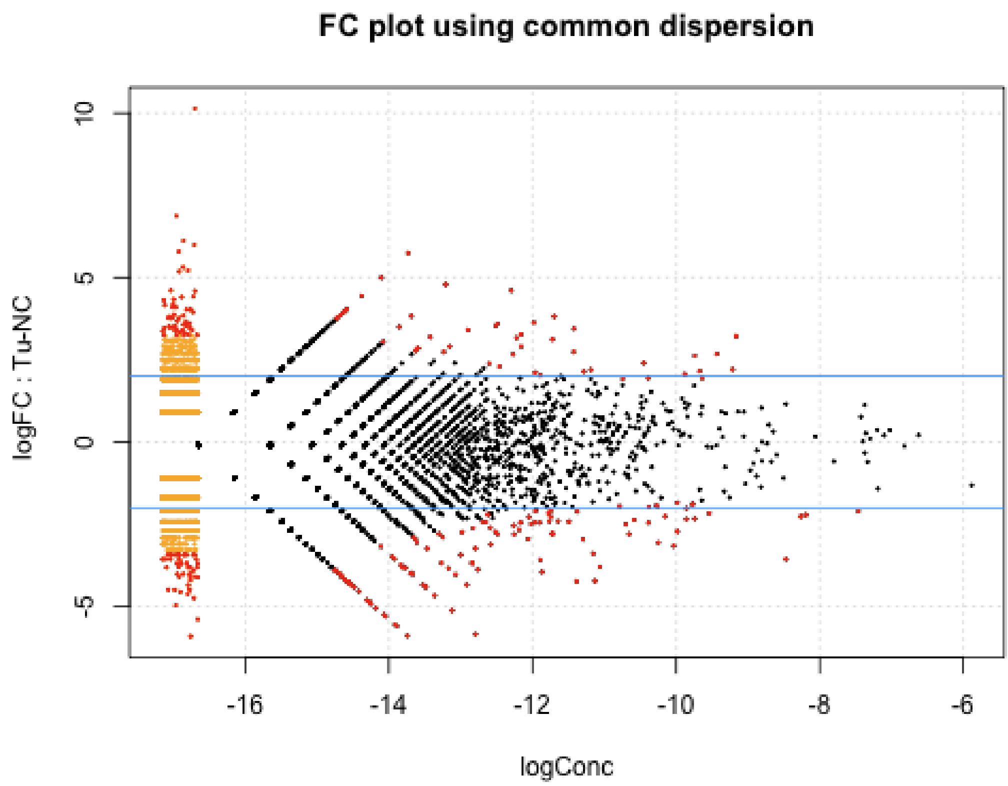


Figure 1: Plot of the log-fold change against the log-concentration for each tag. The 264 most differentially expressed tags as identified by `edgeR` using the common dispersion are outlined in red.

In order to determine how much moderation is necessary, we require an estimate of the dispersion parameter, so we use the common dispersion estimate from `estimateCommonDisp`. The smoothing parameter, `prior.n` can then be calculated. As we see below, for this dataset the estimate of the smoothing parameter is very large, so if we were to use this large value for the weight parameter we would moderate the tagwise dispersion estimates so strongly that it would be equivalent to using the common dispersion estimate.

```
> prior.n <- estimateSmoothing(d)
> prior.n
```

```
[1] 29481
```

Alternatively, the value for the weight parameter `prior.n` can be selected *a priori*, instead of being estimated. In an experiment such as that we consider here, in which we have just four samples, two in each group, and thus two degrees of freedom for estimating the dispersion parameter, setting the `prior.n` to be relatively large should be appropriate, so that individual tagwise dispersion estimates are 'squeezed' quite strongly towards the common dispersion. Here, we choose `prior.n` to be 100. This means that the common likelihood receives the weight of 100 individual tags, so there will be a reasonable degree of 'squeezing', but there is still scope to estimate an individual dispersion for each gene.

The function `estimateTagwiseDisp` produces a DGEList object that contains all of the elements present in the object produced by `estimateCommonDisp`, as well as the value for `prior.n` used (`d$prior.n`) and the tagwise dispersion estimates (`d$tagwise.dispersion`), as we see below.

```
> d <- estimateTagwiseDisp(d, prior.n = 100)

Using grid search to estimate tagwise dispersion.

> names(d)

[1] "samples"            "common.dispersion"  "prior.n"
[4] "tagwise.dispersion" "counts"             "pseudo.alt"
[7] "conc"               "common.lib.size"

> d$prior.n

[1] 100

> head(d$tagwise.dispersion)

[1] 0.99002 0.12528 0.08252 0.22934 0.11896 0.15795
```

It is interesting to consider the distribution of the tagwise dispersion estimates. As we can see from the output below, the tagwise dispersion estimates range from a minimum of 0.08 to a maximum of 0.99, but the tags in the middle two quartiles of the tagwise dispersion estimates have dispersion estimates just slightly larger than the common dispersion estimate.

```
> summary(d$tagwise.dispersion)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0825  0.2000  0.2000  0.2000  0.2000  0.9900

> d$common.dispersion

[1] 0.1988
```

4.5.2 Testing

The testing procedures when using tagwise dispersion estimates are carried out exactly as for the common dispersion, as described above, but we add the argument `common.disp=FALSE` to the call to `exactTest`.

```
> de.tgw <- exactTest(d, common.disp = FALSE)

Comparison of groups:  Tu - NC
```

The output below shows that when using tagwise dispersions, the `edgeR` package still identifies a lot of differential expression between the normal colon cell group and the primary CR cancer cell group. The top DE tags are given very small p-values, even after adjusting for multiple testing. As with the analysis using the common dispersion, all of the top tags have a large fold change, indicating that these changes in expression are likely to be biologically meaningful. We note that the ranking is different, however, and not all of the top ten tags according to using the common dispersion are found to be among the top ten tags using tagwise dispersions.

```
> topTags(de.tgw)

Comparison of groups:  Tu - NC
           logConc   logFC    PValue       FDR
AGCTGTTCCC -28.240  43.553 1.541e-10 8.855e-06
TCACCGGTCA -11.130  -4.229 3.892e-09 1.118e-04
GTCATCACCA -30.421 -39.189 7.815e-08 1.497e-03
TAAATTGCAA -11.390  -4.244 1.846e-07 2.355e-03
CTTGGGTTTT -29.879  40.274 2.049e-07 2.355e-03
TAATTTTTGC -13.735   5.741 4.550e-07 4.356e-03
CTTGACATAC -30.687 -38.658 5.735e-07 4.378e-03
ATTTCAAGAT -13.747  -5.893 6.097e-07 4.378e-03
TGCTCCTACC  -9.985  -2.722 7.139e-07 4.407e-03
GTGCGCTGAG -30.672  38.688 7.671e-07 4.407e-03
```

The table below shows the raw counts for the tags that `edgeR` has identified as the most differentially expressed using tagwise dispersions. For these genes there seems to be very large differences between the groups, suggesting that the DE genes identified are truly differentially expressed, and not false positives.

We note that in general, when using tagwise dispersions, the counts are more consistent within groups, as using tagwise dispersions instead of the common dispersion penalises tags which are highly variable within groups. The smaller the value selected for `prior.n`, the more highly variable tags will be penalised, as there is less 'squeezing' of the tagwise dispersions towards the common value.

```
> detags.tgw <- rownames(topTags(de.tgw)$table)
> d$counts[detags.tgw, ]
```

```
           GSM728 GSM729 GSM755 GSM756
AGCTGTTCCC      0      0    119   1011
TCACCGGTCA    118     75      6      5
GTCATCACCA     35     20      0      0
TAAATTGCAA    103     59      3      6
CTTGGGTTTT      0      0     21     97
TAATTTTTGC      0      1     37     21
CTTGACATAC     18     20      0      0
ATTTCAAGAT     35     21      0      1
TGCTCCTACC    140    113     22     19
GTGCGCTGAG      0      0     18     23
```

Of course, we can sort the top table differently, as we did earlier.

We see in the output below that 254 genes are significantly differentially expressed according to `edgeR` when using the tagwise dispersion estimates (ten fewer than when using the common dispersion). Of those tags, 87 are up-regulated in the cancer cells compared with the normal cells and 167 are down-regulated in the cancer cells compared with normal cells. These proportions of up- and down-regulated tags are similar to those found using the common dispersion, but there is a slightly higher proportion of down-regulated tags in those identified as DE using tagwise dispersions.

```
> sum(de.tgw$table$p.value < 0.01)

[1] 254

> top254 <- topTags(de.tgw, n = 254)
> sum(top254$table$logFC > 0)

[1] 87

> sum(top254$table$logFC < 0)

[1] 167
```

Furthermore, we see below that 28 tags (0.05% of the total number) have a p-value of less than 0.05 after adjusting for multiple testing using the Benjamini and Hochberg [1995] method for controlling the FDR, which is strong evidence for differential expression.

```
> sum(p.adjust(de.tgw$table$p.value, method = "BH") < 0.05)

[1] 28

> mean(p.adjust(de.tgw$table$p.value, method = "BH") < 0.05) *
+     100

[1] 0.04874
```

4.5.3 Visualising DGE results

Shown below is the code for producing the default fold-change plot using `plotSmear` with the DE tags as determined using tagwise dispersions highlighted, and the result of this code is shown in Figure 6.

```
> detags254 <- rownames(topTags(de.tgw, n = 254)$table)

> plotSmear(d, de.tags = detags254, main = "FC plot using tagwise dispersions")
> abline(h = c(-2, 2), col = "dodgerblue")
```

In Figure 6, the 254 tags identified as differentially expressed (i.e. those identified as significant (p-value less than 0.01) by `edgeR` using the tagwise dispersions) are highlighted in red. We see that the pattern of differential expression using tagwise dispersions that we see in Figure 6 is very similar to that obtained using the common dispersion that we saw in Figure 5.

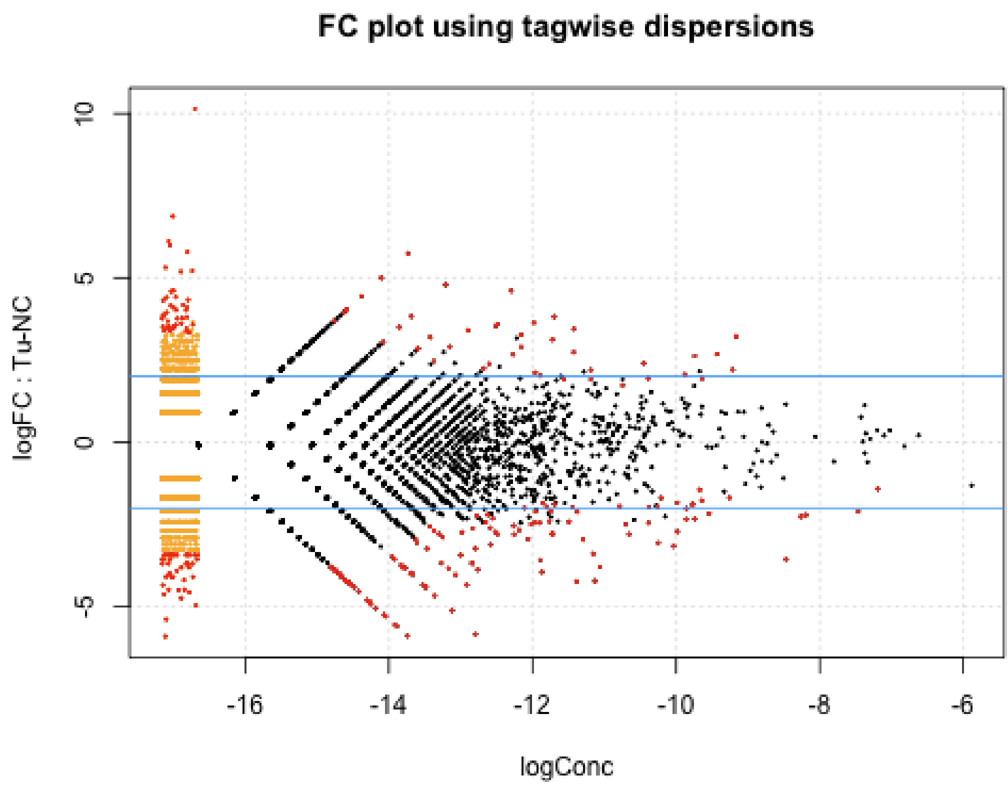


Figure 2: Plot of the log-fold change against the log-concentration for each tag. The 264 most differentially expressed tags as identified by `edgeR` are outlined in red.

