UHasselt

2nd Master Bioinformatics

Advanced Methods for Genomics

Differential Expression Analysis for RNA Sequencing

Author:
Pieter Moris

Supervisor:
Dr. Jurgen Claesen

1 April 2016

universiteit hasselt
KNOWLEDGE IN ACTION

Contents

1 Introduction

The search for genes that are differentially expressed between different conditions (gene expression profiling) plays a key role in elucidating the underlying molecular mechanisms of diseases, treatments or biological variation in general. Traditionally micro-arrays were the most prominent tool to conduct these transcriptome analyses, but due to advances in next-generation sequencing methods, RNA-sequencing has emerged as an alternative (Soneson and Delorenzi 2013). Rather than comparing samples to a pre-specified, known library of gene probes, RNA-seq tries to sequence all the mRNA transcripts that are present in a specific tissue (or even cell) at a certain time point. First, the mRNA is isolated from the sample and reverse transcribed into cDNA. Then, next-generation sequencing is used to generate millions of short reads, which are then mapped to a reference genome. In the end, the total number of reads that map to each specific gene (or other genomic region of interest, e.g. exons) is used as a proxy for the expression level. Many different methods have been proposed to analyse these counts and they differ in various aspects, such as their strategies for normalisation and significance testing (Oshlack, Robinson, and Young 2010).

In this report we compare four different approaches for the analysis of differential expression for RNA-seq: a parametric, a non-parametric, a transformation and a Bayesian method. These methods will be described in more detail in the methodology section of this report. We will analyse a dataset with gene counts for six samples, three of which are controls, while the other three were obtained from patients who received a novel treatment for squamous cell carcinoma, a form of skin cancer.

Note that the starting point of this study is already a dataset with gene counts, ready for differential expression analysis. All of the earlier steps of the RNA-seq workflow, i.e. library construction, base-calling, mapping of the reads and conversion to gene counts, were already performed. However, we have to keep in mind that each of those steps came with its own set of potential pitfalls and that all our down-stream analyses will depend on the validity of the earlier methods. For example, we do not know for sure how non-unique reads were dealt with (discarded, counted for each possible gene, etc.) and this can, at least in theory, have a large effect on the final gene counts (Oshlack, Robinson, and Young 2010). Another issue that can occur during the mapping phase is the difficulty in mapping reads that span exon junctions: using a genomic reference will tend to give greater coverage of transcripts with fewer exon junctions (at the same expression levels), but this can be dealt with in multiple ways, each with their own benefits and drawbacks.

Lastly, we are only interested in the performance of these methods to detect differential expression between samples. We will not perform any follow-up analyses, such as gene ontology or enrichment analyses.

2 Methodology

2.1 Parametric approach - edgeR and the negative binomial model

The first method we employed was the parametric approach of the edgeR package by Robinson, McCarthy, and Smyth (2010). There are two important concepts to discuss: normalisation of the count data and testing for differentially expressed genes. Since these aspects will also be important for the other methods, we will devote more time to them here the first time.

normalisation is necessary to compensate for biases and to allow for accurate comparisons. There are a number of biases inherent to the next-generation sequencing methods used for RNA-seq. For example, coverage is not uniform across the genome (e.g. due to GC-bias) and mapping of reads is positively correlated with gene length (Soneson and Delorenzi 2013). Fortunately, these within-sample biases can be ignored in the case of differential experiments, because we can assume that they affect all samples in the same manner and will cancel out (Chen et al. 2015). In other words, only relative changes between samples are of interest, not the absolute quantification of expression levels between genes within a sample.

Between-sample biases on the other hand do need to be accounted for. The most important one is probably the sequencing depth or library size, which differs between samples. A larger

library size inherently leads to higher gene counts at the same expression level, so it makes intuitive sense to scale the counts according to the library size (Soneson and Delorenzi 2013). However, this does not yet account for the RNA composition effect; very high expression of a small number of genes can lead to under-sampling of the other genes and give the false impression of down-regulation (Chen et al. 2015). EdgeR uses a model-based normalisation, the TMM normalisation (trimmed mean of M-values), which assumes that most genes are in fact not differentially expressed and consequently should have similar counts. The raw data is used to estimate scaling factors that minimize the log-fold changes between samples for most genes (Robinson and Oshlack 2010). The product of the library sizes and the scaling factors is the effective library size that will be used in the next steps of the analysis. An advantage of the scaling factor method is that it preserves the raw counts and as such can be used for count-based parametric models (Oshlack, Robinson, and Young 2010). Moreover, it has been shown to perform adequately on simulation data compared to other methods such as RPKM (reads per kilobase per million mapped reads) normalisation (Dillies et al. 2013).

Prior to the normalisation, we filtered out genes with a count per million lower than one and which were expressed in fewer than three samples (because there are three replicates per condition). This cut-off is ad-hoc, but recommended by Chen et al. (2015) since transcripts need to reach a certain level of abundance before they are translated into proteins with a meaningful biological effect.

For the differential expression test, edgeR fits a negative binomial (NB) model to the count data (since we are dealing with a discrete distribution). This is appropriate for biological replicates, since here the variance is often higher than expected under the assumptions of a more simple Poisson model (Robinson, McCarthy, and Smyth 2010). The NB model has two parameters that need to be estimated, the mean and the dispersion, the latter of which can allow for overdispersion (unlike the simpler Poisson model). The dispersion is first estimated as a common dispersion factor, which assumes that all genes have the same mean-variance relation. Since we have a simple one factor experiment, this can be achieved with the quantile-adjusted conditional maximum likelihood (qCML) method (Robinson, McCarthy, and Smyth 2010). This method naturally lends itself to RNA-seq because it performs best when there are many small samples (genes) with a common dispersion (Robinson and Smyth 2008). Then, this estimate is improved by using an empirical Bayes method to shrink unique dispersion estimates of each gene (tagwise dispersions) to the common dispersion (Robinson and Smyth 2007). Intuitively this can be thought of as borrowing information between genes to estimate the variance more accurately (Robinson, McCarthy, and Smyth 2010). This gives a more accurate estimation of the genewise dispersion parameters, which is otherwise difficult to achieve due to small samples sizes (Soneson and Delorenzi 2013). The amount of shrinkage is based on the dispersion trend (a prior weight) (Chen, Lun, and Smyth 2014). Note that for more complex experiments there exists an extension of this framework, which is based on generalized linear models, but the overall idea is the same.

