A Pipeline for Differential Expression Analysis of RNA-seq Data and The Effect of Filter Cutoff on Performance

by

Bonnie-Jean Robert
B.Sc., Vancouver Island University, 2013

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Mathematics and Statistics

A Pipeline for Differential Expression Analysis of RNA-seq Data and
The Effect of Filter Cutoff on Performance

by

Bonnie-Jean Robert
B.Sc., Vancouver Island University, 2013

Supervisory Committee

Dr. M. Lesperance, Supervisor
(Department of Mathematics and Statistics, UVic)

Dr. J. Zhou, Departmental Member
(Department of Mathematics and Statistics, UVic)

Supervisory Committee

Dr. M. Lesperance, Supervisor
(Department of Mathematics and Statistics, UVic)

Dr. J. Zhou, Departmental Member
(Department of Mathematics and Statistics, UVic)

ABSTRACT

RNA sequencing is a powerful new approach to analyzing differential expression of transcripts between treatments. Many statistical methods are now available to test for differential expression, each one reports results differently. This thesis presents a workflow of five popular methods and discusses the results. A pipeline was built in the R language to analyze four of these packages using a real RNA-seq dataset.

At present, researchers must prepare RNA-seq data prior to analysis to achieve reliable results. Filtering is a necessary preparatory step in which transcripts exhibiting low levels of genetic expression are removed from further analysis. Yet, little research is available to guide researchers on how best to choose this threshold. This thesis introduces a study designed to determine if the choice of filter threshold has a significant effect on individual package performance. Increasing the filtering threshold was shown to decrease the sensitivity and increase the specificity of the four statistical methods studied.

Contents

List of Tables

List of Figures

List of Abbreviations

RNA	RiboNucleic Acid
GLM	Generalized Linear Model
SAM	Significance Analysis of Microarrays
CPM	Counts Per Million
UQ	Upper Quartile
RPKM	Reads Per Kilobase of transcript per Million
ROC	Receiver Operating Characteristic
MA	Mean Average
MD	Mean Difference
PCA	Principal Component Analysis
DE	Differentially Expressed or Differential Expression
NDE	Non-Differentially Expressed or Non-Differential Expression
FDR	False Discovery Rate
TC	Total Count
TMM	Trimmed Method of Means

NB	Negative Binomial
QTL	Quantitative Trait Locus
FWER	Family-Wise Error Rate
RLE	Relative Log Expression
MLE	Maximum Likelihood Estimators
APL	Adjusted Profile Likelihood
BIC	Bayesian Information Criterion
AIC	Akaike Information Criterion
TPR	True Positive Rate
TNR	True Negative Rate
TP	True Positive
TN	True Negative
LFC	Log Fold Change
cDNA	Complementary DeoxyriboNucleic acid
MAP	Maximum a Posteriori Probability
LRT	Likelihood Ratio Test
CV	Coefficient of Variation
BCV	Biological Coefficient of Variation
CR	Cox Reid
REML	Residual Maximum Likelihood

ACKNOWLEDGEMENTS

I would like to thank:

Dr. Mary Lesperance, for your mentoring and support.

My parents for your encouragement and love.

My children, Eve and Mathieu for your understanding and patience.

DEDICATION

This paper is dedicated to my children, Eve and Mathieu Robert. They were with me through every step of this journey, through the best and most difficult times. They were the sole reason I kept working on days when I wanted to stop. I hope, in time, they will both find their dreams and the support and courage needed to pursue them.

Chapter 1

Introduction

Transcriptome profiling is a method known to be useful in understanding cell development and disease. One of the newest approaches to transcriptome profiling, RNA sequencing, uses deep sequencing technologies to achieve higher resolution than previously achieved by its predecessors. RNA sequencing technology was developed to overcome the short-comings of earlier microarray and tag-based sequencing technologies such as the limited dynamic range and mapping abilities of short tag sequencing. Amongst some of the unique benefits to RNA sequencing are intron/extron identification, boundary mapping and identification of start sites.

The sequencing process starts with obtaining a library of DNA fragments from a sample of RNA. While the details of the complex biochemical process is beyond the scope of this thesis, in general it involves transcribing RNA into cDNA, which is then reduced to specific size fragments that are introduced into a library. The library is sequenced in a state-of-the-art sequencer to obtain a collection of reads that are subsequently mapped to a reference genome, if one is available. Otherwise, the reads are aligned in an attempt to reconstruct a genome in a process termed *de novo*. After mapping the reads to the genomic features of interest, such as transcripts, the reads for each feature are counted and reported. The read counts are analyzed to identify features for which there exists a statistically significant difference between experimental conditions.

The work in this report focuses on the identification of RNA transcripts that exhibit significant differences in abundance between treatment conditions. If a transcript has been

determined to be **differentially expressed (DE)** between treatment conditions, this would indicate that RNA abundance transcribed from this area of the genome has been affected by the treatment imposed on the biological entity. Analysis of differential expression is complex and must consider information such as the sources of variation affecting transcript read counts, and sample **library size**, or total number of mapped reads for a particular sample. The existence of many lowly expressed transcripts versus a few highly expressed ones can have a signinfcant impact on analysis as well.

Various methods have been developed to model RNA-seq read counts. Several are based on the Poisson distribution, which has been shown to model data well when the sources of variation arise purely from technical sources. Overdispersion is a phenomenon that has been shown to be inherent to RNA-seq data. Overdispersion is due to the nature of biological entities, which express RNA at differing relative abundancies. For this reason, many methods use the negative binomial, or overdispersed Poisson distribution to model RNA-seq data as it has an extra disperison parameter. Several non-parametric and Bayesian methods have been developed model RNA-seq data and two of them, SAMSeq [45] and BaySeq [17] are reviewed in this paper.

Because the underlying theoretical framework for these packages can be vastly different, it is not clear which method is 'best'. It has been suggested that each method performs slightly differently depending on the conditions of the dataset, with no one method outperfoming the others [35]. When choosing a method with which to use to analyze an RNA-seq dataset, an option is to examine several methods and choose the one best suited to the data.