4.6 Setup

This analysis of Zhang et al. [1997]'s SAGE data was conducted on:

```
> sessionInfo()

R version 2.9.1 (2009-06-26)
i386-apple-darwin8.11.1

locale:
en_AU.UTF-8/en_AU.UTF-8/C/C/en_AU.UTF-8/en_AU.UTF-8

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

other attached packages:
[1] edgeR_1.3.6

loaded via a namespace (and not attached):
[1] limma_2.18.2
```

and took 2–3 minutes to carry out on an Apple MacBook with a 2.4 Ghz Intel Core 2 Duo processor and 4 Gb of 1067 MHz DDR3 memory.

5 Case Study: deep-sequenced short tags

5.1 Introduction

This section provides a detailed analysis of data from an experiment seeking to compare deep-sequenced tag-based expression profiling to the microarray platforms that had been previously used to conduct such studies ['t Hoen et al., 2008].

5.2 Source of the data

't Hoen et al. [2008] address both biological and technical questions in their study. The biological question addressed was the identification of transcripts differentially expressed in the hippocampus between wild-type mice and transgenic mice overexpressing a splice variant of the δC-doublecortin-like kinase-1 (*Dclk1*) gene. The splice variant, DCLK-short, makes the kinase constitutively active and causes subtle behavioural phenotypes.

On the technical side, the researchers compare the robustness, resolution and inter-lab portability of Solexa/Illumina's DGE tag profiling approach and five microarray platforms ['t Hoen et al., 2008]. The tag-based gene expression technology in this experiment could be thought of as a hybrid between SAGE and RNA-seq—like SAGE it uses short sequence tags ($\sim$ 17bp) to identify transcripts, but it uses the deep sequencing capabilities of Solexa/Illumina 1G Genome Analyzer to greatly increase the number of tags that can be sequenced. For our purposes we will concentrate solely on the DGE data generated in the experiment.

The RNA samples came from wild-type male C57/BL6j mice and transgenic mice overexpressing DCLK-short with a C57/BL6j background. Tissue samples were collected from four individuals in each of the two groups by dissecting out both hippocampi from each mouse. Total RNA was isolated and extracted from the hippocampus cells and sequence tags were prepared using Illumina's Digital Gene Expression Tag Profiling Kit according to the manufacturer's protocol.

Sequencing was done using Solexa/Illumina's Whole Genome Sequencer. RNA from each biological sample was supplied to an individual lane in one Illumina 1G flowcell. The instrument conducted 18 cycles of base incorporation, then image analysis and basecalling were performed using the Illumina Pipeline. Sorting and counting the unique tags followed, and the raw data (tag sequences and counts) are what we will analyze here. 't Hoen et al. [2008] went on to annotate the tags by mapping them back to the genome. In general, the mapping of tags is an important and highly non-trivial part of a DGE experiment, but we shall not deal with this task in this case study.

The researchers obtained ~ 2.4 million sequence tags per sample, with tag abundance spanning four orders of magnitude. They found the results to be highly reproducible, even across laboratories. Using a dedicated Bayesian model, they found 3179 transcripts to be differentially expressed with a FDR of 8.5%. This is a much higher figure than was found for the microarrays. 't Hoen et al. [2008] conclude that deep-sequencing offers a major advance in robustness, comparability and richness of expression profiling data.

5.3 Reading in the data and creating a `DGEList` object

Our first task is to load the `edgeR` package, read the data into `R` and organise the data into an object that the functions in the package can recognise. In this case, the tag counts for the eight individual libraries are stored in eight separate plain text files, `GSM272105.txt`, `GSM272106.txt`, `GSM272318.txt`, `GSM272319.txt`, `GSM272320.txt`, `GSM272321.txt`, `GSM272322.txt` and `GSM272323.txt`.

In each file, the tag IDs and counts for each tag are provided in a table. It is best to create a tab-delimited, plain-text 'Targets' file, which, under the headings 'files', 'group' and 'description', gives the filename, the group and a brief description for each sample.

The `targets` object is produced when the 'Targets.txt' file is read into the `R` session. This object makes a convenient argument to the function `readDGE` which reads the tables of counts into our `R` session, calculates the sizes of the count libraries and produces a `DGEList` object for use by subsequent functions.

```
> setwd("/Users/davismcc/Documents/Honours/Data/Long_SAGE_Data")
> library(edgeR)
> targets <- read.delim(file = "targets.txt", stringsAsFactors = FALSE)
> targets
```

```
          files group                          description
1 GSM272105.txt  DCLK transgenic (Dclk1) mouse hippocampus
2 GSM272106.txt    WT          wild-type mouse hippocampus
3 GSM272318.txt  DCLK transgenic (Dclk1) mouse hippocampus
4 GSM272319.txt    WT          wild-type mouse hippocampus
5 GSM272320.txt  DCLK transgenic (Dclk1) mouse hippocampus
6 GSM272321.txt    WT          wild-type mouse hippocampus
7 GSM272322.txt  DCLK transgenic (Dclk1) mouse hippocampus
8 GSM272323.txt    WT          wild-type mouse hippocampus
```

```
> d <- readDGE(targets, skip = 5, comment.char = "!")
> d
```

```
An object of class "DGEList"
$samples
                  files group                          description
GSM272105 GSM272105.txt  DCLK transgenic (Dclk1) mouse hippocampus
GSM272106 GSM272106.txt    WT          wild-type mouse hippocampus
GSM272318 GSM272318.txt  DCLK transgenic (Dclk1) mouse hippocampus
GSM272319 GSM272319.txt    WT          wild-type mouse hippocampus
GSM272320 GSM272320.txt  DCLK transgenic (Dclk1) mouse hippocampus
GSM272321 GSM272321.txt    WT          wild-type mouse hippocampus
GSM272322 GSM272322.txt  DCLK transgenic (Dclk1) mouse hippocampus
GSM272323 GSM272323.txt    WT          wild-type mouse hippocampus
          lib.size
GSM272105  2685418
GSM272106  3517977
GSM272318  3202246
GSM272319  3558260
GSM272320  2460753
GSM272321   294909
GSM272322   651172
GSM272323  3142280

$counts
                  GSM272105 GSM272106 GSM272318 GSM272319 GSM272320
CATCGCCAGCGGGCACC         1         0         0         0         0
AAGGTCGACTCTGAAGT         1         1         0         0         0
CCTTCCTGGCTCTATGG         1         0         0         0         0
TCTGCTGAGCGTCTGTT         1         0         0         0         0
CCCCAGAGCGAATCAGG         1         1         2         1         1
                  GSM272321 GSM272322 GSM272323
CATCGCCAGCGGGCACC         0         0         0
```

```
AAGGTCGACTCTGAAGT          0          0          0
CCTTCCTGGCTCTATGG          0          0          0
TCTGCTGAGCGTCTGTT          0          0          0
CCCCAGAGCGAATCAGG          0          2          1
844311 more rows ...
```

This DGEList is now ready to be passed to the functions that do the calculations to determine differential expression levels for the genes. Note that when we 'see' the DGEList object d, the counts for just the first five genes in the table are shown, as well as the samples element, which is a data frame constructed from the 'Targets.txt' file and provides the filenames, groups, descriptions and library sizes for the samples.

However, for this dataset, there were over 800 000 unique tags sequenced, most of which have a very small number of counts in total across all libraries. Since it is not possible to achieve statistical significance with fewer than six counts in total for a tag, we filter out tags with five or fewer counts in total—this also helps to speed up the calculations we need to perform. The subsetting capability of DGEList objects makes such filtering very easy to carry out.

```
> d <- d[rowSums(d$counts) > 5, ]
```

Now the dataset is ready to be analysed for differential expression.

5.4 Analysis using common dispersion

5.4.1 Estimating the common dispersion

As discussed for the SAGE data, the first major step in the analysis of DGE data using the NB model is to estimate the dispersion parameter for each tag. Like in the earlier case study, we begin by estimating the common dispersion using the function estimateCommonDisp.

```
> d <- estimateCommonDisp(d)
> names(d)

[1] "samples"           "common.dispersion" "counts"
[4] "pseudo.alt"        "conc"              "common.lib.size"
```

We see in the output below that the total counts in each library of the pseudocounts agrees well with the common library size, as desired.

```
> d$samples$lib.size

[1] 2685418 3517977 3202246 3558260 2460753  294909  651172 3142280

> d$common.lib.size
```

```
[1] 1885653

> colSums(d$pseudo.alt)

GSM272105 GSM272106 GSM272318 GSM272319 GSM272320 GSM272321 GSM272322
  1781297   1783960   1778180   1774107   1789066   1777588   1795899
GSM272323
  1783746
```

Here the coefficient of variation of biological variation (square root of the common dispersion) is found to be 0.44. We also note that a common dispersion estimate of 0.2 means that there is a lot more variability in the data that can be accounted for by the Poisson model—if a tag has just 200 counts, then the estimate of the tag's variance under the NB model is over 40 times greater than it would be under the Poisson model.

```
> d$common.dispersion

[1] 0.1977094

> sqrt(d$common.dispersion)

[1] 0.4446452
```

5.4.2 Testing

Once we have an estimate of the common dispersion, we can proceed with testing procedures for determining differential expression. As for the SAGE data, there are only two groups here, so the pair need not be specified in the call to `exactTest`.

```
> de.common <- exactTest(d)

Comparison of groups:  WT - DCLK
```

The results of the NB exact test can be accessed conveniently using the `topTags` function applied to the object produced by `exactTest`. The table below shows the top 10 DE genes ranked by p-value.