After estimating the gene-specific biological variation and fitting the NB model, we can test for differential expression using the exact test for the negative binomial distribution (for experiments with a single factor) which directly calculates p-values (Chen et al. 2015; Robinson and Smyth 2008). This test bears some resemblance to Fisher's exact test. It is a pairwise testing procedure, but since we only have two groups, this is not an issue. We opted for the deviance goodness-of-fit statistic to define the rejection region (the conditional likelihood ratio test), which has good theoretical properties but is slightly slower, compared to the alternative options. We used the Benjamini-Hochberg false discovery rate (FDR) to protect against false positives after multiple testing; a 5% FDR corresponds to fewer than 5% false positives among all rejected hypotheses.

Finally, edgeR also offers an exploratory method to visualise the differences between the samples, namely a multi-dimensional scaling plot. This is an unsupervised clustering approach where the log-fold change (i.e. the differences in expression) between each pair of samples is used as the distance measure (Chen et al. 2015).

2.2 Non-parametric approach - SAMSeq

For our non-parametric approach, we settled on SAMSeq, which uses Wilcoxon ranks and re-sampling to account for differences in sequencing depth (Li and Tibshirani 2013). Unlike the previous parametric approach, this rank-based strategy does not impose any distributional assumptions on the data. This is an advantage since parametric methods might break down when there is too much deviation from the proposed assumptions. Moreover, this method is also easier extended to more complex outcomes, such as survival, because the underlying models are less complex and not bound to a distribution.

For our two-group experiment the Wilcoxon rank statistic is first computed for each gene. Then, Poisson re-sampling is used, repeatedly, to account for differences in sequencing depth (the reason for which is the same as before). Note that due to this resampling strategy, no prior normalisation is required. Repeated re-sampling is required to increase the power of the Wilcoxon statistic and to alleviate the risks of missing data due to the random sampling process.

Subsequently, a permutation method computes the FDR cut-off value for the obtained test statistics; permutations of the original data are generated to approximate the null distribution of the test statistic. Lastly, a q-value is reported for each gene, which reports significance in terms of the FDR (Storey and Tibshirani 2003).

Unfortunately, this method is known to have poor power when there are fewer than five replicate samples per condition (Soneson and Delorenzi 2013; Seyednasrollah, Laiho, and Elo 2015).

We performed the SAMSeq analysis using both a 5% FDR and the default 20% FDR setting, on both the filtered and unfiltered dataset (see previous section). The number of permutations for the FDR estimation was set to 500 and the number of resamples to construct the test statistic was 100.

2.3 Transformation approach - voom and limma

Transformation-based methods aim to transform the gene counts in such a way that they become appropriate for the traditional differential expression tests employed for microarray analyses (which have log normally distributed outcomes) (Soneson and Delorenzi 2013). The voom method (variance modeling at the observational level) implemented in the limma package (Ritchie et al. 2015; Law et al. 2014) achieves this by converting the counts to the log-scale (log counts per million) and then estimating the mean-variance relationship empirically. Based on the mean-variance relation, weights are computed for each observation and fed into the standard linear models and empirical bayes methods of limma. Once again the idea is to borrow information between genes to estimate the gene-specific variation more accurately, despite small sample sizes (like in the empirical Bayes methods in edgeR). In a sense it is an extension of the familiar t-test, but the standard errors are adjusted.

As is recommended by the authors of the method, we again first normalised the gene counts using the TMM method, as implemented by edgeR, to account for varying library sizes (Smyth et al. 2015). We operated on the filtered dataset and the FDR was kept at 5% once again.

2.4 Bayesian approach - EBSeq

Lastly, we used a Bayesian approach, namely EBSeq (Leng et al. 2013). This method has some similarities to the parametric edgeR method described above, in the sense that both rely on an underlying negative binomial model. The difference lies in the statistical inference, which is entirely Bayesian for EBSeq. A Bayesian method is employed to calculate the posterior probabilities of differential expression between the two conditions, whilst using a negative binomial model as a prior. In the end, a Bayesian FDR is supplied.

Once again we worked with the filtered dataset. For normalisation we used the median normalisation (based on DESeq) as was recommended by the authors of the method (Leng, Dawson, and Kendziorski 2015). Similar to the TMM normalisation, this results in size factors that correct for the differences in library size. The number of iterations was evaluated and seven seemed to result in convergence since the parameter estimates no longer changed substantially after this point.

3 Results

3.1 Parametric - EdgeR results

Despite our reasonably mild choices, the filtering step reduced the number of tags almost by half (14599 v.s. 7875). The exploratory multi-dimensional scaling plot that projects the similarity of the samples into a two-dimensional scatter plot is provided in figure 1.

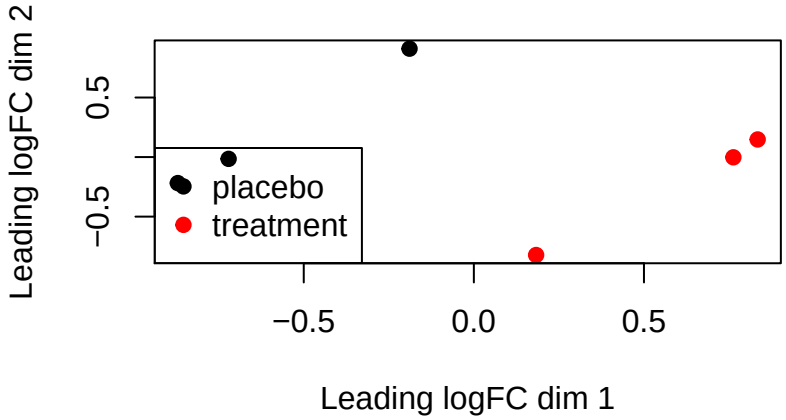

Figure 1: Multi-dimensional scaling plot visualising the distance (log-fold expression) between the samples.

The estimation of the common and gene-specific dispersion factors is shown in figure 2. Note that the biological coefficient of variation is plotted, rather than the actual dispersion.