There are several methods for differential expression analysis available within the R statistical software framework [34]. In this paper we have chosen five of the more popular ones to study. A pipeline is developed that allows us to run a dataset through all five packages simultaneously, producing descriptive statistics and analysis results. The results from running a real RNA-seq dataset through this pipeline are discussed in this paper.

Additionally, we run a study to examine the effects of filtering on package performance. The number of DE hypothesis tests conducted on a typical RNA-Seq dataset can be very large and many truly null hypotheses may give small P-values by chance. Several procedures have been developed to address this multiple testing problem, with the goal to control the number of false positives detected. These procedures can decrease the power of a test, and in turn decrease the ablility to detect truly differentially expressed transcripts [5].

Transcripts with low counts across all samples provide little evidence for differential expression. Many authors of microarray literature have proposed removing transcripts that generate uninformative signals prior to analysis. Several methods for filtering transcripts have been proposed, including filtering transcripts with a total read count less than some chosen cut-off. To our knowledge, little attention has been paid to the impact of this cut-off value on differential expression [5]. It appears that, in practice, it is chosen rather arbitrarily [13] [14]. In this paper we study the effects of varying the filter cut-off on the performance of four of our five chosen packages. The details and results of this study are found in Chapter 8.

In Chapter 2 is a brief description of several pieces of literature cited as key references for this paper. Included in the literature review are four reviews of different models for differential expression and three comparison studies of several R Bioconductor packages available for differential expression analysis. A brief description of the dataset used to test five of the packages is found at the end of the chapter.

Chapter 3 contains technical information on the SAMSeq procedure, one of the methods used in this paper to test for differential expression. SAMSeq [23] is a non-parametric based extension of the R package, samr [45], which was designed to test for differential expression of transcript abundance in microarray analysis.

Chapter 4 contains technical information on the BaySeq procedure, another method used in this paper to test for differential expression. BaySeq [17] is a statistical analysis package developed for the R environment [34] to detect DE transcripts given a set of RNA-seq data. It uses an empirical Bayesian approach to borrow information from across the dataset, improving the accuracy of predictions.

Technical information on the final three methods tested, DESeq, EdgeR, and EdgeR Robust are found in Appendices A and B. Both sections were prepared by UVic Master's degree student Xin Yu for his graduation project under supervisor Dr. Mary Lesperance.

Chapter 5 provides information regarding two important steps in differential expression analysis. Filtering is the process by which low-count transcripts are removed from the dataset. These transcripts tend to add little information to the dataset, and can decrease the power of a test for differential expression. In practice, they are frequently removed. Normalization is the process in which transcript counts are scaled to be directly comparable

between sample replicates.

In Chapter 6, a workflow is described for a sample RNA-seq dataset analyzed for differential expression. Descriptive statistics are reported on the dataset before running the data through five R Bioconductor packages. The differential expression results are reported and examined. The R code for the pipeline is found in this chapter.

In Chapter 7 is a technical review of several algorithms considered for creating simulated data. All were taken from previous publications [24], [47], [41].

Chapter 8 describes an experiment designed to determine what effect the choice of filter cut-off has on a statistical method's ability to detect differential expression. In particular, does increasing the read count cut-off value (0, 1,...) have any effect on test performance?

Plots and output from a generalized linear model are included in Appendix D.

Chapter 2

Background

2.1 Literature Review

While Poisson models are reported to handle technical replicates well, biological replicates are often modelled with a negative binomial, due to the variability inherent between biological replicates. The negative binomial can handle overdispersion, but requires a procedure to adjust for replicate **sequencing depth**, a measure of relative library size [23].

As the cost of RNA sequencing can be considerable, producing replicates for an experiment can be inhibiting. Robinson [40] developed a statistical test to model overdispersion that performs well on small datasets. Counts were modelled using a negative binomial (NB) distribution, and dispersion was estimated through a weighted conditional likelihood framework. The method was run on both simulated and real data and the findings showed that false discovery rates were comparable to those found in a previous study by Lu, Tomfohr and Kepler [28].

Li, Witten et al. [24] proposed another method, PoissonSeq. PoissonSeq is a log-linear based model used for analysis of RNA-seq data and incorporates a novel method of normalizing the data. A new procedure to estimate the **false discovery rate (FDR)** is proposed that does not rely on p-values. A simple transformation is applied to handle overdispersion. PoissonSeq was studied over two real datasets and a simulation study. FDR estimates from differential expression analysis were found to be comparable to those calculated using EdgeR

[39]. It was noted that some assumptions were made that yielded practical limitations, such as transformation power depending only on gene expression, and the genes being independent from each other. It was noted that default parameters set in the paper may not perform well in every case.

Through a data simulation analysis, Li and Tibshirani [23] show that the presence of outliers can have a large impact on the FDR estimate if an underlying parametric distribution is assumed. This motivated the development of a non-parametric method to detect differential expression. According to the authors, SamSEQ performs comparatively to parametric based methods when their distribution assumption holds, and better when the assumption does not hold. SamSeq is able to handle data of several types, including quantitative, multiple-class and survival outcomes.

Zhou, Lindsey and Robinson [47] developed a new method for modelling RNA-seq data using weighted observations to dampen the global effects of extremely high counts, or outliers, on the parameter estimates. This method is one of the methods used to analyze our data for this study and is referred to as *robust EdgeR*. Robust EdgeR is implemented in the R Bioconductor package EdgeR [39], used for differential expression analysis. Performance was measured against several other popular R software packages, including DeSeq, a differential expression estimation method based on the negative binomial model. The method was shown to decrease EdgeRs sensitivity to outliers while still exhibiting good power. The methodology is easily adaptable to other methods. A simulator was developed for the paper, it is described in Chapter 2 and can be sourced online [38].

Lior Pachter reviewed several models for estimating relative transcript abundance of RNA-seq counts [32]. These include models based on the Poisson, negative binomial and multinomial distributions. He notes models based on the multinomial and Poisson distributions have similar mathematical forms and produce equivalent differential expression results. He suggests that models based on the negative binomial tend produce slightly more conservative differential expression estimates than multinomial models. Overall, the author suggests one general model may suffice, as no one model out performs the other in all circumstances.