The table in the output from `topTags` shows that the `edgeR` package identifies a good deal of differential expression between the wild-type and the DCLK-transgenic groups. The top DE tags are given very small p-values, even after adjusting for multiple testing. Furthermore, all of the top tags have a large fold change, indicating that these tags are likely to be biologically meaningful. As suggested in the SAGE case study, a Gene Ontology analysis could be carried out using the list of top tags and p-values provided by `topTags` in order to obtain more systematic and functional information about the differentially expressed genes.

```
> topTags(de.common)

Comparison of groups:  WT - DCLK
                    logConc     logFC       PValue          FDR
AATTTCTTCCTCTTCCT -17.29988 11.605093 1.028275e-36 1.149704e-31
CCGTCTTCTGCTTGTCG -10.57698  5.571165 1.076926e-23 6.020501e-19
TCTGTACGCAGTCAGGC -18.46570 -9.731983 7.100854e-23 2.646464e-18
CCGTCTTCTGCTTGTAA -14.44395  5.448158 1.083062e-21 3.027402e-17
CCGTCTTCTGCTTGTCA -15.45499  5.496920 2.759755e-20 6.171309e-16
AAGACTCAGGACTCATC -32.27026 35.491588 7.516087e-20 1.400610e-15
CCGTCTTCTGCTTGTAG -15.57138  4.803709 3.572708e-17 5.453064e-13
AGTGTGACGTGACCGGG -19.06213  8.067070 3.901700e-17 5.453064e-13
AAATTCTTCCTCTTCCT -19.14713  7.910596 2.838381e-16 3.526184e-12
TGTGTATCCCACAAGGG -18.68579  6.864501 5.262632e-16 5.884096e-12
```

The table below shows the raw counts for the tags that edgeR has identified as the most differentially expressed. For these tags there seem to be very large differences between the groups, suggesting that the DE genes identified are truly differentially expressed, and not false positives.

```
> detags.com <- rownames(topTags(de.common)$table)
> d$counts[detags.com, ]

                  GSM272105 GSM272106 GSM272318 GSM272319 GSM272320
AATTTCTTCCTCTTCCT         1        44         0         1         0
CCGTCTTCTGCTTGTCG       106      1485       268       420       601
TCTGTACGCAGTCAGGC       160         0       101         1       440
CCGTCTTCTGCTTGTAA        12        87        21        28        31
CCGTCTTCTGCTTGTCA         2        42         8        17        19
AAGACTCAGGACTCATC         0         6         0         2         0
CCGTCTTCTGCTTGTAG         9        61        11        20        17
AGTGTGACGTGACCGGG         0       249         0         2         1
AAATTCTTCCTCTTCCT         1         6         0         0         0
TGTGTATCCCACAAGGG         1         1         1         0         0
                  GSM272321 GSM272322 GSM272323
AATTTCTTCCTCTTCCT        76         0      3487
CCGTCTTCTGCTTGTCG      5156         5       242
TCTGTACGCAGTCAGGC         0        33         0
CCGTCTTCTGCTTGTAA       352         1        14
CCGTCTTCTGCTTGTCA       183         1        17
AAGACTCAGGACTCATC         4         0       461
CCGTCTTCTGCTTGTAG       133         0         9
AGTGTGACGTGACCGGG         5         0        85
AAATTCTTCCTCTTCCT         2         0       288
TGTGTATCCCACAAGGG         6         0       252
```

If we order the tags by fold change instead of p-value, as in the table below, we see that the genes with the largest fold changes have very small concentrations, and in general the p-values are not as small as when ranked by p-value (not surprisingly). This ranking is dominated by genes that have zero total counts in one group and is less informative than ranking by p-value.

```
> topTags(de.common, sort.by = "logFC")
```

```
Comparison of groups:  WT - DCLK
                    logConc     logFC       PValue          FDR
AAGACTCAGGACTCATC -32.27026  35.49159 7.516087e-20 1.400610e-15
CCTGATGCTACAGAAAA -32.72599  34.58013 5.104779e-15 5.188729e-11
CATAAGTCACAGAGTCG -32.76506 -34.50198 6.541738e-14 5.626348e-10
ACTCTGTGTATTACTCC -32.89435  34.24342 3.132417e-14 2.918603e-10
GATTTTTGTCGTGTTGG -32.95947  34.11317 2.185773e-13 1.745636e-09
AAAAGAAATCACAGTTG -32.96984 -34.09243 7.296827e-12 3.399379e-08
CACATAAGACTTTGGAC -33.09656  33.83899 2.181792e-12 1.434965e-08
AAAATGTTGTTTATGGA -33.10525  33.82162 4.049564e-12 2.156084e-08
GAAATTCTCCATTGATT -33.13607  33.75996 4.049564e-12 2.156084e-08
AAATTATTCCTCTTCCT -33.17108  33.68995 9.017868e-12 4.033115e-08
```

Using their dedicated Bayesian model, 't Hoen et al. [2008] found 3179 transcripts to be differentially expressed with a FDR of 8.5%. We can compare 't Hoen et al. [2008]'s results with the results from `edgeR` by applying the `topTags` function to help look at the tags that have a FDR of less than 0.085 after adjusting for multiple testing using Benjamini and Hochberg [1995]'s method for controlling the FDR.

We see in the output below that 1710 tags (1.5% of the total number analysed) are significantly differentially expressed according to `edgeR` using the common dispersion estimate. Of those tags, 943 (55% of the DE tags) are up-regulated in the wild-type compared with the transgenic samples and 767 (45%) are down-regulated in the wild-type compared with transgenic mice.

```
> sum(p.adjust(de.common$table$p.value, method = "BH") < 0.085)

[1] 1710

> mean(p.adjust(de.common$table$p.value, method = "BH") <
+     0.085) * 100

[1] 1.529394

> top.com <- topTags(de.common, n = 1710)
> sum(top.com$table$logFC > 0)

[1] 943

> sum(top.com$table$logFC < 0)

[1] 767
```

5.4.3 Visualising DGE results

The code for producing the default fold-change plot, with the top 500 most DE tags highlighted in red, is shown below, and the result of this code is shown in Figure 5. In Figure 5, we see that the 500 tags identified as most differentially expressed have large fold changes—almost all of the 500 tags in red fall outside the blue lines at $\log \mathrm{FC} = -2$ and $\log \mathrm{FC} = 2$. This means that most of these tags show at least a 4-fold change in expression level between the samples. This plot suggests strongly that the tags identified by `edgeR` as differentially expressed are truly differentially expressed, and, given the large changes in expression level, are likely to be biologically meaningful.

```
> detags500.com <- rownames(topTags(de.common, n = 500)$table)
```

```
> plotSmear(d, de.tags = detags500.com, main = "FC plot using common dispersion")
> abline(h = c(-2, 2), col = "dodgerblue", lwd = 2)
```

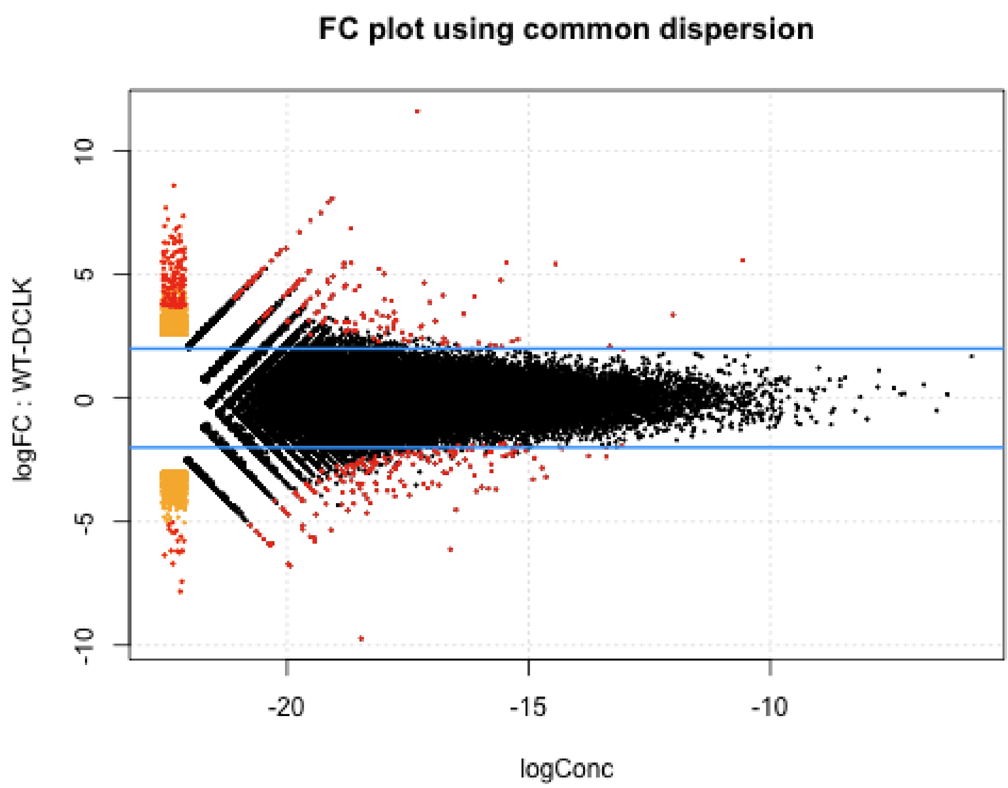


Figure 3: Plot of the log-fold change against the log-concentration for each tag. The 500 most differentially expressed tags as identified by `edgeR` using the common dispersion are outlined in red.

5.5 Analysis using moderated tagwise dispersions

5.5.1 Estimating the tagwise dispersion

An extension to simply using the common dispersion for each tag is to estimate the dispersion separately for each tag, while 'squeezing' these estimates towards the common dispersion estimate in order to improve inference by sharing information between tags. This type of analysis can also be carried out in few steps using the `edgeR` package.

To run the moderated analysis, we need to determine how much moderation is necessary. As discussed in the SAGE case study, we can use an empirical Bayes rule that involves calculating a weight parameter prior.n. However, for many applications (especially if the estimated smoothing parameter is large), using the common dispersion for all tags will give excellent results.

The smoothing parameter, `prior.n` can then be calculated using the function `estimateSmoothing`, which takes the `DGEList` object produced by `estimateCommonDisp` as its argument. As we see below, for this dataset the estimate of the smoothing parameter is very large, so if we were to use this large value for the weight parameter we would moderate the tagwise dispersion estimates so strongly that it would be equivalent to using the common dispersion estimate.

```
> prior.n <- estimateSmoothing(d)
> prior.n
```

```
[1] 15399.17
```

Alternatively, the value for the weight parameter `prior.n` can be selected *a priori*, instead of being estimated. In an experiment such as that we consider here, in which we have eight samples and thus six degrees of freedom for estimating the dispersion parameter, setting the `prior.n` to be ten should be appropriate. This means that the common likelihood receives the weight of ten individual tags, so there will be a reasonable degree of 'squeezing' towards the common dispersion estimate, but there is still enough scope to estimate an individual dispersion for each gene.

The function `estimateTagwiseDisp` produces a DGEList object that contains all of the elements present in the object produced by `estimateCommonDisp`, as well as the value for `prior.n` used (`d$prior.n`) and the tagwise dispersion estimates (`d$tagwise.dispersion`), as we see below.

```
> d <- estimateTagwiseDisp(d, prior.n = 10)
```

```
Using grid search to estimate tagwise dispersion.
```

```
> names(d)
```

```
[1] "samples"            "common.dispersion"  "prior.n"
[4] "tagwise.dispersion" "counts"             "pseudo.alt"
[7] "conc"               "common.lib.size"
```

```
> d$prior.n
```

```
[1] 10
```

```
> head(d$tagwise.dispersion)
```

```
[1] 0.1997564 0.1854905 0.1925808 0.1579529 0.2070189 0.1854905
```

It is interesting to consider the distribution of the tagwise dispersion estimates. As we can see from the output below, the tagwise dispersion estimates range from a minimum of 0.09 to a maximum of 0.97, and the common dispersion estimate lies in between the median and mean values for the tagwise dispersion estimates.

```
> summary(d$tagwise.dispersion)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.08842 0.18550 0.19260 0.19930 0.20700 0.99000
```

```
> d$common.dispersion
```

```
[1] 0.1977094
```

5.5.2 Testing

Once we have an estimate of the common dispersion and/or estimates of the tagwise dispersions, we can proceed with testing procedures for determining differential expression using `exactTest`.