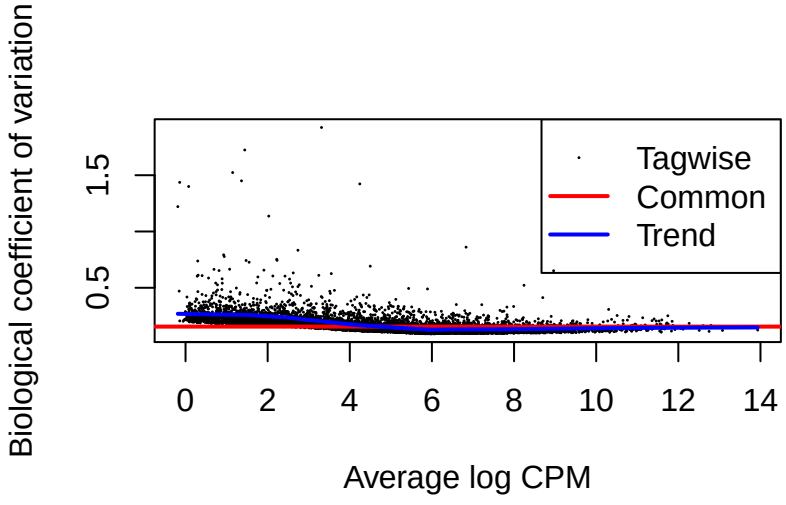

Figure 2: Plot of the biological coefficient of variations (or the square root of the dispersion parameter of the NB model) for the common and tagwise method (trended method not relevant here). CPM = counts per million.

The top ten significant genes as found by the exact test are provided in table 1. In total there are 704 genes significantly up- (372) or down- (332) regulated after controlling the FDR

at 5%. These genes are depicted in figure 3 as the red dots.

Table 1: Top ten differentially expressed genes based on the edgeR exact test.

Genes	logFC	PValue	FDR
9831	-4.60	0.00	0.00
2366	2.92	0.00	0.00
612	-3.21	0.00	0.00
3192	-2.20	0.00	0.00
6948	-2.52	0.00	0.00
10305	-4.11	0.00	0.00
6711	-3.50	0.00	0.00
3411	-2.45	0.00	0.00
30	2.78	0.00	0.00
6495	-3.55	0.00	0.00

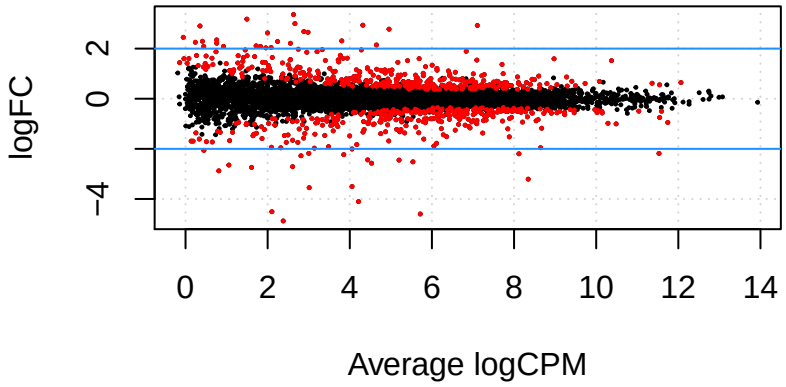

Figure 3: Plot of of the log-fold change against log-counts per million for edgeR. Significant genes (FDR 5%) are highlighted in red. The blue lines indicate a log-fold change of two.

3.2 Non-parametric - SAMSeq results

We ran into a number of issues while performing the SAMSeq analysis. First, a 5% FDR resulted in a set of significant genes that existed exclusively of up-regulated (filtered gene set) or down-regulated (unfiltered set) genes. After increasing the FDR to 20%, this issue was resolved, but at the cost of many more false positives.

Second, the four different options (FDR 5/20% and an (un-)filtered dataset) resulted in extremely heterogeneous results (figure 4). Only thirty-one significant genes were shared between the analyses. Even for the same FDR value, there were large discrepancies between the filtered and unfiltered results. At 20% FDR, 64 of the filtered genes were still found to be significant (5 at 5% FDR), while we would expect none of them to be significant due to the low counts.

These large inconsistencies indicate that the SAMSeq method is not reliable for our dataset. We will discuss possible causes and solutions for these problems in the discussion section.

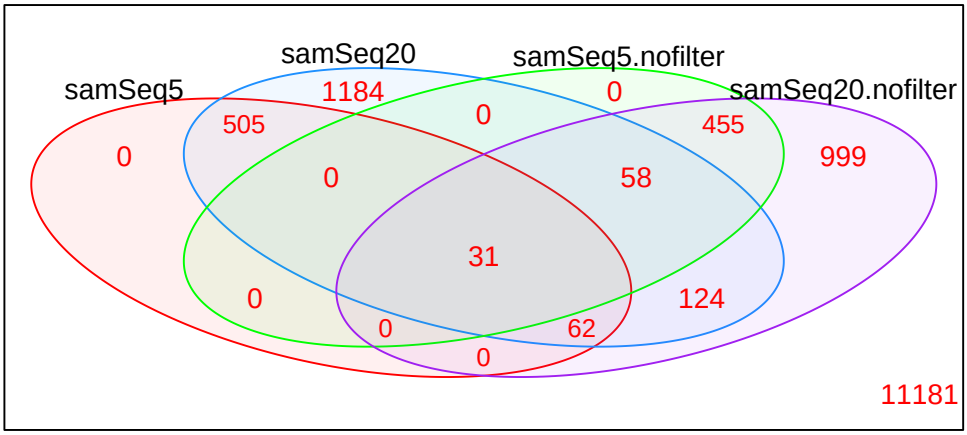

Figure 4: Venn diagram of the number of significant differentially expressed genes found by SAMSeq for an FDR of 5% or 20 % and using the entire or filtered dataset.

Table 2: Top ten differentially expressed genes based on SAMSeq at 5% (left) and 20% FDR (right) respectively.

Gene Name	Fold Change	q-value(%)	Gene Name	Fold Change	q-value(%)
7	1.683	7.074	809	1.196	10.216
15	1.309	7.074	2400	1.4	10.216
16	6.222	7.074	4036	2.005	10.216
29	2.613	7.074	6225	1.192	10.216
35	1.426	7.074	6410	1.08	10.216
37	1.287	7.074	979	1.366	10.437
71	1.561	7.074	2679	1.156	10.437
91	1.362	7.074	4483	1.215	10.437
95	1.517	7.074	5325	1.279	10.437
103	1.536	7.074	53	0.782	10.653

For completeness, the top ten genes as reported by SAMSeq (after filtering), for 5 and 20% FDR, are provided in table 2. Note that there were only 1 and 1964 unique q-values respectively, so the top ten genes are actually not that informative; all of the remaining genes are supposedly equally significant (in the case of 5% FDR).

3.3 Transformation - voom and limma results

The mean-variance relation obtained by voom is shown in figure 5. The top ten significant genes subsequently found by limma can be found in table 3. In total there are 455 genes significantly up- (245) or down- (210) regulated when controlling the FDR at 5%. A visual representation, similar to the one for EdgeR, is provided in figure 6.