Soneson and Delorenzi [43] performed a comparison study of several methods for performing differential expression analysis on simulated and real RNA-seq data. These methods are all available through the statistical computing framework R [34]. DESeq2 [1], EdgeR

[39], NBPSeq [9], TSPM [3], BaySeq [18], EBSeq [21], NOISeq [44], SAMseq [23] and Shrink-Seq [46] work on the counts directly, while vst [1] and voom [37] transform the data before running them through Limma [37]. The study examined the performance of each method in terms of how well it controlled for Type I error rates and true positive rates, the ability to rank truly DE transcripts ahead of non-DE transcripts, and computational time. Varying conditions were placed on the simulated data, including sample size and composition (up-regulated or down-regulated). The study found that presence of outliers, sample size, number of replicates, and composition of the data affected method performance. No one method performed optimally under all tested conditions.

Rapaport, Kahanin et al. [35] performed a comparison study on RNA-seq data of the performance of the following differential expression software tools: Cuffdiff, Limma [37], EdgeR [39], DESeq [1], PoissonSeq [24] and Bayseq [18]. Each tool differs in the way it handles normalization and differential expression analysis of count data. Simulated data was created under varying conditions of number of replicates and sequencing depth. Normalization was evaluated by examining sample clustering post-normalization. Differential expression results were examined by **Receiver Operating Characteristic (ROC)** analysis, p-value density plots and calculating log_2 expression changes. Each method was tested for performance using FDR as a measure. Results showed not all methods performed similarily, and not one method outranked the others under all conditions.

Dillies, Rau et al. [10] compared the following popular methods for normalizing RNA-seq data: Total Count (TC), Upper Quartile (UQ), Median (Med), Trimmed Mean of Mean (TMM), Quantile (Q), Read per Kilobase per Million (RPKM) and the normalization method used in DESeq2 (DESEQ) [26]. Four real datasets were used to perform the comparison and conditions were varied over the simulated dataset. Under simulated conditions of equal library sizes and no high-count transcripts, the seven methods behaved comparably overall. Comparison over real datasets showed significant differences and the authors confirmed previous results that RPKM and TC should not be used. RPKM was reported to be insufficient in removing length bias. It was suggested that the strong assumption of identical read count distributions used by the quantile method might increase between-group variability. Under simulated conditions where high-counts were present, TMM maintained good false-positive rate without loss of power. Results suggest DESeq and TMM perform the best, followed by UQ and Med.

2.2 WEC Data

RNA-seq count data was generated for olfactory bulbs of tadpoles exposed to influent cocktail exposure in 2014. The data is of 4 animals C1, C2, C3 and C4 from the vehicle control exposed set and 4 animals T1, T2, T3 and T4 (T is for treatment) exposed to the chemical cocktail. Each sample consists of counts for 65,499 transcripts.

To determine the effects of the chemical cocktail on the animals, we use several Bioconductor R [34] statistical packages to test for differential expression of the transcripts. A review of recent literature lacked conclusive results as to which statistical method was superior. Therefore five packages were chosen based on popularity. The packages we use are, DESeq2 [26], SamSEQ [45]. BaySeq [17], EdgeR and Robust EdgeR [39],

For this analysis, a workflow was developed to analyze the data from each of the five methods. The workflow consists of an exploratory examination of the data followed by analysis of differential expression levels using each method. Details and discussion on the pipeline are found in Chapter 6.

Chapter 3

SamSeq Technical Review

3.1 Introduction to SamSeq

Many R packages analyze differential transcript expression using parametric methods, which typically model the variance between samples with a Poisson or negative binomial distribution. Non-parametric methods are an alternative which do not rely on knowing from which particular distribution the counts are drawn. Problems encountered by parametric methods such as over-dispersion are avoided with non-parametric methods, and they can be more robust in the presence of outliers [23].

SAMSeq [23] is a non-parametric based extension of the R package, samr [45], which was designed to test for differential expression of transcript abundance in microarray analysis. This method uses a resampling strategy to construct a Wilcoxon statistic. The statistic is used to calculate a cut-off value for the FDR through a permuation plug-in method as outlined below.

3.2 The Data

Suppose we have count data for p transcripts and n samples that is in the form of a p x n matrix $\mathbf{N}$, of counts. Each N_{gi} represents the readcount from transcript g and sample i. The

expected value of N_{gi} depends on the transcript expression as well as the total number of reads in each replicate, or library size. A first step in differential expression is to account for the different library sizes. Normalizing involves obtaining a standardized count, N'_{gi}, that can be used to directly compare counts for transcript g across all replicates.

The term **sequencing depth** is used as a relative measure of the overall read count, or **library size**, of a sample. In practice, the sequencing depth of a sample, d_i, can be estimated by one of several methods including TMM, the default method for EdgeR [39]. Once these sequencing depth have been determined, normalized counts, N'_{gi} can be obtained by letting $N'_{gi} = d_i * N_{gi}$.

According to Li and Tibshirani, [23], rescaling the total read counts by their relative sequencing depths, d_i, may not adequately account for variation within the samples. In their paper, *Finding consistent patterns: A non-parametric approach for identifying differential expression in RNA-Seq data*, they present the following procedure to address this which they refer to as **down-sampling**.

They let $d_1, \cdots, d_n$ be the sequencing depths for samples $i = 1, \cdots, n$ and let sample i_{min} be the sample with the smallest sequencing depth, d_{min}. The rest of the samples are shortened to have sequencing depth d_{min}. This is done by randomly selecting each read within a sample i, with probablility d_{min}/d_i and discarding it with probability $1 - d_{min}/d_i$. Then the number of reads mapped to feature g is drawn from:

$$N'_{gi} \sim \text{Binom}(N_{gi}, d_{min}/d_i) \tag{3.1}$$

Li and Tibshirani, [23] found that the number of reads discarded increase as d_{min} decreased and instead chose to resize the sequencing depths by the geometric mean, $\bar{d} = (\prod_{i=1}^{n} d_i)^{\frac{1}{n}}$ so that

$$N'_{gi} \sim \text{Poisson}(\frac{\bar{d}}{d_i} * N_{gi}) \tag{3.2}$$

They called the above method **Poisson Sampling**. In a comparison using simulated data with the SAMSeq [45] method, down-sampling and Poisson sampling produced similar results when $d_{max}/d_{min} < 10$ [45], otherwise Poisson-sampling performed significantly better.