By default, `exactTest` uses the common dispersion, but by adding the argument `common.disp=FALSE`, tagwise dispersion estimates will be used instead.

```
> de.tagwise <- exactTest(d, common.disp = FALSE)
```

```
Comparison of groups:  WT - DCLK
```

Just as we saw earlier, the object produced by `exactTest` contains two elements. The first is a data frame (`table`) that contains the elements `logConc`, `logFC` and `p.value` and the second is a vector (`comparison`) that lists the names of the groups being compared.

The output below shows that when using tagwise dispersions, the `edgeR` package still identifies a lot of differential expression between the wild-type group and the DCLK-transgenic group. The top DE tags are given very small p-values, even after adjusting for multiple testing. However, We see immediately that the p-values for the top tags are many orders of magnitude greater than those for the top tags identified using the common dispersion.

As with the analysis using the common dispersion, all of the top tags have a large fold change, indicating that these changes in expression are likely to be biologically meaningful, although interestingly we see more tags (7 out of 10) that are down-regulated in the wild-type group compared with the DCLK group, which contrasts with using the common dispersion. We note that the ranking of the tags is different, too, and only three of the top ten tags according to using the common dispersion are found to be among the top ten tags using tagwise dispersions.

```
> topTags(de.tagwise)
```

```
Comparison of groups:  WT - DCLK
                    logConc      logFC       PValue          FDR
TCTGTACGCAGTCAGGC -18.46778  -9.732317 2.271105e-18 2.539300e-13
CATAAGTCACAGAGTCG -32.76303 -34.506051 4.507171e-15 2.519711e-10
CCAAGAATCTGGTCGTA -17.46762  -3.929366 1.966857e-12 7.330412e-08
AATTTCTTCCTCTTCCT -17.30380  11.607564 3.139447e-12 8.775460e-08
ATACTGACATTTCGTAT -16.77800   4.144907 6.097301e-12 1.363466e-07
GCTAATAAATGGCAGAT -14.91149  -3.295240 9.212532e-12 1.716740e-07
CTGCTAAGCAGAAGCAA -16.99434  -3.427292 1.803488e-11 2.880660e-07
TTCCTGAAAATGTGAAG -17.08403  -3.639794 2.653336e-11 3.708335e-07
AAAAGAAATCACAGTTG -32.97330 -34.085504 6.032072e-11 7.493777e-07
AGTGTGACGTGACCGGG -19.07157   8.035868 1.581782e-10 1.768575e-06
```

Of course, we can also rank the top tags using the fold change instead of the p-value. The results of doing this are shown in the table below, but this ranking is dominated by genes that have zero total counts in one group and is less informative than ranking by p-value.

```
> topTags(de.tagwise, n = 10, sort.by = "logFC")
```

```
Comparison of groups:  WT - DCLK
                    logConc     logFC       PValue          FDR
AAGACTCAGGACTCATC -32.29053  35.45105 1.216554e-08 5.667572e-05
CCTGATGCTACAGAAAA -32.75498  34.52215 4.898918e-07 1.165411e-03
CATAAGTCACAGAGTCG -32.76303 -34.50605 4.507171e-15 2.519711e-10
ACTCTGTGTATTACTCC -32.92486  34.18238 2.898915e-08 1.200463e-04
AAAAGAAATCACAGTTG -32.97330 -34.08550 6.032072e-11 7.493777e-07
GATTTTTGTCGTGTTGG -33.00084  34.03043 1.747407e-07 4.765265e-04
CACATAAGACTTTGGAC -33.13007  33.77197 2.927280e-06 3.596662e-03
GAAATTCTCCATTGATT -33.15363  33.72485 7.690677e-07 1.500202e-03
AAAATGTTGTTTATGGA -33.15760  33.71690 5.539764e-06 5.567035e-03
AAATTATTCCTCTTCCT -33.18808  33.65595 1.494026e-06 2.456552e-03
```

The tables below shows the raw counts for the genes that `edgeR` has identified as the most differentially expressed, using the common dispersion and tagwise dispersions. For these tags, using both methods, there seem to be very large differences between the groups, suggesting that the DE genes identified are truly differentially expressed, and not false positives.

Particularly noteworthy, however, is how much more consistent the counts *within* groups are for the top tags identified using tagwise dispersions compared with those identified using the common dispersion. This is to be expected, as allowing tagwise dispersions penalises highly variable tags, so those that have greater variability within groups (especially one or two libraries with extremely high counts) will appear far lower in the ranking using tagwise dispersions than they would using the common dispersion. This difference in the rankings provided by the two approaches to the dispersion parameter could yield valuable information.

```
> detags.tgw <- rownames(topTags(de.tagwise)$table)
> detags.com <- rownames(topTags(de.common)$table)
> d$counts[detags.tgw, ]
```

```
                 GSM272105 GSM272106 GSM272318 GSM272319 GSM272320
TCTGTACGCAGTCAGGC       160         0       101         1       440
CATAAGTCACAGAGTCG        67         0        77         0        58
CCAAGAATCTGGTCGTA        70         3        66         5        47
AATTTCTTCCTCTTCCT         1        44         0         1         0
ATACTGACATTTCGTAT         5       113         5       228         8
GCTAATAAATGGCAGAT       387        45       321        32       132
CTGCTAAGCAGAAGCAA        76         7        88         7        52
TTCCTGAAAATGTGAAG        74         6        70         9        86
AAAAGAAATCACAGTTG        31         0        90         0        42
AGTGTGACGTGACCGGG         0       249         0         2         1
                 GSM272321 GSM272322 GSM272323
TCTGTACGCAGTCAGGC         0        33         0
CATAAGTCACAGAGTCG         0         7         0
CCAAGAATCTGGTCGTA         0        13         7
AATTTCTTCCTCTTCCT        76         0      3487
ATACTGACATTTCGTAT         4         1       104
GCTAATAAATGGCAGAT         1        71        38
CTGCTAAGCAGAAGCAA         0        15        11
TTCCTGAAAATGTGAAG         0        10         7
AAAAGAAATCACAGTTG         0         3         0
AGTGTGACGTGACCGGG         5         0        85
```

```
> d$counts[detags.com, ]
```

```
                 GSM272105 GSM272106 GSM272318 GSM272319 GSM272320
AATTTCTTCCTCTTCCT         1        44         0         1         0
CCGTCTTCTGCTTGTCG       106      1485       268       420       601
TCTGTACGCAGTCAGGC       160         0       101         1       440
CCGTCTTCTGCTTGTAA        12        87        21        28        31
CCGTCTTCTGCTTGTCA         2        42         8        17        19
AAGACTCAGGACTCATC         0         6         0         2         0
CCGTCTTCTGCTTGTAG         9        61        11        20        17
AGTGTGACGTGACCGGG         0       249         0         2         1
AAATTCTTCCTCTTCCT         1         6         0         0         0
TGTGTATCCCACAAGGG         1         1         1         0         0
                 GSM272321 GSM272322 GSM272323
AATTTCTTCCTCTTCCT        76         0      3487
CCGTCTTCTGCTTGTCG      5156         5       242
```

```
TCTGTACGCAGTCAGGC        0          33          0
CCGTCTTCTGCTTGTAA      352           1         14
CCGTCTTCTGCTTGTCA      183           1         17
AAGACTCAGGACTCATC        4           0        461
CCGTCTTCTGCTTGTAG      133           0          9
AGTGTGACGTGACCGGG        5           0         85
AAATTCTTCCTCTTCCT        2           0        288
TGTGTATCCCACAAGGG        6           0        252
```

We might also be interested in comparing the top-ranking genes as identified by `edgeR` using the common dispersion and tagwise dispersions. The output below shows, firstly, that there are three tags that appear in the top ten most DE tags using both common and tagwise dispersions. Secondly, we see that of the top 1000 most DE tags as identified using tagwise dispersions, 77% of these tags are also in the list of the 1000 most DE tags as identified using the common dispersion. This shows that although we do get quite different results depending on which method we use, there is still a great deal of agreement as to which tags are DE.

```
> sum(rownames(topTags(de.tagwise)$table) %in% rownames(topTags(de.common)$table))

[1] 3

> sum(rownames(topTags(de.tagwise, n = 1000)$table) %in% rownames(topTags(de.common,
+     n = 1000)$table))/1000 * 100

[1] 76.7
```

Using their dedicated Bayesian model, 't Hoen et al. [2008] found 3179 transcripts to be differentially expressed with a FDR of 8.5%. The output below shows that using Benjamini and Hochberg [1995]'s approach for controlling the FDR at 8.5%, `edgeR` identifies 1717 tags as DE using common dispersion and 1441 tags as DE using tagwise dispersions. This means that we determine 1.54% and 1.29% of tags to be DE using common and tagwise dispersions, respectively.

```
> sum(p.adjust(de.common$table$p.value, method = "BH") < 0.085)

[1] 1710

> mean(p.adjust(de.common$table$p.value, method = "BH") <
+     0.085) * 100

[1] 1.529394

> sum(p.adjust(de.tagwise$table$p.value, method = "BH") <
+     0.085)
```

```
[1] 1441
```

```
> mean(p.adjust(de.tagwise$table$p.value, method = "BH") <
+     0.085) * 100
```

```
[1] 1.288805
```

Of the 1441 tags identified as DE using tagwise dispersions, 729 (51%) are up-regulated in wild-type and 712 (49%) are up-regulated in the transgenic mice. The proportions of up- and down-regulated genes identified using the two approaches to modeling the dispersion are similar, but using the common dispersion identifies slightly more tags up-regulated in wild-type mice as DE.

```
> top.tgw <- topTags(de.tagwise, n = 1441)
> sum(top.tgw$table$logFC > 0)
```

```
[1] 729
```

```
> sum(top.tgw$table$logFC < 0)
```

```
[1] 712
```

5.5.3 Visualising DGE results

As discussed earlier, the function `plotSmear` can be used to generate a plot of the log-fold change against the log-concentration for each tag (analogous to an MA-plot in the microarray context). We identify the top 500 most DE tags using both common dispersion and tagwise dispersions so we can highlight them on the plots and compare what we see. The code for producing the fold-change plots is shown below, and the result of this code is shown in Figure 6.

```
> detags500.com <- rownames(topTags(de.common, n = 500)$table)
> detags500.tgw <- rownames(topTags(de.tagwise, n = 500)$table)
```

```
> par(mfcol = c(2, 1))
> plotSmear(d, de.tags = detags500.com, main = "Using common dispersion")
> abline(h = c(-2, 2), col = "dodgerblue", lwd = 2)
> plotSmear(d, de.tags = detags500.tgw, main = "Using tagwise dispersions")
> abline(h = c(-2, 2), col = "dodgerblue", lwd = 2)
```

In Figure 6, the top 500 most differentially expressed tags (those identified as significant by `edgeR` using the common dispersion (top) and tagwise dispersions (bottom)) are highlighted in red. Looking at Figure 6, we see that, generally speaking, the pattern of differential expression looks similar using the two different methods, and the tags identified as DE have convincingly large fold changes.