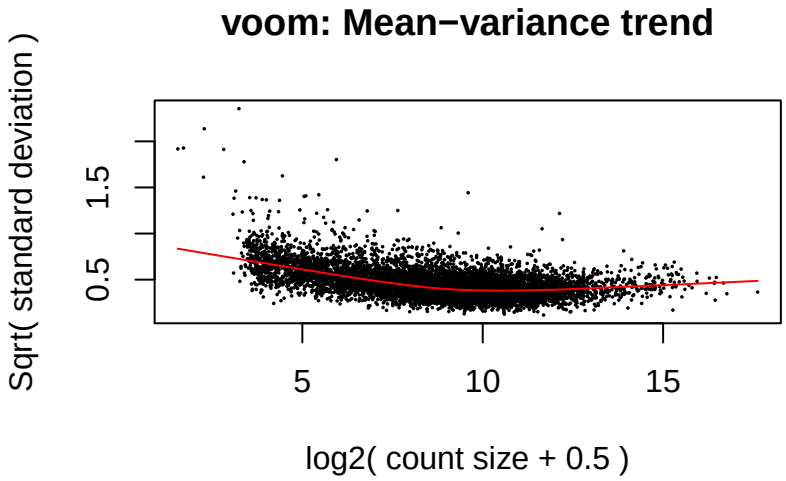

Figure 5: Mean-variance relation estimated by voom.

Table 3: Top ten differentially expressed genes based on voom+limma.

Genes	logFC	P.Value	FDR
2366	2.94	0.00	0.00
612	-3.17	0.00	0.00
9831	-4.61	0.00	0.00
3192	-2.20	0.00	0.00
6948	-2.53	0.00	0.00
3411	-2.42	0.00	0.00
30	2.79	0.00	0.00
11422	2.29	0.00	0.00
10305	-4.11	0.00	0.00
6711	-3.47	0.00	0.00

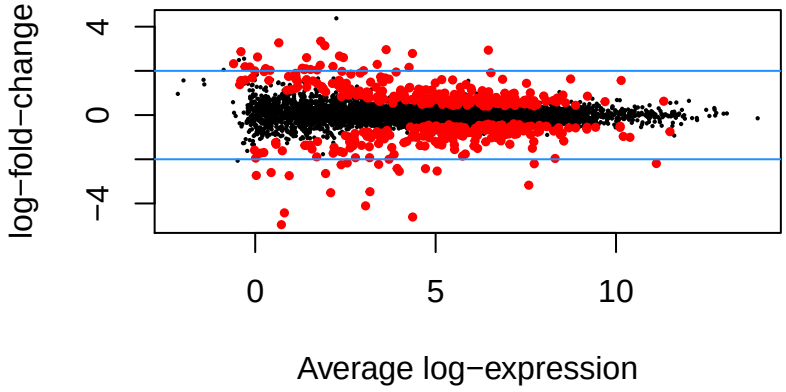

Figure 6: Plot of the log-fold change against log-counts per million for limma. Significant genes (FDR 5%) are highlighted in red.

3.4 Bayesian - EBSeq

In total there were 423 genes that were found to be significantly up- or down-regulated when controlling the Bayesian FDR at 5%. Almost all of these (412) had a posterior probability of one, so note that table 4 is arbritrary in this sense, since the genes are merely sorted by their identifier.

Table 4: Top ten differentially expressed genes based on EBSeq and their posterior probability of being differentially expressed.

Genes	Posterior P. DE
8214	0.00
7617	0.00
831	0.00
10004	0.00
4508	0.00
11741	0.00
6240	0.00
1141	0.00
5697	0.00
6870	0.00

3.5 Agreement and differences between the methods

The following Venn diagram depicts the overlap among the set of differentially expressed genes that were reported as significant by the four methods (figure 7). Because the SAMSeq method was performing poorly and resulted in extremely different results, we also created an overlap plot for only edgeR, voom+limma and EBSeq (figure 8).

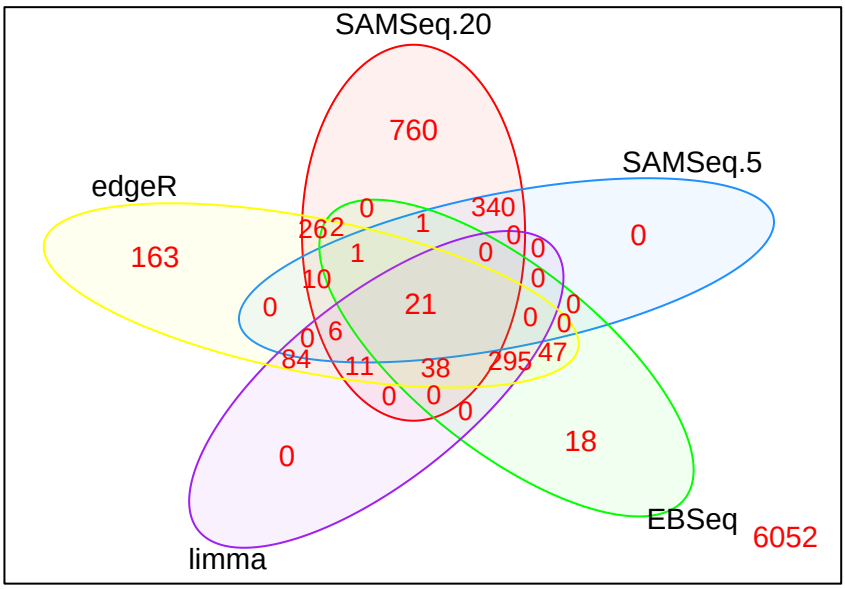

Figure 7: The number of significantly differentially expressed genes (either up- or down-regulated) for edgeR, voom+limma, EBSeq (FDR 5%) and SAMSeq (FDR 5 or 20%).

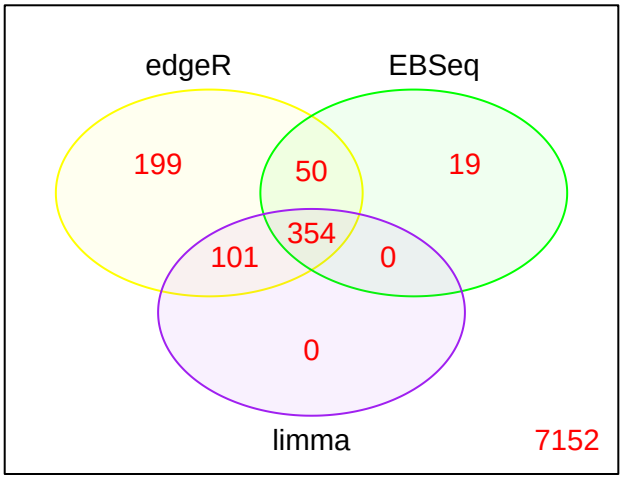

Figure 8: The number of significantly differentially expressed genes (either up- or down-regulated) at FDR 5% for edgeR, voom+limma and EBSeq.