For further details regarding estimating sequencing depths, see Chapter 5.

3.3 Wilcoxon Statistic

The purpose of many comparative RNA-seq experiments is to identify features that are over or under expressed between two or more groups. Consider the following two-group case.

Define the set C_1 containing n_1 samples from group 1, and C_2 containing n_2 samples from group 2. Let N_{gi} be the set of counts normalized by sequencing depth, d_i. Then the counts are directly comparable and $N_{gi_1} > N_{gi_2}$ means the expression of transcript g is higher in sample i_1 then sample i_2.

Each count is assigned a rank, $R(N)$, across the n samples, where $1 \leq R_i(N) \leq n$, such that $R_{gi_1}(N) \leq R_{gi_2}(N)$ if and only if $N_{gi_1} < N_{gi_2}$ for some transcript g. Ties can broken by adding a small random number, ϵ to each count, where $\epsilon_{gi} \sim$ Uniform $(0, 0.1)$, $1 \leq i \leq n, 1 \leq g \leq p$.

Then the two-sample Wilcoxon Statistic for each feature, g, is

$$T_g = \sum_{i \in C_1} R_{gi}(N) - n_1(n+1)/2 \tag{3.3}$$

The expected value under the null distribution of no differential expression for $\sum_{i \in C_1} R_{gi}(N)$ is $n_1(n+1)/2$. When feature g is not differentially expressed,

$$\mathbf{E}T_g = \mathbf{E}\big[\sum_{i \in C_1} R_{gi}(N) - n_1(n+1)/2\big] = 0 \tag{3.4}$$

3.4 Multiple Resampling

There are two drawbacks to the above process. One is that many reads are discarded in the above resampling process, and the other is that resampling and the procedure used to

break ties both bring randomness to the results, which may be significant for transcripts with small counts. This in turn can decrease the power of the Wilcoxon statistic. The loss can be minimized by resampling S times, and taking the average. Resampling $S = 20$ times has been shown to be large enough to reduce noise. Then the statistic is

$$T_g^* = \frac{1}{S}\big[\sum_{s=1}^{S}\sum_{i\in C_1} R_{gi}(N_s') - n_1(n+1)/2\big]. \tag{3.5}$$

It's important to note that although the distribution of Wilcoxon statistics, T_g, is known, the distribution of T_{g^*} is not.

3.5 Estimating the FDR

As the number of features in a typical RNA-Seq dataset is typically large, it is preferable to consider the **False Discovery Rate (FDR)** as a measure of differential expression. The FDR is the expected proportion of false positives in the set of significant features.

As the true distribution of T_g^* is unknown, the FDR must be estimated. SAMSeq uses the following method to estimate the FDR.

3.6 The SAM Procedure

Consider the set of counts, N_{gi}, $g = 1, 2, \cdots, p$ and $i = 1, 2, \cdots, n$.

1. Compute the statistic, T_g^*
2. Compute order statistics $T_{(1)}^* \leq T_{(2)}^* \cdots \leq T_{(p)}^*$
3. Take B sets of permutations of the response values y_j. The response variable for two-class data is a grouping indicating treatment like *(untreated, treated)*. For each permutation b, $b = 1, \cdots, B$ compute statistics T_g^{*b} and corresponding order statistics $T_{(1)}^{*b} \leq T_{(2)}^{*b} \cdots \leq T_{(p)}^{*b}$

4. From the set of B permutations, estimate the expected order statistics by $\bar{T}_{(g)} = 1/B \sum_b T^{*b}_{(g)}$ for $g = 1, 2, \cdots p$

5. Plot the $T^*_{(g)}$ values versus the $\bar{T}_{(g)}$.

6. For a fixed threshold Δ, starting at the origin, and moving up to the right find the first $g = g_1$ such that $T_{(g)} - \bar{T}_{(g)} > \Delta$. All transcripts past g_1 are called **significant positive**. Similarly, starting at origin, move down to the left and find the first $g = g_2$ such that $\bar{T}_{(g)} - T_{(g)} > \Delta$. All transcripts past g_2 are called **significant negative**. For each Δ define the upper cut-point $cut_{up}(\Delta)$ as the smallest T_g among the significant positive transcripts, and similarly define the lower cut-point $cut_{low}(\Delta)$.

7. For a grid of Δ values, compute the total number of significant transcripts (from the previous step), and the median number of falsely called transcripts, by computing the median number of values among each of the B sets of $T^{*b}_{(g)}, i = 1, 2, \cdots p$, that fall above $cut_{up}(\Delta)$ or below $cut_{low}(\Delta)$. Similarly for the $90th$ percentile of falsely called transcripts.

8. Estimate π_0, the proportion of true null (unaffected) transcripts in the dataset, as follows:

   (a) Compute $q25$, $q75 = 25\%$ and 75% points of the permuted T values (if $p = \#$ transcripts, $B = \#$ permutations, there are pB such T values).

   (b) Compute $\hat{\pi}_0 = \#(T_g \in (q25, q75)) / (.5p)$ (the T_g are the values for the original dataset: there are p such values.)

   (c) Let $\hat{\pi}_0 = \min(\hat{\pi}_0, 1)$ (i.e., truncate at 1).

9. The median and $90th$ percentile of the number of falsely called transcripts from step 6, are multiplied by $\hat{\pi}_0$.

10. User then picks a Δ and the significant transcripts are listed.

11. The **False Discovery Rate (FDR)** is computed as [median (or $90th$ percentile) of the number of falsely called transcripts] divided by [the number of transcripts called significant]

3.7 Estimating Other Measures

Fold Change Suppose $\bar{N}_{g1}$ and $\bar{N}_{g2}$ are the average expression levels of transcript g under each of two conditions. These averages refer to raw data. Then if a nonzero fold change t is specified, then a positive transcript must satisfy $|\bar{N}_{g1}/\ \bar{N}_{g2}| \leq 1/t$ in order to be called significant and a negative transcript must satisfy $|\bar{N}_{g1}/\ \bar{N}_{g2}| \geq t$ to be called significant. When a fold change is specified, transcripts with either $\bar{N}_{g1} \leq 0$ or $\bar{N}_{g2} \leq 0$ (or both) are automatically left off the significant transcript list, as their fold change cannot be unambiguously determined. When such fold changes are reported in output, they are indicated by `NA`. [45]

q-value The q-value of a transcript is the false discovery rate for the transcript list that includes that transcript and all transcripts that are more significant. It is computed by finding the smallest value of $\hat{\Delta}$ for which the transcript is called significant, and then is the FDR corresponding to $\hat{\Delta}$. [45]

local FDR The local FDR for a transcript, g, is the FDR for transcripts having a similar score, T_g^* (Equation 3.5). It is estimated by taking a symmetric window of 0.5% of the transcripts on each side of the target transcript, and estimating the FDR in that window. If the total number of transcripts is less than 50, then the percentage is increased so that the number of transcripts is 50 [45].