5.6 Setup

This analysis of 't Hoen et al. [2008]'s tag-based DGE data was conducted on:

```
> sessionInfo()

R version 2.9.1 (2009-06-26)
i386-apple-darwin8.11.1

locale:
en_AU.UTF-8/en_AU.UTF-8/C/C/en_AU.UTF-8/en_AU.UTF-8

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

other attached packages:
[1] edgeR_1.3.6

loaded via a namespace (and not attached):
[1] limma_2.18.2
```

and took 5–10 minutes to carry out on an Apple MacBook with a 2.4 Ghz Intel Core 2 Duo processor and 4 Gb of 1067 MHz DDR3 memory.

6 Case Study: RNA-seq data

6.1 Introduction

This section provides a detailed analysis of data from a study by Li et al. [2008] designed to address a range of practical issues in RNA-seq experiments:

1. How many annotated genes are detected in a single cell type?

2. What is the number of tags that is necessary for the analysis of differentially regulated genes under different experimental conditions?

3. To what extent can different mRNA isoforms be detected?

4. How can one quantify alternative splicing by using a single or combination of existing technologies?

Li et al. [2008] attempt to address all of these issues on an androgen-sensitive prostate cancer cell model. We are interested primarily in the second question, and the challenge of identifying differentially regulated genes under different experimental conditions. We will demonstrate the use of the edgeR package for analyzing RNA-seq data for differential gene expression.

6.2 Source of the data

Li et al. [2008] sequenced poly$(A)^+$ RNA from mock-treated or androgen sensitive LNCaP cells (a cell line of human cells commonly used in the field of oncology) on the Illumina 1G Genome Analyzer. The researchers used a double-random priming approach that was capable of generating strand-specific information, although this is not of relevance to our analysis here. The raw RNA-seq data provided by Li et al. consists of 7 'lanes' of 35bp reads. [1] Approximately 10 million sequence tags were generated from both control and hormone-treated cells (treated with DHT), and Li et al. [2008]'s analysis suggests that this tag density is sufficient for quantitative analysis of gene expression.

The 10 million sequenced tags arise from four libraries from control cells and three libraries for hormone-treated cells, giving a total of seven libraries to analyse. From Li et al. [2008] and its companion paper [Li et al., 2006] it is unclear as to whether the treatments are independent or not. The following analysis shows how a quantitative analysis of gene expression, focusing on identifying differentially expressed genes, can be conducted for these seven libraries using `edgeR`.

6.3 Reading in the data and creating a `DGEList` object

Our first task is to load the `edgeR` package and read the data into `R`. In this case, the tag counts for the libraries are stored in a single table in a plain text file `pnas_expression.txt`, in which the rows of the table represent tags and the columns represent the different libraries.

To turn the raw RNA-seq data into a table of counts, reads were mapped to the NCBI36 build of the human genome using `bowtie`, allowing up to two mismatches. Reads which did not map uniquely were discarded. The number of mapped reads that overlapped ENSEMBL gene annotations (version 53) was then counted. In counting reads associated with genes, reads which mapped to non-coding gene regions, such as introns, were included in the count.

Unlike in the other datasets we have look at, counts here are aggregated at the gene, not at the tag, level.

The `files` object provides the name of the data file, and makes a convenient argument to the function `read.delim` which reads the table of counts into our `R` session.

```
> setwd("/Users/davismcc/Documents/Honours/Data/LiData")
> library(edgeR)
> raw.data <- read.delim("pnas_expression.txt")
> names(raw.data)

[1] "ensembl_ID" "lane1"      "lane2"      "lane3"      "lane4"
[6] "lane5"      "lane6"      "lane8"      "len"
```

The raw data is stored in a table with columns representing the gene names, the counts for the seven libraries and a column giving the length of each gene. The gene length is not currently used

[1] The Illumina instrument requires samples to be placed in a 'flow cell' which contains eight 'lanes'—each lane has a sample of cDNA and generates a library of sequence counts for that sample.

by edgeR, but this information could be used in future versions of the package. In the code below, we assign the counts matrix to an object d with the appropriate rownames, define the groups to which the samples belong, and then pass these arguments to DGEList, which calculates the library sizes and constructs a DGEList containing all of the data we require for the analysis.

```
> d <- raw.data[, 2:8]
> rownames(d) <- raw.data[, 1]
> group <- c(rep("Control", 4), rep("DHT", 3))
> d <- DGEList(counts = d, group = group)
> d
```

```
An object of class "DGEList"
$samples
        group lib.size
lane1 Control   978576
lane2 Control  1156844
lane3 Control  1442169
lane4 Control  1485604
lane5     DHT  1823460
lane6     DHT  1834335
lane8     DHT   681743

$counts
                lane1 lane2 lane3 lane4 lane5 lane6 lane8
ENSG00000124208   478   619   628   744   483   716   240
ENSG00000182463    27    20    27    26    48    55    24
ENSG00000124201   180   218   293   275   373   301    88
ENSG00000124205     0     0     5     5     0     0     0
ENSG00000124207    76    80    85    97    80    81    37
21872 more rows ...
```

This DGEList is now ready to be passed to the functions that do the calculations to determine differential expression levels for the genes.

6.4 Analysis using common dispersion

6.4.1 Estimating the common dispersion

As discussed for the SAGE data, the first major step in the analysis of DGE data using the NB model is to estimate the dispersion parameter for each tag. Like in the earlier case study, we begin by estimating the common dispersion using the function estimateCommonDisp, and analysing the data using the common dispersion.

```
> d <- estimateCommonDisp(d)
> names(d)
```

```
[1] "samples"           "common.dispersion" "counts"
[4] "pseudo.alt"        "conc"              "common.lib.size"
```

```
> d
```

```
An object of class "DGEList"
$samples
        group lib.size
lane1 Control   978576
lane2 Control  1156844
lane3 Control  1442169
lane4 Control  1485604
lane5     DHT  1823460
lane6     DHT  1834335
lane8     DHT   681743

$common.dispersion
[1] 0.02021596

$counts
                lane1 lane2 lane3 lane4 lane5 lane6 lane8
ENSG00000124208   478   619   628   744   483   716   240
ENSG00000182463    27    20    27    26    48    55    24
ENSG00000124201   180   218   293   275   373   301    88
ENSG00000124205     0     0     5     5     0     0     0
ENSG00000124207    76    80    85    97    80    81    37
21872 more rows ...

$pseudo.alt
                      lane1       lane2      lane3      lane4     lane5
ENSG00000124208 623.7011345 682.94207324 555.660542 639.479771 336.71526
ENSG00000182463  34.3356362  22.18976707  23.832567  22.180621  33.16394
ENSG00000124201 235.0851553 240.62900197 259.559061 236.246548 262.61397
ENSG00000124205   0.1502441   0.05777303   4.539068   4.427231   0.00000
ENSG00000124207  98.4667235  88.25763482  75.005547  83.279817  55.73657
                    lane6        lane8
ENSG00000124208 499.57554 4.485689e+02
ENSG00000182463  38.29654 4.367427e+01
ENSG00000124201 209.30644 1.685512e+02
ENSG00000124205   0.00000 1.509887e-10
```

```
ENSG00000124207  56.12481 6.774033e+01
21872 more rows ...

$conc
$conc.common
ENSG00000124208 ENSG00000182463 ENSG00000124201 ENSG00000124205
   4.236152e-04    2.419602e-05    1.809021e-04    1.063407e-06
ENSG00000124207
   5.831078e-05
21872 more elements ...

$conc.group
                     Control          DHT
ENSG00000124208 4.897757e-04 3.350953e-04
ENSG00000182463 1.990269e-05 2.966863e-05
ENSG00000124201 1.902894e-04 1.684999e-04
ENSG00000124205 1.963536e-06 8.783496e-16
ENSG00000124207 6.735245e-05 4.640575e-05
21872 more rows ...


$common.lib.size
[1] 1276768
```

The output of `estimateCommonDisp` is a DGEList object with several new elements. The element `common.dispersion`, as the name suggests, provides the estimate of the common dispersion. The pseudocounts calculated under the alternative hypothesis are given by `pseudo.alt`. The element `conc` gives the estimates of the overall concentration of each tag across all of the original samples (`conc$conc.common`) and the estimate of the concentration of each tag within each group (`conc$conc.group`). The element `common.lib.size` gives the library size to which the original libraries have been adjusted in the pseudocounts.

We see in the output below that the total counts in each library of the pseudocounts agrees well with the common library size, as desired.

```
> d$samples$lib.size

[1]  978576 1156844 1442169 1485604 1823460 1834335  681743

> d$common.lib.size

[1] 1276768

> colSums(d$pseudo.alt)
```

```
  lane1   lane2   lane3   lane4   lane5   lane6   lane8
1277021 1276791 1276935 1277035 1277209 1277069 1277919
```

Here the coefficient of variation of biological variation (square root of the common dispersion) is found to be 0.142. We also note that although a common dispersion estimate of 0.02 may seem 'small', if a tag has just 200 counts, then the estimate of the tag's variance is 5 times greater than it would be under the Poisson model.

```
> d$common.dispersion

[1] 0.02021596

> sqrt(d$common.dispersion)

[1] 0.1421828
```

6.4.2 Testing

Once we have an estimate of the common dispersion, we can proceed with testing procedures for determining differential expression. As for the SAGE data, there are only two groups here, so the pair need not be specified in the call to `exactTest`.

```
> de.com <- exactTest(d)

Comparison of groups:  DHT - Control

> names(de.com)

[1] "table"      "comparison"
```

The results of the NB exact test can be accessed conveniently using the `topTags` function applied to the object produced by `exactTest`. The table below shows the top 10 DE genes ranked by p-value.

The table in the output from `topTags` shows that the `edgeR` package identifies a great deal of differential expression, and gives the top genes extremely small p-values, even after adjusting for multiple testing. Furthermore, all of the top genes have a very large fold change (indicating that these tags are likely to be biologically meaningful), and all are up-regulated in the DHT-treatment group compared to the control group.