4 Discussion

The multidimensional scaling plot in figure 1 indicates that the samples are reasonably well separated, although it seems that each group has one observation which behaves slightly differently (separation according to second axis).

Overall, there is quite a high degree of agreement between the parametric, Bayesian and transformation-based approaches (figure 8). This strengthens our confidence that the differentially expressed genes reported by these methods truly are so. We note that edgeR found 199 unique genes, whereas EBSeq only found 19 and limma did not find any. This might simply reflect the fact that edgeR reported almost half as many genes (704) as EBSeq (423) and limma (455). In the comparison by Soneson and Delorenzi (2013) edgeR proved to be too liberal for small sample sizes as well. This is also, at least partly, consistent with the results of Seyednasrollah, Laiho, and Elo (2015): in their simulation studies both edgeR and EBSeq proved to be too liberal and reported many more false positives than limma.

Looking at the top set of genes reported by edgeR (table 1) and limma (table 3), we see the same ones popping up. For EBSeq the list is a bit different (table 4), but this is an artefact of the internal sorting of the method, since a large portion of genes shared the same posterior probability.

The non-parametric SAMSeq method did not perform adequately in this analysis, but this was to be expected since it has poor power for small sample sizes (Seyednasrollah, Laiho, and Elo 2015; Soneson and Delorenzi 2013). However, it was still striking to observe that at a reasonable FDR cut-off of 5%, the method reported only up- or down-regulated genes, but never both. This issue has since been addressed by one of the authors, in the form of another package npSeq (Li 2011). It implements the same methods that were described in the original SAMSeq paper (Li and Tibshirani 2013), but with symmetric, rather than asymmetric cut-offs for the non-parametric statistic. This could avoid the problem of exclusively finding up- or down-regulated genes, and instead find both simultaneously. Unfortunately this package was only available for Linux platforms at the time of writing, so we did not try it out. Moreover, the fact that filtering had such a drastic effect on the called genes and that the overall agreement between different FDR cut-off values was so low, also shows how unstable this method was for our analysis (figure 4). Lastly, we want to address our arbitrary choice of number of permutations and resampling

rounds. We did not find any recommendations, apart from the default values implemented in the package. We increased these to be on the safe side, but a safer method would have been to slowly increase the number of simulations and evaluate when the results started to converge.

The dangers of a low number of samples are not merely limited to the non-parametric method and we should be cautious for the other methods too. It is interesting that limma performs well in our dataset, despite the fact that it also suffers from low power for small sample sizes (Soneson and Delorenzi 2013). The tendency of edgeR to be too liberal also decreases with growing sample size.

In general, all the methods become more similar and stable for increasing sample sizes (Seyednasrollah, Laiho, and Elo 2015). However, the optimal method still depends on the experimental conditions (heterogeneity of the sample, outliers, sample size) (Soneson and Delorenzi 2013). The choice of method will also depend on the complexity of the experimental design. In this regard, the non-parametric SAMSeq method has an advantage, but limma and edgeR also offer support for this. Lastly, the final choice of method can in general be guided by practical issues as well, such as the availability of in-depth documentation and case-studies.

References

[1] Yunshun Chen, Aaron T. L. Lun, and Gordon K. Smyth. "Statistical Analysis of Next Generation Sequencing Data". In: ed. by Somnath Datta and Dan Nettleton. Cham: Springer International Publishing, 2014, pp. 51–74. URL: http://dx.doi.org/10.1007/978-3-319-07212-8₃.

[2] Yunshun Chen et al. *edgeR: differential expression analysis of digital gene expression data: User's Guide*. Aug. 10, 2015. URL: https://bioconductor.org/packages/release/bioc/html/edgeR.html (visited on 09/03/2016).

[3] M.-A. Dillies et al. "A comprehensive evaluation of normalization methods for Illumina high-throughput RNA sequencing data analysis". In: *Briefings in Bioinformatics* 14.6 (Nov. 1, 2013), pp. 671–683. DOI: 10.1093/bib/bbs046. URL: http://bib.oxfordjournals.org/cgi/doi/10.1093/bib/bbs046 (visited on 03/09/2016).

[4] Charity W Law et al. "voom: precision weights unlock linear model analysis tools for RNA-seq read counts". In: *Genome Biology* 15.2 (2014), R29. DOI: 10.1186/gb-2014-15-2-r29. URL: http://genomebiology.com/2014/15/2/R29 (visited on 03/12/2016).

[5] N. Leng et al. "EBSeq: an empirical Bayes hierarchical model for inference in RNA-seq experiments". In: *Bioinformatics* 29.8 (Apr. 15, 2013), pp. 1035–1043. DOI: 10.1093/bioinformatics/btt087. URL: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btt087 (visited on 03/13/2016).

[6] Ning Leng, John Dawson, and Christina Kendziorski. *EBSeq: An R package for differential expression analysis using RNA-seq data*. Aug. 12, 2015. URL: https://www.bioconductor.org/packages/devel/bioc/vignettes/EBSeq/inst/doc/EBSeq_Vignette.pdf (visited on 03/12/2016).

[7] J. Li and R. Tibshirani. "Finding consistent patterns: A nonparametric approach for identifying differential expression in RNA-Seq data". In: *Statistical Methods in Medical Research* 22.5 (Oct. 1, 2013), pp. 519–536. DOI: 10.1177/0962280211428386. URL: http://smm.sagepub.com/cgi/doi/10.1177/0962280211428386 (visited on 03/10/2016).

[8] Jun Li. *Using npSeq (Version 1.1) to discover differential expression based on sequencing data*. 2011. URL: https://www3.nd.edu/~jli9/npSeq/npSeq_instructions.pdf (visited on 11/03/2016).

[9] Alicia Oshlack, Mark D Robinson, and Matthew D Young. "From RNA-seq reads to differential expression results". In: *Genome Biology* 11.12 (2010), p. 220. DOI: 10.1186/gb-2010-11-12-220. URL: http://genomebiology.com/2010/11/12/220 (visited on 03/09/2016).