3.8 Simulation Study

Li and Tibshirani [45] compared the SamSEQ method against the following popular parametric-based packages: DeSeq [1], PoissonSeq [24] and edgeR [39] in a simulation study. Datasets were constructed to satisfy both of the following models.

$$N_{gi} \sim NB(\mu_{gi}, \phi) \tag{3.6}$$

and

$$N_{gi} \sim \text{Poisson}(\mu_{gi}) \tag{3.7}$$

With link functions:

$$\log \mu_{gi} = \log d_i + \log N_g + \gamma_g I_{j \in C2}. \tag{3.8}$$

Here, d_i is the sequencing depth for sample i, N_g is the expression level of transcript g in the first group, and γ_g is the differential expression level of transcript g.

As in [24], the simulation satisfied the following:

- d_i were similar to real RNA-Seq experiments and maximum d_i was approximately seven times the minimum.
- the profile of transcript expression levels is similar to a real RNA-Seq dataset.
- γ_g are simulated so that the average fold change for significant features was approximately 2.7.
- p = 20,000 features were simulated, which is roughly the number of transcripts in the human genome.
- the dispersion parameter for the NB distribution was set to 0.25.
- 30% of the features were set to be differentially expressed, as opposed to 10% set by Li et Tibshirani [24].
- The data consisted of 12 samples in each of the two groups.

The results of the comparison show SAMSeq yields competitive true false discovery rates (FDRs) when the parametric assumptions of the other methods hold, no outliers exist in the data, and sample size is moderate.

Outliers were introduced by first generating μ_{gi} according to (3.8), and then setting $\mu_{gi} = 10\mu_{gi}$ with probability 0.01. Results showed that once the outliers were introduced,

parametric assumptions were violated. The true FDR for all three parametric methods became unacceptably high and they were still underestimated. SAMSeq's performance was not affected by the presence of outliers.

For datasets with small sample sizes (≤ 5 per group) and outliers introduced as above, Li and Tibshirani reported that SAMSeq still appeared to perform acceptably well. This was despite concerns about the ability for SAMSeq to generate an accurate null distribution, and the comparable efficiency of parametric methods when their distribution assumptions hold. In general, SAMseq overestimates the FDR mainly because of the overestimation of π_0, the true proportion of non-differentially expressed transcripts. Fortunately, the authors felt this overestimation should still be acceptable. The three parametric methods failed to give reasonable estimates of FDRs.

Three real RNA-seq datasets were analyzed to compare SAMSeqs performance with other methods. One dataset was Poisson distributed, contained little noise and consisted of few outliers. SAMSeq achieved competitive FDRs with this dataset. There was a large overlap amongst differentially expressed transcripts found by the different methods.

Two other datasets were examined, both overdispersed with outliers. There was significant difference in performance from all methods. The overlap of common transcripts found was minimal. It cannot be ascertained which method performed better, as the truly differentially expressed transcripts were unknown. However, closer examination of the top transcripts revealed that those found by SAMSeq tended to have consistently larger counts across samples in one group over the other. Whereas the parametric methods tended to find features which could be considered outliers.

SAMSeq can be extended to handle multiple classes by means of an altered Kruskal-Wallis test statistic. Quantitative and survival data can also be handled by SAMseq. This was demonstrated by a simulation study with 20,000 features and 24 samples. Overall, the developers of SAMSeq claim it is method that is robust to outliers. They claim SAMSeq tends to discover differentially expressed transcripts with a strong treatment effect, one that is seen across most of the samples in a treatment group.

Chapter 4

BaySeq Technical Review

4.1 Introduction to BaySeq

BaySeq [17] is a statistical analysis package developed for the R environment [34] to detect DE transcripts given a set of RNA-seq data. It uses an empirical Bayesian approach to borrow information from across the dataset, improving the accuracy of predictions. The data are assumed to be negative binomially distributed, as shown in previous work by Robinson and Smyth [40], and Lu et al. [28]. It is assumed that samples exhibiting similar patterns of differential expression share parameters drawn from the same prior distribution. From these assumptions, an empirical distribution is derived from which to estimate posterior probabilities for each model. Differential expression is determined from these estimated posterior probablities.

In this paper we focus on a set of models designed for simple pairwise comparisons, however it should be noted that baySeq allows for more complex experimental designs [18]. In their paper, *baySeq: Empirical Bayesian Methods for Identifying Differential Expression in Sequence Count Data*, the authors Hardcastle and Kelly showed this method to be comparable or more superior to existing packages in terms of performance on simulated and real datasets [18].

4.2 Methods

4.2.1 Model Definition

Consider a set of RNA-seq data, with n samples and p transcripts. Let $A = \{A_1, \cdots, A_n\}$, A_i is a vector of length p, be the entire set of samples with library size scaling factors $\{l_i, \cdots, l_n\}$. Let u_{gi}, $i = 1, \cdots, n$, $g = 1, \cdots, p$ be the count of transcript g in sample i. Then we can define our data for transcript g to be $D_g = \{(u_{g1}, \cdots, u_{gn}), (l_1, \cdots, l_n)\}$.

Then a model can be defined by sets of samples $E_1, \cdots, E_m$ such that two samples, A_i and A_j, are in the same set, E_q, if and only if they share the same underlying distribution parameters, θ_q. It is assumed that the θ's are independent from each other and we define $K = \{\theta_1, \cdots, \theta_m\}$ to be the set of all parameters in the model.

For example, consider a set of RNA-seq data consisting of three control and three treatment samples. Then models for no differential expression, M_0, and differential expression, M_1, are as follows.