Of course, for many applications the ranking for differential expression is more important than the p-value, and `topTags` provides such a ranking. As suggested in the SAGE case study, a Gene Ontology analysis could be carried out using the list of top gene and p-values provided by `topTags` in order to obtain more systematic and functional information about the differentially expressed genes.

```
> topTags(de.com)

Comparison of groups:  DHT - Control
                  logConc    logFC        PValue           FDR
ENSG00000151503 -11.94799 5.705233 7.744047e-185 1.694165e-180
ENSG00000096060 -11.33288 4.893134 5.349073e-155 5.851083e-151
ENSG00000127954 -15.63280 8.118692 7.331162e-148 5.346128e-144
ENSG00000166451 -12.28742 4.570439 7.122773e-128 3.895623e-124
ENSG00000131016 -14.42856 5.190737 1.701099e-104 7.442990e-101
ENSG00000113594 -12.83343 4.000650  2.579172e-96  9.404091e-93
ENSG00000116285 -13.56732 4.087861  8.108902e-88  2.534264e-84
ENSG00000123983 -12.09539 3.544802  1.444749e-86  3.950846e-83
ENSG00000166086 -15.24551 5.390848  6.219612e-86  1.511849e-82
ENSG00000162772 -10.81704 3.201950  7.560702e-80  1.654055e-76
```

The table below shows the raw counts for the genes that `edgeR` has identified as the most differentially expressed. For these genes there seems to be very large differences between the groups, suggesting that the DE genes identified are truly differentially expressed.

```
> detags.com <- rownames(topTags(de.com)$table)
> d$counts[detags.com, ]

                lane1 lane2 lane3 lane4 lane5 lane6 lane8
ENSG00000151503    35    35    49    59  3307  3439  1224
ENSG00000096060    65    79   105   113  3975  3727  1451
ENSG00000127954     0     0     3     3   607   602   220
ENSG00000166451    41    52    57    57  1750  1654   728
ENSG00000131016     9     5    18     6   564   377   213
ENSG00000113594    37    36    57    43   936   959   418
ENSG00000116285    18    28    23    32   645   630   218
ENSG00000123983    62    76    94   108  1354  1258   628
ENSG00000166086     9     2     3     6   296   298   121
ENSG00000162772   172   204   250   304  2972  3269  1112
```

If we order the genes by fold change instead of p-value, we see that the genes with the largest fold changes have very small concentrations. This ranking is dominated by genes that have zero total counts in one group and is less informative than ranking by p-value.

```
> topTags(de.com, n = 10, sort.by = "logFC")

Comparison of groups:  DHT - Control
                  logConc     logFC       PValue          FDR
ENSG00000091972 -31.75811 -36.51589 8.689939e-56 6.789635e-53
```

```
ENSG00000164120 -32.27738  35.47735 4.636882e-44 2.205241e-41
ENSG00000100373 -32.93100 -34.17010 2.739292e-17 2.147939e-15
ENSG00000118513 -33.00607 -34.01998 8.893707e-16 5.895989e-14
ENSG00000081237 -33.15786 -33.71640 5.634630e-13 2.691459e-11
ENSG00000196660 -33.22302 -33.58606 3.870150e-12 1.647223e-10
ENSG00000117245 -33.23863 -33.55484 1.051358e-11 4.151725e-10
ENSG00000019549 -33.39510  33.24191 2.420329e-13 1.225684e-11
ENSG00000137404 -33.41447 -33.20316 1.041329e-08 2.556807e-07
ENSG00000059804 -33.43964  33.15284 2.174556e-12 9.630115e-11
```

We can see how many genes are identified as differentially expressed between the control group (untreated LNCaP cells) and the DHT-treated LNCaP cells, for a given threshold for the exact p-value or for the adjusted p-value.

As the output below shows, `edgeR` detects a huge number of differentially expressed genes in this dataset. Almost 5000 genes are given a p-value less than 0.01.

```
> sum(de.com$table$p.value < 0.01)
```

```
[1] 4760
```

The output below shows that over 4835 genes are given an adjusted p-value of less than 0.05. This means that if we set our control the FDR for differential expression at 5%, then `edgeR` identifies 22% of all the genes in the dataset as differentially expressed.

```
> sum(p.adjust(de.com$table$p.value, method = "BH") < 0.05)
```

```
[1] 4835
```

```
> mean(p.adjust(de.com$table$p.value, method = "BH") < 0.05) *
+     100
```

```
[1] 22.10084
```

Of the genes identified as DE above, 1911 (40% of the DE genes) are up-regulated in DHT-treated compared with control cells, and 2924 (60%) are up-regulated in the control cells compared with DHT-treated cells. It is interesting to note that although we detect far more genes as DE that are up-regulated in the control group, all of the top ten genes were up-regulated in the DHT-treated group.

```
> top.com <- topTags(de.com, n = 4835)
> sum(top.com$table$logFC > 0)
```

```
[1] 1911
```

```
> sum(top.com$table$logFC < 0)
```

```
[1] 2924
```

6.4.3 Visualising DGE results

The code for producing the default fold-change plot, with the top 500 most DE tags highlighted in red, is shown below, and the result of this code is shown in Figure 5. In Figure 5, we see that the 500 tags identified as most differentially expressed have large fold changes—almost all of the 500 tags in red fall outside the blue lines at $\log \mathrm{FC} = -2$ and $\log \mathrm{FC} = 2$. This means that most of these tags show at least a 4-fold change in expression level between the samples. This plot suggests strongly that the tags identified by `edgeR` as differentially expressed are truly differentially expressed, and, given the large changes in expression level, are likely to be biologically meaningful.

```
> detags500.com <- rownames(topTags(de.com, n = 500)$table)

> plotSmear(d, de.tags = detags500.com, main = "FC plot using common dispersion")
> abline(h = c(-2, 2), col = "dodgerblue", lwd = 2)
```

6.5 Analysis using moderated tagwise dispersions

6.5.1 Estimating the tagwise dispersion

As discussed in the previous case studies, an extension to simply using the common dispersion for each tag is to estimate the dispersion separately for each tag, while 'squeezing' these estimates towards the common dispersion estimate. The goal of this moderation of the dispersion estimates is to improve inference by sharing information between tags. This type of analysis can be carried out in few steps using the `edgeR` package.

To run the moderated analysis, we need to determine how much moderation is necessary. As discussed in the SAGE case study, we can use an empirical Bayes rule that involves calculating a weight parameter prior.n. However, for many applications (especially if the estimated smoothing parameter is large), using the common dispersion for all tags will give excellent results.

The smoothing parameter, `prior.n` can then be calculated using the function `estimateSmoothing`, which takes the `DGEList` object produced by `estimateCommonDisp` as its argument. As we see below, for this dataset the estimate of the smoothing parameter is very small, so if we were to use this tiny value for the weight parameter we would moderate the tagwise dispersion estimates so little that it would be equivalent to using just using the tagwise dispersion for each gene without any 'squeezing' at all towards the common dispersion.

```
> prior.n <- estimateSmoothing(d)
> prior.n

[1] 0.0008377133
```

As we only have seven libraries, a small sample size, we should not be too confident about the accuracy of the tagwise dispersions. Therefore it is recommended to use a larger value for `prior.n`, which can be selected *a priori*, instead of being estimated. In an experiment such as that we consider here, in which we have seven samples and thus five degrees of freedom for estimating the dispersion parameter, setting the `prior.n` to be ten should be appropriate. This means that the common likelihood receives the weight of ten individual tags, so there will be a reasonable degree of 'squeezing' towards the common dispersion estimate, but there is still enough scope to allow flexibility with the individual dispersion for each gene.

The function `estimateTagwiseDisp` produces a DGEList object that contains all of the elements present in the object produced by `estimateCommonDisp`, as well as the value for `prior.n` used (`d$prior.n`) and the tagwise dispersion estimates (`d$tagwise.dispersion`), as we see below. Here we set `grid.length=500` for greater precision in the tagwise dispersion estimates.

```
> d <- estimateTagwiseDisp(d, prior.n = 10, grid.length = 500)

Using grid search to estimate tagwise dispersion.

> names(d)

[1] "samples"            "common.dispersion"  "prior.n"
[4] "tagwise.dispersion" "counts"             "pseudo.alt"
[7] "conc"               "common.lib.size"

> d$prior.n

[1] 10

> head(d$tagwise.dispersion)

[1] 0.01936799 0.01729400 0.01936799 0.02145046 0.01522843 0.02774923
```

It is interesting to consider the distribution of the tagwise dispersion estimates. As we can see from the output below, the tagwise dispersion estimates range from a minimum of 0.005 to a maximum of 0.236, and the common dispersion estimate lies in between the median and mean values for the tagwise dispersion estimates.

```
> summary(d$tagwise.dispersion)

    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.005025 0.019370 0.019370 0.020640 0.021450 0.236100

> d$common.dispersion

[1] 0.02021596
```

6.5.2 Testing

Once we have an estimate of the common dispersion and/or estimates of the tagwise dispersions, we can proceed with testing procedures for determining differential expression using `exactTest`.

By default, `exactTest` uses the common dispersion, but by adding the argument `common.disp=FALSE`, tagwise dispersion estimates will be used instead.

```
> de.tgw <- exactTest(d, common.disp = FALSE)

Comparison of groups:  DHT - Control
```

The output below shows that when using tagwise dispersions, the `edgeR` package still identifies a huge amount of differential expression between the control group and the DHT-treated group. The top DE tags are given even smaller p-values than using the common dispersion—many, many orders of magnitude smaller.

As with the analysis using the common dispersion, all of the top genes have large fold change, indicating that these changes in expression are likely to be biologically meaningful. Again, all of the top genes are up-regulated in the DHT-treated group compared with the control group. We note that the ranking of the tags is similar, with seven of the top ten genes using the common dispersion to be found among the top ten genes using tagwise dispersions.

```
> topTags(de.tgw)

Comparison of groups:  DHT - Control
                  logConc    logFC        PValue           FDR
ENSG00000151503 -11.947221 5.704024 2.214558e-301 4.844788e-297
ENSG00000096060 -11.332026 4.891305 1.545034e-259 1.690035e-255
ENSG00000166451 -12.288475 4.570589 8.545084e-180 6.231360e-176
ENSG00000127954 -15.632016 8.117608 5.274185e-171 2.884584e-167
ENSG00000162772 -10.816415 3.201287 6.821241e-136 2.984566e-132
ENSG00000113594 -12.834292 3.999954 8.573960e-119 3.126209e-115
ENSG00000116133 -11.741194 3.128433 2.367018e-118 7.397607e-115
ENSG00000116285 -13.566933 4.089714 4.303139e-116 1.176747e-112
ENSG00000115648  -8.831597 2.481756 1.967977e-114 4.783715e-111
ENSG00000130066 -10.322217 2.494893 6.742514e-108 1.475060e-104
```

Of course, we can also rank the top tags using the fold change instead of the p-value, as described above, this ranking is dominated by genes that have zero total counts in one group and is less informative than ranking by p-value.

```
> topTags(de.tgw, n = 10, sort.by = "logFC")
```

```
Comparison of groups:  DHT - Control
                 logConc     logFC       PValue          FDR
ENSG00000091972 -31.75810 -36.51592 2.059993e-63 1.502216e-60
ENSG00000164120 -32.27722  35.47767 1.732888e-44 5.123027e-42
ENSG00000100373 -32.93097 -34.17017 1.170463e-17 7.759461e-16
ENSG00000118513 -33.00748 -34.01715 6.961165e-15 3.696345e-13
ENSG00000081237 -33.15838 -33.71535 1.283161e-12 5.113246e-11
ENSG00000196660 -33.22317 -33.58577 3.427356e-12 1.288321e-10
ENSG00000117245 -33.23845 -33.55521 1.635293e-11 5.651708e-10
ENSG00000019549 -33.39438  33.24334 2.743940e-13 1.175645e-11
ENSG00000059804 -33.43993  33.15225 2.018339e-12 7.801273e-11
ENSG00000137404 -33.44594 -33.14022 5.612659e-07 9.183854e-06
```

The tables below shows the quantile-adjusted counts (i.e. counts for equalised library sizes) for the genes that `edgeR` has identified as the most differentially expressed, using the common dispersion and tagwise dispersions. For these tags, using both methods, there seem to be very large differences between the groups, suggesting that the DE genes identified are truly differentially expressed, and not false positives.