[10] M. E. Ritchie et al. "limma powers differential expression analyses for RNA-sequencing and microarray studies". In: *Nucleic Acids Research* 43.7 (Apr. 20, 2015), e47–e47. DOI: 10.1093/nar/gkv007. URL: http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gkv007 (visited on 03/12/2016).

[11] M. D. Robinson, D. J. McCarthy, and G. K. Smyth. "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data". In: *Bioinformatics* 26.1 (Jan. 1, 2010), pp. 139–140. DOI: 10.1093/bioinformatics/btp616. URL: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btp616 (visited on 03/07/2016).

[12] M. D. Robinson and G. K. Smyth. "Moderated statistical tests for assessing differences in tag abundance". In: *Bioinformatics* 23.21 (Nov. 1, 2007), pp. 2881–2887. DOI: 10.1093/bioinformatics/btm453. URL: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btm453 (visited on 03/09/2016).

[13] Mark D Robinson and Alicia Oshlack. "A scaling normalization method for differential expression analysis of RNA-seq data". In: *Genome Biology* 11.3 (2010), R25. DOI: 10.1186/gb-2010-11-3-r25. URL: http://genomebiology.com/2010/11/3/R25 (visited on 03/09/2016).

[14] Mark D. Robinson and Gordon K. Smyth. "Small-sample estimation of negative binomial dispersion, with applications to SAGE data". In: *Biostatistics (Oxford, England)* 9.2 (Apr. 2008), pp. 321–332. DOI: 10.1093/biostatistics/kxm030. pmid: 17728317.

[15] Sean Ruddy. *edgeR Tutorial: Differential Expression in RNA-Seq Data*. edgeR Tutorial: Differential Expression in RNA-Seq Data. Sept. 26, 2011. URL: https://cgrlucb.wikispaces.com/file/view/edgeR_Tutorial.pdf (visited on 08/03/2016).

[16] F. Seyednasrollah, A. Laiho, and L. L. Elo. "Comparison of software packages for detecting differential expression in RNA-seq studies". In: *Briefings in Bioinformatics* 16.1 (Jan. 1, 2015), pp. 59–70. DOI: 10.1093/bib/bbt086. URL: http://bib.oxfordjournals.org/cgi/doi/10.1093/bib/bbt086 (visited on 03/10/2016).

[17] Gordon K. Smyth et al. *limma: Linear Models for Microarray and RNA-Seq Data User's Guide*. Aug. 9, 2015. URL: https://www.bioconductor.org/packages/release/bioc/vignettes/limma/inst/doc/usersguide.pdf (visited on 12/03/2016).

[18] Charlotte Soneson and Mauro Delorenzi. "A comparison of methods for differential expression analysis of RNA-seq data". In: *BMC Bioinformatics* 14.1 (2013), p. 91. DOI: 10.1186/1471-2105-14-91. URL: http://www.biomedcentral.com/1471-2105/14/91 (visited on 03/08/2016).

[19] John D. Storey and Robert Tibshirani. "Statistical significance for genomewide studies". In: *Proceedings of the National Academy of Sciences of the United States of America* 100.16 (Aug. 5, 2003), pp. 9440–9445. DOI: 10.1073/pnas.1530509100. pmid: 12883005.