Model 0 : $M_0 = E_1$ where $A_i \in E_1, i = 1, .., 6$

Model 1 : $M_1 = E_1, E_2$ where $A_i \in E_1, A_j \in E_2$ for $i = 1, 2, 3$ and $j = 4, 5, 6$

Then, in the above case of two groups, the BaySeq method seeks to find posterior likelihoods for any given model, M, using Bayes rule:

$$\mathbb{P}(M|D_g) = \frac{\mathbb{P}(D_g|M)\mathbb{P}(M)}{\mathbb{P}(D_g)} \tag{4.1}$$

We can then attempt to calculate $\mathbb{P}(M|D_g)$ by considering the marginal likelihood.

$$\mathbb{P}(D_g|M) = \int \mathbb{P}(D_g|K, M)\mathbb{P}(K|M)\,\mathrm{d}K \tag{4.2}$$

4.2.2 Distribution

There are several choices of distributions for $(D_g|K, M)$ and $(K|M)$. A natural choice is the Poisson distribution with parameters drawn from a Gamma distribution, however the Poisson distribution does not account for the extra variability introduced by biological replicates. The negative binomial (NB) distribution has been shown to be a good choice of distribution for modelling RNA-seq data [40] and is robust even when the data are not NB distributed [28]. Even if the data are not truly NB distributed, the NB can still be used [18]. From here forward, the data are assumed to follow a negative binomial distribution with parameters $\theta_q = (\mu_q, \phi_q)$.

Given equal library sizes across all samples, an exact test can be developed for the likelihood of observing the data given there is no difference in expression of transcripts between groups. In the case of unequal library sizes, pseudodata can be generated with a distribution equivelent to that of the real data [40]. In the BaySeq method, original library sizes are retained and used as a scaling factor for the real data.

Consider a count, u_{gi} from sample A_i, that has scaling factor l_i and belongs to group E_q . Then $u_{gi} \sim NB(\mu_q l_i, \phi_q)$ where $\theta_q = (\mu_q, \phi_q)$, where $\mu_q l_i$ is the mean and ϕ_q is the dispersion parameter. Then

$$\mathbb{P}(u_{gi}; l_i, \mu_q, \phi_q) = \frac{\Gamma(u_{gi} + \phi_q^{-1})}{\Gamma(\phi_q^{-1}) u_{gi}!} \left[\frac{1}{1 + l_i \mu_q \phi_q}\right]^{\phi_q^{-1}} \left[\frac{l_i \mu_q}{\phi_q^{-1} + l_i \mu_q}\right]^{u_{gi}} \tag{4.3}$$

Note this implies that the count for sample A_i is dependent on l_i. To solve eq. (4.1) for $\mathbb{P}(M|D_g)$ we can numerically compute $P(D_g|M)$ using

$$\mathbb{P}(D_g|M) = \int \mathbb{P}(D_g|K, M)\mathbb{P}(K|M)\mathrm{d}K \tag{4.4}$$

An empirical distribution can be defined on K allowing for numerical estimation of $\mathbb{P}(D_g|M)$. If $\theta_q \in K$ are assumed to be independent with respect to q, and D_{qg} is the data associated with transcript g in set E_q, $q \leq m$, then eq. (4.1) becomes

$$\mathbb{P}(D_g|M) = \prod_q \int \mathbb{P}(D_{qg}|\theta q)\mathbb{P}(\theta_q)d\theta_q \tag{4.5}$$

reducing the dimensions of the integral and improving the accuracy of numerical approximation. For a set of values, Θ_q of size $|\Theta_q|$, sampled from the distribution of θ_q, the following approximation can be derived.

$$\mathbb{P}(D_g|M) = \prod_q \frac{1}{|\Theta_q|} \sum_{\Theta_q} \left[\prod_{i:A_i \in E_q} \frac{\Gamma(u_{gi} + \phi_q^{-1})}{\Gamma(\phi_q^{-1})u_{gi}!} \left[\frac{1}{1 + l_i\mu_q\phi_q} \right]^{\phi_q^{-1}} \left[\frac{l_i\mu_q}{\phi_q^{-1} + l_i\mu_q} \right]^{u_{gi}} \right] \tag{4.6}$$

Consider the data, D_{gq}, associated with some transcript g from group q. If the mean and dispersion for D_{gq} can be estimated for a large number of transcripts, then this is sufficient to create the set, Θ_q that defines a distribution from which to draw θ_q.

4.2.3 Empirically Derived Distributions on K

One difficulty that lies in estimating the dispersion is that it is unknown in advance whether a transcript is differentially expressed. If the model assumes no differential expression, then the dispersion will be over-estimated in transcripts that truly exhibit differential expression. To work around this, the replicate structure must be considered in estimating the dispersion.

Consider the two sample design consisting of the set $\{F_1, F_2\}$ where $i, j \in F_r$ if A_i and A_j are replicates. That is, F_1 defines the controls, $\{A_i|i = 1, 2, 3\}$, and F_2 defines the treatments $\{A_i|i = 4, 5, 6\}$. Quasi-likelihood methods can be used to estimate the dispersion for some set of data, D_g, by first defining $\hat{\mu}_{gr} = \left\langle \left\{ \frac{u_{gi}}{l_i} | i \in r \right\} \right\rangle, r = 1, 2$ and choose ϕ_g so that

$$2\sum_r \sum_{i \in F_r} \left\{ u_{gi} \log \left[\frac{u_{gi}}{l_i\hat{\mu}_{gr}} \right] - (u_{gi} + \phi_g^{-1}) \log \left[\frac{u_{gi} + \phi_g^{-1}}{\phi_{gr}^{-1} + l_i\hat{\mu}_{gr}} \right] \right\} = n - 1 \tag{4.7}$$

The ϕ_g that satisfies eq. (4.7) can be used to estimate $\hat{\mu}_{gi}$ by maximiziming the likelihoods for μ_{gi}, $i \in F_r$.