We saw for 't Hoen et al. [2008]'s data how much more consistent the counts *within* groups are for the top tags identified using tagwise dispersions compared with those identified using the common dispersion. This effect is not nearly as pronounced here.

```
> detags.tgw <- rownames(topTags(de.tgw)$table)
> detags.com <- rownames(topTags(de.com)$table)
> round(d$pseudo.alt[detags.tgw, ])

                lane1 lane2 lane3 lane4 lane5 lane6 lane8
ENSG00000151503    46    39    43    51  2315  2394  2293
ENSG00000096060    85    87    93    97  2783  2594  2717
ENSG00000166451    53    57    50    49  1225  1151  1361
ENSG00000127954     0     0     3     3   425   419   412
ENSG00000162772   225   225   221   262  2081  2276  2083
ENSG00000113594    48    40    51    37   655   667   780
ENSG00000116133   127   118   137   123  1143  1121  1041
ENSG00000116285    24    31    20    28   452   439   409
ENSG00000115648  1226  1196  1166  1156  6813  6920  6130
ENSG00000130066   403   426   440   410  2239  2301  2571

> round(d$pseudo.alt[detags.com, ])

                lane1 lane2 lane3 lane4 lane5 lane6 lane8
ENSG00000151503    46    39    43    51  2315  2394  2293
ENSG00000096060    85    87    93    97  2783  2594  2717
```

```
ENSG00000127954      0      0      3      3    425    419    412
ENSG00000166451     53     57     50     49   1225   1151   1361
ENSG00000131016     12      6     16      5    396    261    396
ENSG00000113594     48     40     51     37    655    667    780
ENSG00000116285     24     31     20     28    452    439    409
ENSG00000123983     81     84     83     93    948    875   1172
ENSG00000166086     11      2      3      5    207    207    226
ENSG00000162772    225    225    221    262   2081   2276   2083
```

We might also be interested in comparing the top-ranking genes as identified by `edgeR` using the common dispersion and tagwise dispersions. We see in the output below that of the top 1000 most DE tags as identified using tagwise dispersions, 87% of these tags are also in the list of the 1000 most DE tags as identified using the common dispersion. This shows that for this dataset there is a great deal of agreement between the common and tagwise dispersion approaches as to which tags are DE.

```
> sum(rownames(topTags(de.tgw, n = 1000)$table) %in% rownames(topTags(de.com,
+     n = 1000)$table))/1000 * 100
```

```
[1] 87.3
```

Using the common dispersion we found that 4835 genes (22% of the total number) are given an adjusted p-value of less than 0.05. In the output below, we see that using tagwise dispersions we obtain slightly more DE genes, namely 4933, or 23% of all of the genes in the dataset.

```
> sum(p.adjust(de.tgw$table$p.value, method = "BH") < 0.05)
```

```
[1] 4933
```

```
> mean(p.adjust(de.tgw$table$p.value, method = "BH") < 0.05) *
+     100
```

```
[1] 22.54880
```

Of the 4933 tags identified as DE using tagwise dispersions, 1981 (40%) are up-regulated in DHT-treated cells and 2952 (60%) are up-regulated in the control cells. The proportions of up- and down-regulated genes identified using the two approaches to modeling the dispersion are equal.

```
> top.tgw <- topTags(de.tgw, n = 4933)
> sum(top.tgw$table$logFC > 0)
```

```
[1] 1981
```

```
> sum(top.tgw$table$logFC < 0)
```

```
[1] 2952
```

6.5.3 Visualising DGE results

As discussed earlier, the function `plotSmear` can be used to generate a plot of the log-fold change against the log-concentration for each tag. We identify the top 500 most DE tags using both common dispersion and tagwise dispersions so we can highlight them on the plots and compare what we see. The code for producing the fold-change plots (in the one frame for purposes of comparison) is shown below, and the result of this code is shown in Figure 6.

```
> detags500.com <- rownames(topTags(de.com, n = 500)$table)
> detags500.tgw <- rownames(topTags(de.tgw, n = 500)$table)
```

```
> par(mfcol = c(2, 1))
> plotSmear(d, de.tags = detags500.com, main = "Using common dispersion")
> abline(h = c(-2, 2), col = "dodgerblue", lwd = 2)
> plotSmear(d, de.tags = detags500.tgw, main = "Using tagwise dispersions")
> abline(h = c(-2, 2), col = "dodgerblue", lwd = 2)
```

In Figure 6, the top 500 most differentially expressed genes (those identified as significant by `edgeR` using the common dispersion (top) and tagwise dispersions (bottom)) are highlighted in red. Looking at Figure 6, we see that, generally speaking, the pattern of differential expression looks similar using the two different methods, and the genes identified as DE have convincingly large fold changes.

6.6 Setup

This analysis of Li et al. [2008]'s RNA-seq data was conducted on:

```
> sessionInfo()
```

```
R version 2.9.1 (2009-06-26)
i386-apple-darwin8.11.1

locale:
en_AU.UTF-8/en_AU.UTF-8/C/C/en_AU.UTF-8/en_AU.UTF-8

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

other attached packages:
[1] edgeR_1.3.6

loaded via a namespace (and not attached):
[1] limma_2.18.2
```

and took 2–4 minutes to carry out on an Apple MacBook with a 2.4 Ghz Intel Core 2 Duo processor and 4 Gb of 1067 MHz DDR3 memory.

7 Poisson example

It has been noted that, in some deep sequencing approaches, not a great deal of overdispersion is observed. Specifically, the means and variances appear to be very close to each other, suggesting the Poisson distribution is a good fit. Methods within the `edgeR` package may still be useful, including the quantile adjustment (effectively a normalization) and the exact testing routines.

To illustrate this, we sample Poisson data and run `de4DGE` with the `doPoisson` argument set to `TRUE`. The data is quantile-adjusted and before the exact test is invoked, the dispersion parameter is set to 0. Currently, elements from the output of `de4DGE` must be manually added to the `DGEList` object, as `exactTest` operates only on `DGEList` objects, as illustrated in the case studies above.

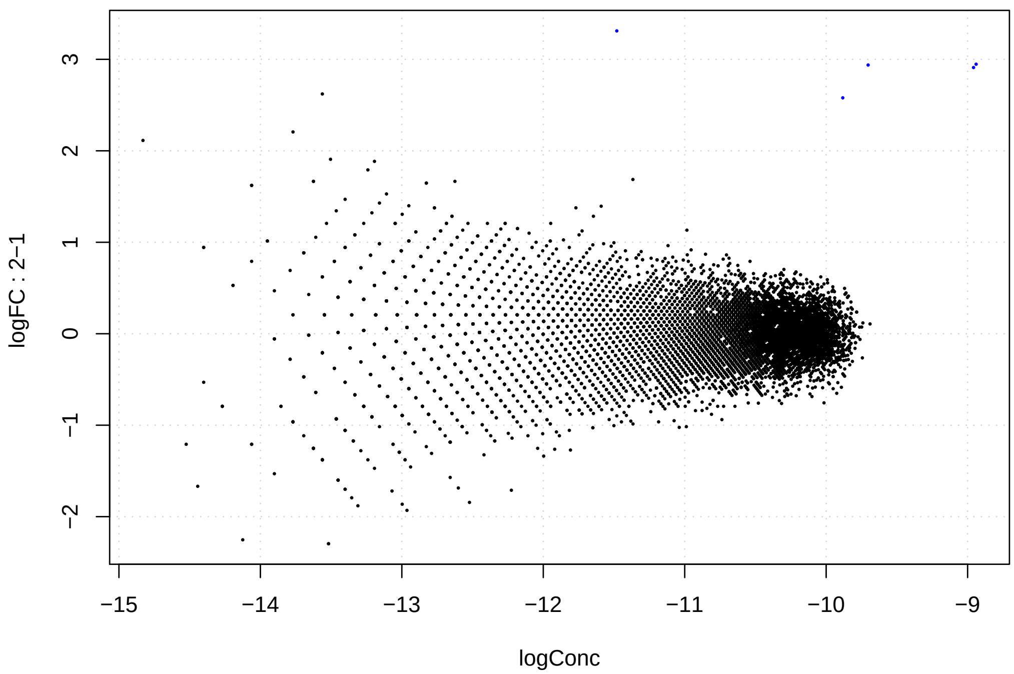
Nevertheless, an analysis using the Poisson distribution can be carried out as follows:

```
> set.seed(101)
> n <- 10000
> lib.sizes <- c(40000, 50000, 38000, 40000)
> p <- runif(n, min = 1e-04, 0.001)
> mu <- outer(p, lib.sizes)
> mu[1:5, 3:4] <- mu[1:5, 3:4] * 8
> y <- matrix(rpois(4 * n, lambda = mu), nrow = n)
> dP <- DGEList(counts = y, group = rep(1:2, each = 2), lib.size = lib.sizes)
> msP <- de4DGE(dP, doPoisson = TRUE)

Quantile adjusting as Poisson.

> dP$pseudo.alt <- msP$pseudo
> dP$common.dispersion <- 1e-06
> dP$conc <- msP$conc
> dP$common.lib.size <- msP$M
```

And you can proceed as before:

```
> de.P <- exactTest(dP)

Comparison of groups:  2 - 1

> topTags(de.P)

Comparison of groups:  2 - 1
        logConc     logFC       PValue          FDR
3     -8.939897  2.946691 3.076540e-81 3.076540e-77
4     -8.958118  2.910250 1.425021e-77 7.125106e-74
1     -9.703306  2.937857 1.635095e-47 5.450317e-44
5     -9.882605  2.579261 2.030845e-33 5.077113e-30
2    -11.480315  3.310787 6.058052e-18 1.211610e-14
```

```
9796 -11.366488  1.686444 3.397842e-06 5.663070e-03
2893 -10.984522  1.132451 3.740766e-04 4.471809e-01
142  -11.590674  1.394078 3.940818e-04 4.471809e-01
3541 -13.561931  2.621489 4.024628e-04 4.471809e-01
9783 -12.225718 -1.711087 6.061462e-04 6.061462e-01

> plotSmear(dP, col = c(rep("blue", 5), rep("black", n - 5)))
```



8 Future improvements and extension

Here, we list some improvements/extensions that are planned for the `edgeR` package:

1. As the packages for the processing of raw high-throughput sequencing data become more mature, `edgeR` may need to adapt and operate on different objects. As shown above, `edgeR` operates on a simple object containing simple data summaries which will presumably be readily available from pre-processing steps.

2. Significant speed improvements have been made since the earlier versions of `edgeR`, but as the datasets become larger, some further optimizations may be necessary. We are exploring various ways to do this.

3. The current quantile-based normalization assumes the library sizes are fixed. Depending on the circumstances of the samples in question, it may be necessary to explore something like an *effective* library size.

4. We are exploring more general testing procedures.

9 Setup

This vignette was built on:

```
> sessionInfo()

R version 2.10.0 Under development (unstable) (2009-05-22 r48594)
i386-apple-darwin8.11.1

locale:
[1] en_CA.UTF-8/en_CA.UTF-8/C/C/en_CA.UTF-8/en_CA.UTF-8

attached base packages:
[1] tools     stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
[1] edgeR_1.3.7

loaded via a namespace (and not attached):
[1] limma_3.0.0
```

References

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B*, 57:289–300, 1995.

H. R Li, J. Wang-Rodriguez, T. M Nair, J. M Yeakley, Y. S Kwon, M. Bibikova, C. Zheng, L. Zhou, K. Zhang, and T. Downs. Two-dimensional transcriptome profiling: identification of messenger rna isoform signatures in prostate cancer from archived paraffin-embedded cancer specimens. *Cancer Research*, 66(8):4079–4088, 2006.

H. R Li, M. T Lovci, Y-S. Kwon, M. G Rosenfeld, X-D. Fua, and G. W Yeo. Determination of tag density required for digital transcriptome analysis: Application to an androgen-sensitive prostate cancer model. *Proceedings of the National Academy of Sciences of the USA*, 105(51): 20179–20184, 2008.