A Source code

```
# source('http://bioconductor.org/biocLite.R')
# biocLite('edgeR')
library(readxl)
DEdata <- read_excel("dataset.xlsx")
DEdata <- DEdata[, -1]
groupLabels <- c(rep("placebo", 3), rep("treatment",
    3))
################################################## edgeR
library(edgeR)
DEcounts <- DGEList(counts = DEdata, group = factor(groupLabels))
DEcounts.original <- DEcounts
# filtering relative to library size
keep <- rowSums(cpm(DEcounts) > 1) >= 3  # 1 read per million for at least 3 samples
DEcounts <- DEcounts[keep, , keep.lib.sizes = FALSE]
DEcounts$samples$lib.size <- colSums(DEcounts$counts)  # recompute library sizes
# How many tags were removed?
ntags.filtered <- dim(DEcounts)[1]
ntags.original <- dim(DEcounts.original)[1]
# trimmed mean of M-values (TMM)
# normalisation for sample
# specific-effects - MODEL BASED not a
# transformation!
DEcounts <- calcNormFactors(DEcounts, method = "TMM")
# smear plot before and after correction
par(mfrow = c(1, 2))
maPlot(DEcounts$counts[, 1], DEcounts$counts[,
    4], normalise = TRUE, pch = 19, cex = 0.4,
    ylim = c(-8, 8))
grid(col = "blue")
abline(h = log2(DEcounts$samples$norm.factors[2]/DEcounts$samples$norm.factors[1]),
```

```
    col = "red", lwd = 4)
eff.libsize <- DEcounts$samples$lib.size *
    DEcounts$samples$norm.factors
maPlot(DEcounts$counts[, 1]/eff.libsize[1],
    DEcounts$counts[, 2]/eff.libsize[2],
    normalise = FALSE, pch = 19, cex = 0.4,
    ylim = c(-8, 8))
grid(col = "blue")
par(mfrow = c(1, 1))
plotMDS(DEcounts, col = as.numeric(DEcounts$samples$group),
    pch = 19)
legend("bottomleft", as.character(unique(DEcounts$samples$group)),
    col = 1:2, pch = 19)
# Negative Binomial model
DEcounts <- estimateDisp(DEcounts)
plotBCV(DEcounts)  # dispersion estimation
et <- exactTest(DEcounts, dispersion = "tagwise",
    rejection.region = "deviance")
# deviance slightly less conservative,
# conditional likelihood ratio test
edge.top <- topTags(et, n = 10, adjust.method = "BH")  #  false discovery rate,
# = the expected proportion of false
# discoveries amongst the rejected
# hypotheses
library(xtable)
toptable <- as.data.frame(edge.top[, -2])
toptable <- cbind(rownames(toptable), toptable)
colnames(toptable)[1] <- "Genes"
tab <- xtable(toptable, booktabs = T, caption = paste("Top ten differentially expressed genes\n
    label = "edgeR-top")
print(tab, caption.placement = "top", table.placement = "H",
    include.rownames = F)
# # Show counts for top 10 genes o <-
# order(et$table$PValue)
# cpm(DEcounts)[o[1:10],]
# total number of differentially
# expressed genes at FDR 5%
sign.genes.edger <- summary(de <- decideTestsDGE(et,
    adjust.method = "BH", p.value = 0.05))
# gene namelist
genelist.edger <- rownames(DEdata[keep, ][de ==
    1 | de == -1, ])
# Plot log-fold change against log-counts
# per million - DE genes highlighted:
detags <- rownames(DEcounts)[as.logical(de)]
plotSmear(et, de.tags = detags)
abline(h = c(-2, 2), col = "dodgerblue")
################################################## SAMSeq
library(samr)
samfit.5 <- SAMseq(x = DEdata[keep, ], y = c(rep("1",
    3), rep("2", 3)), resp.type = "Two class unpaired",
    nperms = 500, random.seed = 43893, nresamp = 100,
    fdr.output = 0.05)
plot(samfit.5)
samfit.5.nofilter <- SAMseq(x = DEdata, y = c(rep("1",
    3), rep("2", 3)), resp.type = "Two class unpaired",
```

```
    nperms = 500, random.seed = 43893, nresamp = 100,
    fdr.output = 0.05)
plot(samfit.5.nofilter)
samfit.20 <- SAMseq(x = DEdata[keep, ], y = c(rep("1",
    3), rep("2", 3)), resp.type = "Two class unpaired",
    nperms = 500, random.seed = 43893, nresamp = 100,
    fdr.output = 0.2)
print(samfit.20)
plot(samfit.20)
samfit.20.nofilter <- SAMseq(x = DEdata,
    y = c(rep("1", 3), rep("2", 3)), resp.type = "Two class unpaired",
    nperms = 500, random.seed = 43893, nresamp = 100,
    fdr.output = 0.2)
plot(samfit.20.nofilter)
# retrieve list of significant gene names
genelist.samseq.5 <- c(samfit.5$siggenes.table$genes.up[,
    2], samfit.5$siggenes.table$genes.lo[,
    2])
genelist.samseq.5.nofilter <- c(samfit.5.nofilter$siggenes.table$genes.up[,
    2], samfit.5.nofilter$siggenes.table$genes.lo[,
    2])
genelist.samseq.20 <- c(samfit.20$siggenes.table$genes.up[,
    2], samfit.20$siggenes.table$genes.lo[,
    2])
genelist.samseq.20.nofilter <- c(samfit.20.nofilter$siggenes.table$genes.up[,
    2], samfit.20.nofilter$siggenes.table$genes.lo[,
    2])
# create testresults matrix-like object
# for limma venn diagrams (can decidetest
# be used on samseq output to make this
# easier?)  dont use DEdata[keep,]
# because then we cant compare filtered
# and unfiltered sets
sam.5.testresults <- sapply(rownames(DEdata),
    function(x) ifelse(x %in% genelist.samseq.5,
        1, 0))
sam.5.nofilter.testresults <- sapply(rownames(DEdata),
    function(x) ifelse(x %in% genelist.samseq.5.nofilter,
        1, 0))
sam.20.testresults <- sapply(rownames(DEdata),
    function(x) ifelse(x %in% genelist.samseq.20,
        1, 0))
sam.20.nofilter.testresults <- sapply(rownames(DEdata),
    function(x) ifelse(x %in% genelist.samseq.20.nofilter,
        1, 0))
# How many filtered genes were called as
# significant?
filtered.genelist <- rownames(DEdata[!keep,
    ])
sig.filter.5 <- sum(sapply(filtered.genelist,
    function(x) x %in% genelist.samseq.5.nofilter))
sig.filter.20 <- sum(sapply(filtered.genelist,
    function(x) x %in% genelist.samseq.20.nofilter))
# Comparison between filtered and
# unfiltered SAMSeq analysis
sam5.vs.nofilter <- cbind(sam5nofilter = sam.5.nofilter.testresults,
```

```
    sam5 = sam.5.testresults)
sam20.vs.nofilter <- cbind(sam20nofilter = sam.20.nofilter.testresults,
    sam5 = sam.20.testresults)
sam5.vs.sam20 <- cbind(sam20 = sam.20.testresults,
    sam5 = sam.5.testresults)
vennCounts(sam5.vs.nofilter)
vennCounts(sam20.vs.nofilter)
vennCounts(sam5.vs.sam20)
# sum(is.element(genelist.samseq,genelist.edger))
vennDiagram(vennCounts(cbind(samSeq5 = sam.5.testresults,
   samSeq20 = sam.20.testresults, samSeq5.nofilter = sam.5.nofilter.testresults,
    samSeq20.nofilter = sam.20.nofilter.testresults)),
    include = "both", counts.col = c("red",
        "dodgerblue"), circle.col = c("red",
        "dodgerblue", "green", "purple"))
# Comparing SAMSeq with edgeR need to use
# DEdata[keep,] for comparison with edgeR
sam.5.testresults.filter <- sapply(rownames(DEdata[keep,
    ]), function(x) ifelse(x %in% genelist.samseq.5,
    1, 0))