$$\mathbb{P}\big(u_{gi}|l_i,\hat{\mu}_{gr},\phi_g\big)=\prod_{i\in F_r}\frac{\Gamma(u_{gi}+\phi_g^{-1})}{\Gamma(\phi_g^{-1})u_{gi}!}\left[\frac{1}{1+l_i\hat{\mu}_{gr}\phi_g}\right]^{\phi_g^{-1}}\left[\frac{l_i\hat{\mu}_{gr}}{\phi_g^{-1}+l_i\hat{\mu}_{gr}}\right]^{u_{gi}} \tag{4.8}$$

These estimates for ϕ_g and $\hat{\mu_{gi}}$ can then be used as starting values for an iterative process of re-estimation of ϕ_g. The final value for ϕ_g yields an MLE for μ_{gq} for each q.

$$\mathbb{P}(D_{gq},\phi_g,\mu_{gq})=\prod_{i:A_i\in E_q}\frac{\Gamma(u_{gi}+\phi_g^{-1})}{\Gamma(\phi_g^{-1})u_{gi}!}\left[\frac{1}{1+l_i\hat{\mu}_{gr}\phi_g}\right]^{\phi_g^{-1}}\left[\frac{l_i\hat{\mu}_{gr}}{\phi_g^{-1}+l_i\hat{\mu}_{gr}}\right]^{u_{gi}} \tag{4.9}$$

Repeating this process for multiple tuples, a dataset of $\Theta_q=(\mu_{gq},\phi_g)$ can be formed and used to solve equation (4.6) for $\mathbb{P}(D_g|M)$.

This method of estimating the dispersion assumes the dispersion of a transcript is constant across different sets of sample replicates. The authors note that for small n, this assumption is sufficient [18].

4.2.4 Prior on Model

Finding a reasonable prior to use in equation 4.1 can occasionally involve estimating a distribution using other sources. It is assumed here that earlier methods used in microarray analysis can be adopted to achieve this.

An initial prior probability, p^*, is chosen based on prior knowledge of the models, and is used to estimate $\mathbb{P}(M|D_g)$. A new estimate, $p^{*'}=\big\langle\mathbb{P}(M|D_g)\big\rangle$ can be formed by iterating through $\mathbb{P}(D_g|M)$ from equation 4.6 until convergence. The authors note that the initial value for p^* has no substantial effect on the value of convergence.

This approach is easily implemented but has the potential for positive feedback resulting in over-estimation of the prior probablity over of a model.

4.2.5 Scaling Factor

The term $\mathbb{P}(D_g)$ from Equation 4.6 is refered to as a **scaling factor**. The scaling factor is calculated by summing over all possible values for M. In the two group case, the set of M is small. Were M to be quite large, strategies to limit the number of models could be imposed, include considering only those that are biologically plausible, and imposing a distribution on the number of sets for M.

4.3 Performance and Conclusion

Hardcastle and Kelly [18] measured the performance of baySeq by reproducing simulations that had been performed by Robinson and Smyth [40] to assess EdgeR's performance. They used a real RNA-seq dataset with a known treatment outcome to assess baySeq's ability to detect true differential expression. It was determined that baySeq perfomed well or better than several existing techniques, including EdgeR and DESeq, in identification of pairwise differential expression in count data.

While BaySeq addresses more complex experimental designs, it works well for simple two group comparisons such as *control versus treatment*. Due to its computationally intensive nature, it has been implemented to take advantage of parallel processing techniques. In practice, this requires the use of the R Bioconductor package `snow`.

Chapter 5

Filtering and Normalization

5.1 Normalization

Normalization of RNA-seq data has been shown to be necessary to accurately compare counts between experimental groups [10]. It is a standard implementation of many R packages designed to test for differential expression of transcripts. Normalization is required to adjust for between-sample sources of variation, such as library size. There are several methods currently used to normalize count data and a few are outlined in this section. Our WEC data is used to draw comparisons between different methods used to normalize data.

Normalization has been shown to **stabilize** samples, i.e. transform transcript counts so they are directly comparable between samples [10]. Figure 5.1 shows boxplots of the WEC data before and after applying the UQ method of normalization outlined below. Further summary statistics for this dataset are shown in Section 6.1. The library size for sample $C2$ is approximately 1.8 times that of the others. Normalization scales the log_2 counts in $C2$ down, increasing the interquartile range. For samples across both treatment groups, the log_2 means are pulled towards each other.

Total Count Normalization(TC) To account for difference in library sizes, counts are scaled by a *scaling factor*. The library size, $l'_i = N_{.i}$, is a natural choice of scaling factor. All counts in sample i can be divided by $N_{.i}$ to generate a dataset that is scaled and comparable.