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

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

P. A. C 't Hoen, Y. Ariyurek, H. H Thygesen, E. Vreugdenhil, R. H. A. M Vossen, R. X De Menezes, J. M Boer, G-J. B Van Ommen, and J. T Den Dunnen. Deep sequencing-based expression analysis shows major advances in robustness, resolution and inter-lab portability over five microarray platforms. *Nucleic Acids Research*, 36(21):e141, 2008.

L. Zhang, W. Zhou, V. E Velculescu, S. E Kern, R. H Hruban, S. R Hamilton, B. Vogelstein, and K. W Kinzler. Gene expression profiles in normal and cancer cells. *Science*, 276(5316): 1268–1272, May 1997.

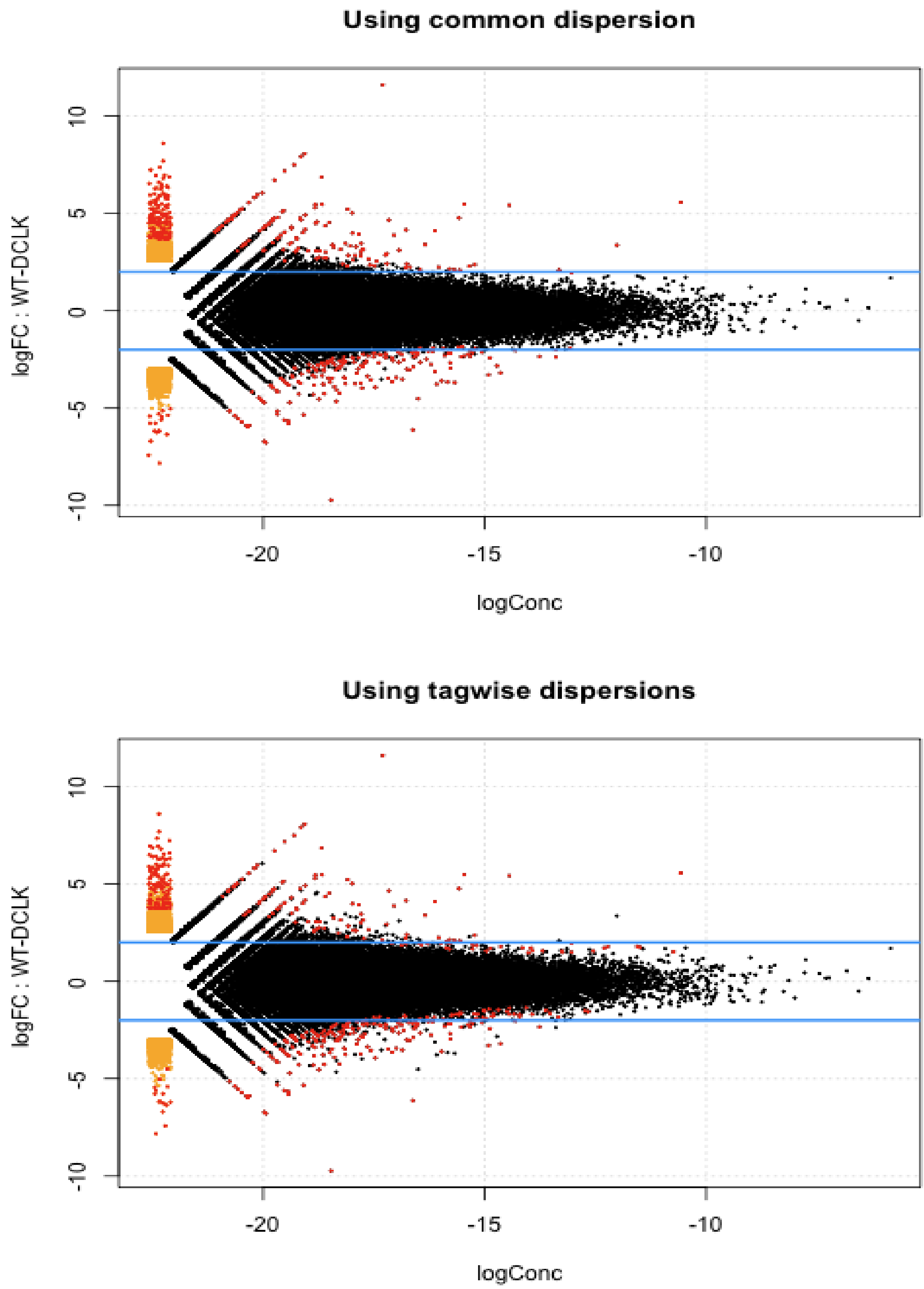


Figure 4: Plots of the log-fold change against the log-concentration for each tag, using the common dispersion (upper), and tagwise dispersions (lower). Tags with positive fold-change here are up-regulated in wild-type compared with transgenic mice. The 500 most differentially expressed tags according to each method are highlighted in red on both plots.

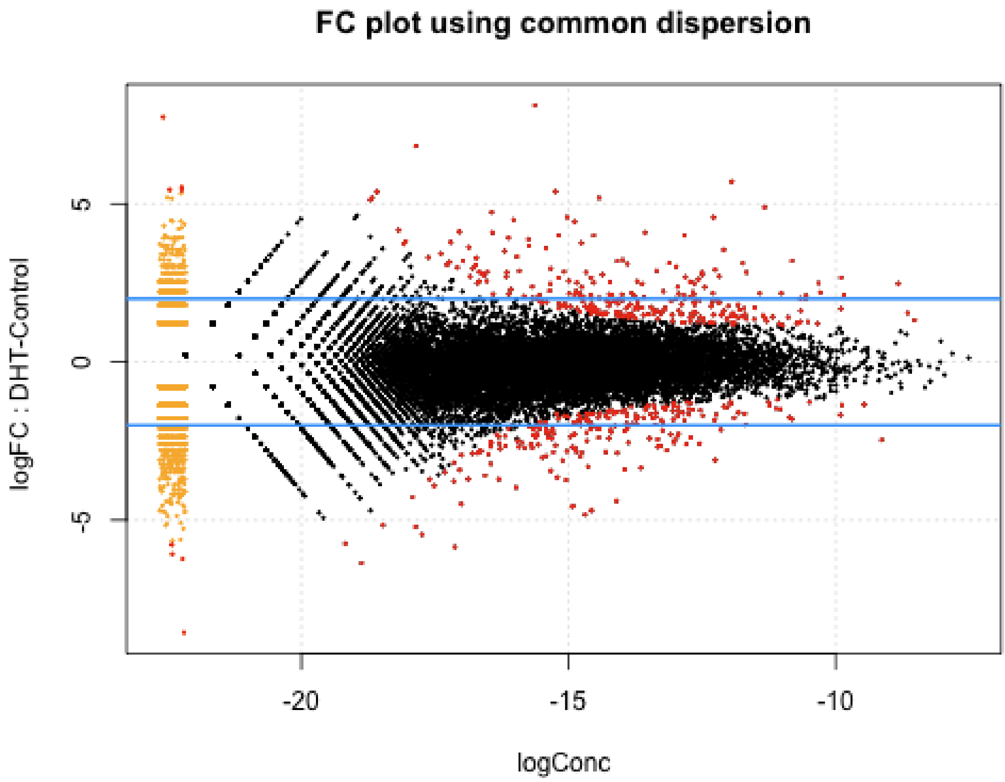


Figure 5: Plot of the log-fold change against the log-concentration for each tag. The 500 most differentially expressed tags as identified by `edgeR` using the common dispersion are outlined in red.

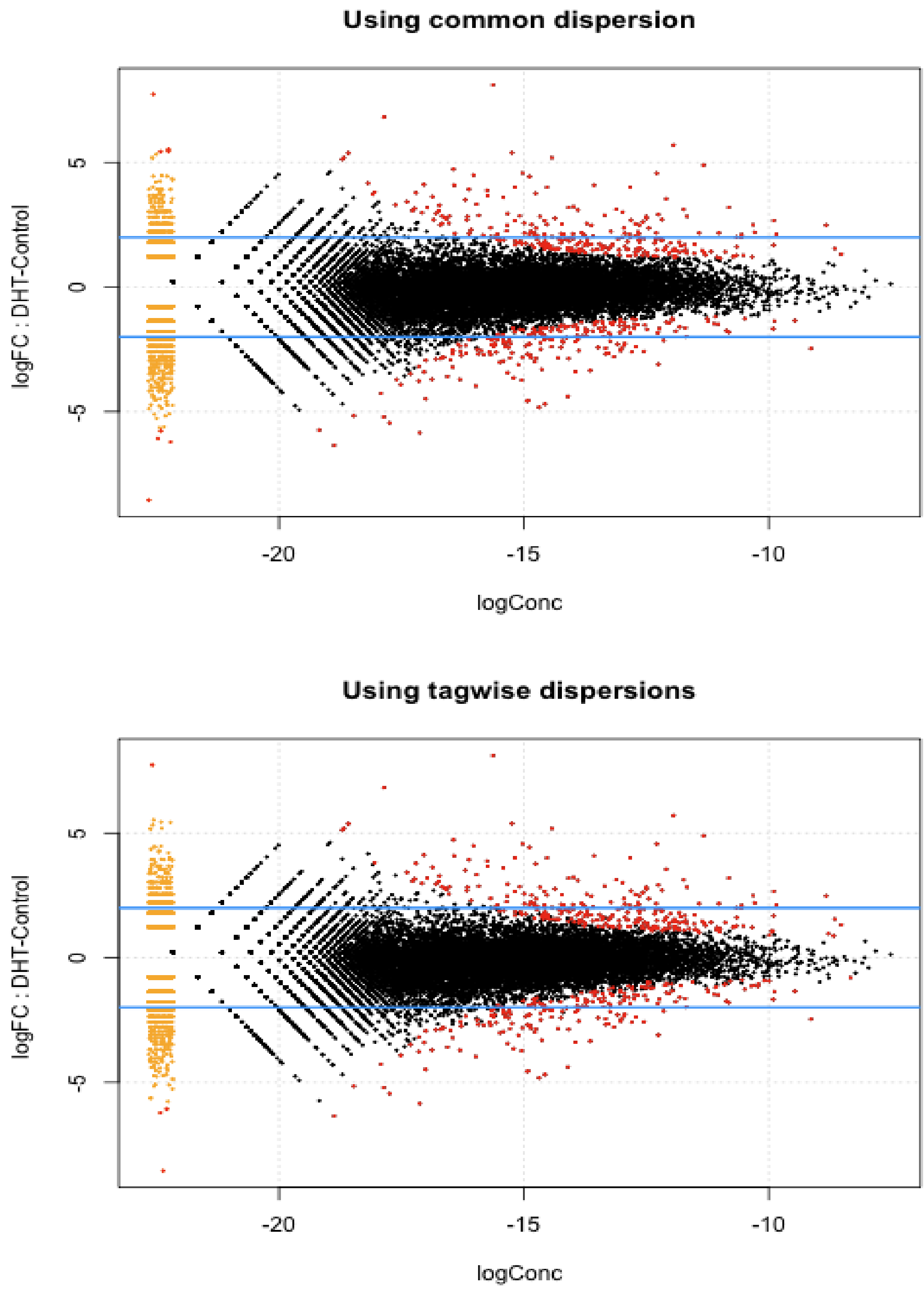


Figure 6: Plots of the log-fold change against the log-concentration for each tag, using the common dispersion (top), and tagwise dispersions (bottom). Tags with positive fold-change here are up-regulated in DHT-treated cells compared with control cells. The 500 most differentially expressed tags according to each method are highlighted in red on both plots.