sam.20.testresults.filter <- sapply(rownames(DEdata[keep,
    ]), function(x) ifelse(x %in% genelist.samseq.20,
    1, 0))
edger.vs.samseq.5 <- cbind(edgeR = as.numeric(de@.Data),
    samSeq5 = sam.5.testresults.filter)
edger.vs.samseq.20 <- cbind(edgeR = as.numeric(de@.Data),
    samSeq20 = sam.20.testresults.filter)
# Venn diagram counts
vennCounts(edger.vs.samseq.5)
vennCounts(edger.vs.samseq.20)
vennDiagram(vennCounts(edger.vs.samseq.5),
    include = "both", counts.col = c("red",
        "dodgerblue"), circle.col = c("red",
        "dodgerblue", "green"))
vennDiagram(vennCounts(edger.vs.samseq.20),
    include = "both", counts.col = c("red",
        "dodgerblue"), circle.col = c("red",
        "dodgerblue", "green"))
# npseq symmetric cutoffs, not available
# in windows?
vennDiagram(vennCounts(cbind(samSeq5 = sam.5.testresults,
   samSeq20 = sam.20.testresults, samSeq5.nofilter = sam.5.nofilter.testresults,
    samSeq20.nofilter = sam.20.nofilter.testresults)),
    include = "both", cex = c(0.9, 0.9, 0.8),
    counts.col = c("red", "dodgerblue"),
    circle.col = c("red", "dodgerblue", "green",
        "purple"))
samseq.5.table <- as.data.frame(rbind(samfit.5$siggenes.table$genes.up,
    samfit.5$siggenes.table$genes.lo))
samseq.5.table <- samseq.5.table[order(samseq.5.table$`q-value(%)`),
    ]
# length(unique(samseq.5.table$`q-value(%)`))
# length(samseq.5.table$`q-value(%)`)
samseq.20.table <- as.data.frame(rbind(samfit.20$siggenes.table$genes.up,
    samfit.20$siggenes.table$genes.lo))
samseq.20.table <- samseq.20.table[order(samseq.20.table$`q-value(%)`),
```

```
    ]
# length(unique(samseq.20.table$`q-value(%)`))
unique.20 <- length(samseq.20.table$`q-value(%)`)
samseqtable <- xtable(cbind(samseq.5.table[1:10,
    -c(1, 3)], samseq.20.table[1:10, -c(1,
    3)]), include.rownames = F, booktabs = T,
   caption = paste("Top ten differentially expressed genes based on SAMSeq at 5\\% (left) and 20\\% FD
    label = "samseq-table")
print(samseqtable, caption.placement = "top",
    table.placement = "H", size = "scriptsize",
    include.rownames = F)
################################################## limma + voom
library(limma)
dge <- DGEList(counts = DEdata, group = factor(groupLabels))
dge <- dge[keep, , keep.lib.sizes = FALSE]  # filter
dge$samples$lib.size <- colSums(dge$counts)  # recompute library sizes
dge <- calcNormFactors(dge, method = "TMM")  # normalisation
# voom transform
design <- model.matrix(~as.factor(groupLabels))
v <- voom(dge, design, plot = TRUE)
plotMDS(v)
# limma model
fit <- lmFit(v, design)
fit <- eBayes(fit)
# top ten genes
limma.top <- topTable(fit, coef = ncol(design),
    adjust.method = "BH", number = 10)
# number of significant genes
all.limma <- topTable(fit, coef = ncol(design),
    adjust.method = "BH", number = dim(dge$counts)[1])
sign.limma <- all.limma[all.limma$adj.P.Val <
    0.05, ]
dim(sign.limma)[1]
# or look at treatment effect (ignore
# intercept)
sign.genes.limma <- summary(de.limma <- decideTests(fit,
    adjust.method = "BH", p.value = 0.05))
sign.genes.limma[, 2]
# gene namelist
limma.toptable <- as.data.frame(limma.top[,
    -c(2, 3, 6)])
limma.toptable <- cbind(rownames(limma.toptable),
    limma.toptable)
colnames(limma.toptable)[1] <- "Genes"
colnames(limma.toptable)[4] <- "FDR"
# table with top 10 genes
limma.tab <- xtable(limma.toptable, booktabs = T,
   caption = paste("Top ten differentially expressed genes\n                    based on voom+limma."
    label = "limma-top", include.rownames = F)
# volcanoplot(fit,coef = 2,highlight =
# dim(sign.limma)[1])
voom(dge, design, plot = TRUE)
print(limma.tab, caption.placement = "top",
    table.placement = "H", include.rownames = F)
limma::plotMA(fit, main = NULL)
# o <- order(fit$p.value[,2]) o <-
```

```r
# all.limma[order(all.limma[,'adj.P.Val']),]
# x <- fit$Amean y <-
# fit$coefficients[,2]
# points(x[o[1:dim(sign.limma)[1]]],
# y[o[1:dim(sign.limma)[1]]],
# col='red',pch=19,cex=0.5)
o <- all.limma[order(all.limma[, "adj.P.Val"]),
    ][1:dim(sign.limma)[1], ]
x <- o$AveExpr
y <- o$logFC
points(x, y, col = "red", pch = 19, cex = 0.5)
abline(h = c(-2, 2), col = "dodgerblue")
################################################## EBSeq
library(EBSeq)
# median normalisation of DESeq
Sizes = MedianNorm(DEdata[keep, ])
Conditions <- as.factor(groupLabels)
EBOut = EBTest(Data = as.matrix(DEdata[keep,
    ]), Conditions = Conditions, sizeFactors = Sizes,
    maxround = 7)
# check convergence
EBOut$Alpha
EBOut$Beta
EBOut$P
# bayesian fdr
EBDERes = GetDEResults(EBOut, FDR = 0.05)
# two columns PPEE and PPDE,
# corresponding to the posterior
# probabilities of being EE or DE for
# each gene
head(EBDERes$PPMat)
# contains each gene's status called by
# EBSeq
head(EBDERes$Status)
# top 10 genes
ebseq.toptable <- as.data.frame(EBDERes$PPMat)
ebseq.toptable <- ebseq.toptable[order(ebseq.toptable$PPDE,
    decreasing = T), ]
ebseq.toptable <- cbind(rownames(ebseq.toptable),
    ebseq.toptable)
ebseq.toptable <- ebseq.toptable[, -2]
colnames(ebseq.toptable)[1] <- "Genes"
colnames(ebseq.toptable)[2] <- "Posterior P. DE"
ebseq.tab <- xtable(ebseq.toptable[1:10,
   ], booktabs = T, caption = paste("Top ten differentially expressed genes based on EBSeq and their p
    label = "EBSeq-top", include.rownames = F)
# number of significant genes
length(EBDERes$DEfound)
length(EBDERes$Status[EBDERes$Status == "DE"])
# test-results-like matrix
EBSeq.testresults <- sapply(rownames(DEdata[keep,
    ]), function(x) ifelse(x %in% EBDERes$DEfound,
    1, 0))
print(ebseq.tab, caption.placement = "top",
    table.placement = "H", include.rownames = F)
# PlotPostVsRawFC(EBOut,PostFC(EBOut))
```

```
# Venn Diagrams
gene.comparison <- cbind(edgeR = as.numeric(de@.Data),
    SAMSeq.20 = sam.20.testresults.filter,
    SAMSeq.5 = sam.5.testresults.filter,
    EBSeq = EBSeq.testresults, limma = as.numeric(de.limma@.Data[,
        2]))
# Venn diagram counts
vennCounts(gene.comparison)
# Venn diagram plot
vennDiagram(vennCounts(gene.comparison),
    include = "both", counts.col = c("red",
        "dodgerblue"), cex = c(0.9, 0.9,
        0.8), circle.col = c("red", "dodgerblue",
        "green", "purple", "yellow"))
# without SAMSeq
vennDiagram(vennCounts(cbind(edgeR = as.numeric(de@.Data),
    EBSeq = as.numeric(EBSeq.testresults),
    limma = as.numeric(de.limma@.Data[, 2]))),
    include = "both", counts.col = c("red",
        "dodgerblue"), cex = c(0.9, 0.9,
        0.8), circle.col = c("yellow", "green",
        "purple"))
```