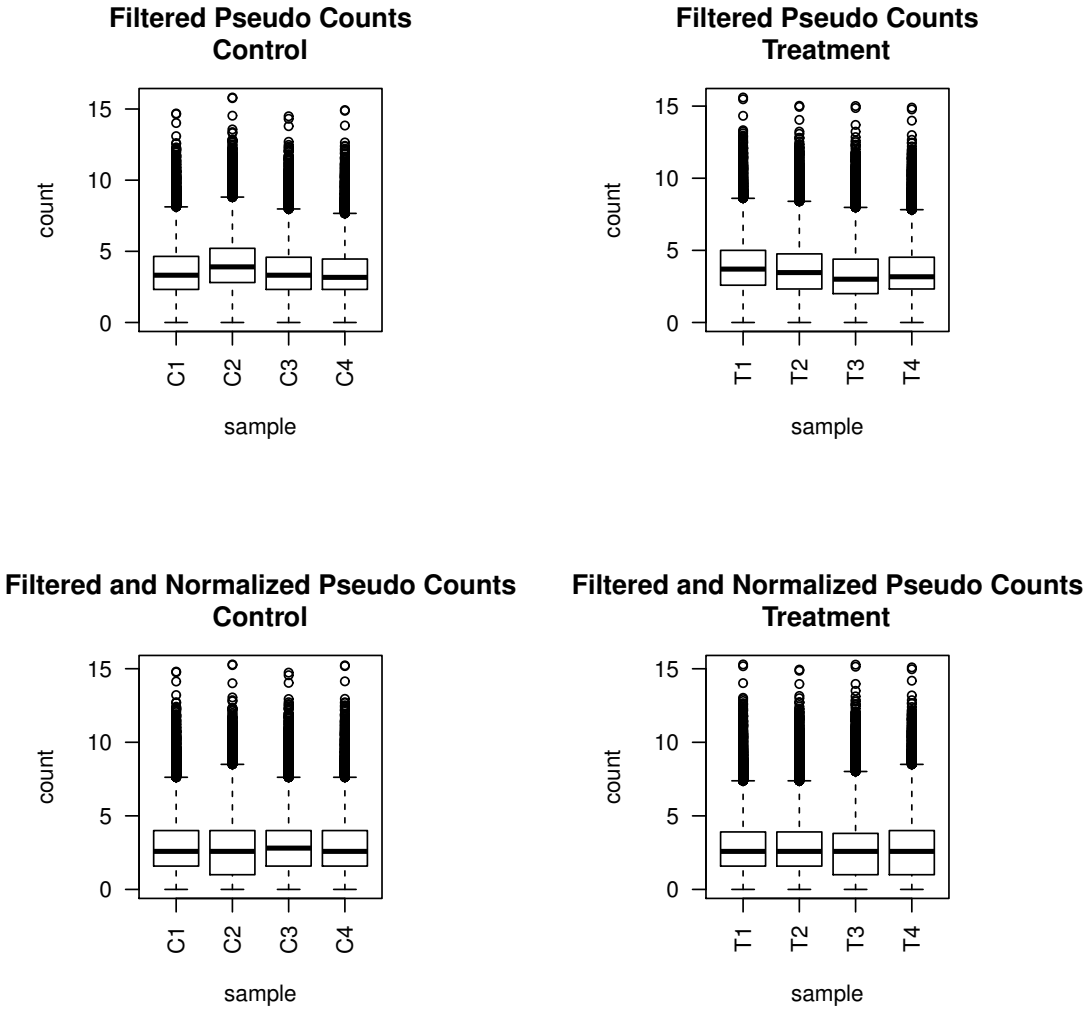

Figure 5.1: Boxplots of log_2 before and after applying UQ normalization for the WEC data.

This method of normalization shrinks the influence of larger samples so that the effects of high count transcripts within those samples are dampened [10].

Total count normalization is a method shown to produce highly inaccurate results [24]. Consider a two-sample experiment with counts N_{g1} and N_{g2} for transcript $g = 1, ..., 101$. Suppose the following:

$$N_{g1} = 100 \qquad \text{for } g = 1, \cdots, 100$$
$$N_{g1} = 0 \qquad \text{for } g = 101$$
$$N_{g2} = 80 \qquad \text{for } g = 1, \cdots, 100$$
$$N_{g2} = 2000 \qquad \text{for } g = 101$$

The library sizes, l_1 and l_2 for both samples are 10000, suggesting that $l_1' = l_2'$ and the counts are directly comparable. Direct comparison suggests the possiblity that all transcripts are differentially expressed. However, by considering the distribution of the entire samples, it is more likely that only transcript 101 is differentially expressed. M.Dillies et al. suggested that TC normalization should not be considered a method of normalization for analysis [10] as it ignores the fact that different biological samples may express RNA at different rates. In addition, it may be biased by the behavior of a small number of high-count transcripts that are expressed at different levels across biological conditions.

Table 5.1 shows a sample of our WEC data. Table 5.2 shows the same data after TC normalization has been applied.

	C1	C2	C3	C4	T1	T2	T3	T4
CCH-0004-C_J3968175	0	0	0	0	0	0	0	0
CCH-0004-C_J3968307	0	0	0	0	0	0	0	4
CCH-0004-C_J3968402	881	1526	1319	1173	1176	1510	1688	1504
CCH-0004-C_J3968416	130	169	110	113	187	136	99	103
CCH-0004-C_J3968457	13	29	15	12	27	23	15	15
CCH-0004-C_J3968510	0	0	0	0	2	0	0	0
CCH-0004-C_J3968589	4	6	6	4	3	4	0	2
CCH-0004-C_J3968598	0	0	0	0	0	2	0	2
CCH-0004-C_J3968657	0	0	0	0	2	0	0	2
CCH-0004-C_J3968658	3	13	10	0	4	4	4	0

Table 5.1: Sample of rawcounts from WEC dataset

	C1	C2	C3	C4	T1	T2	T3	T4
CCH-0004-C_J3968175	0	0	0	0	0	0	0	0
CCH-0004-C_J3968307	0	0	0	0	0	0	0	5
CCH-0004-C_J3968402	951	1075	1543	1434	958	1433	2037	1727
CCH-0004-C_J3968416	140	119	129	138	152	129	119	118
CCH-0004-C_J3968457	14	20	18	15	22	22	18	17
CCH-0004-C_J3968510	0	0	0	0	2	0	0	0
CCH-0004-C_J3968589	4	4	7	5	2	4	0	2
CCH-0004-C_J3968598	0	0	0	0	0	2	0	2
CCH-0004-C_J3968657	0	0	0	0	2	0	0	2
CCH-0004-C_J3968658	3	9	12	0	3	4	5	0

Table 5.2: Normalized data using Total Read Count method

Li et al. [24] use a similar approach by deriving an estimate of the sequencing depth, $\hat{d}_i$, and use it as the scaling factor. In their paper they define $\hat{d}_i = N_{\cdot i}$, and later as $\hat{d}_i = \frac{N_{\cdot i}}{N_{\cdot\cdot}}$. The choice to include $N_{\cdot\cdot}$ in the denominator appears arbitrary and the normalized count is $\hat{d}_i * N_{gi}$.

Adjusted Total Count Normalization (Adjusted TC) Li et al. [24] suggest an adjustment to TC that is the default normalization method used in the R-package PoissonSeq. The method is defined in [24] and is very similar to TC except that the scaling factor is calculated using only a subset, S, of transcript counts for which no differential expression is suspected. The subset of transcripts, S, is identified using a Poisson goodness-of-fit test as follows.

1. Define some $\epsilon \in (0, \frac{1}{2})$ where ϵ is some fixed constant. Li et al. [24] set ϵ to be 0.25
2. Let $\hat{d}_i = \frac{N_{\cdot i}}{N_{\cdot\cdot}}$ be the initial estimate for $\hat{d}_i$
3. Calculate $GOF_g = \sum_{i=1}^{n} \frac{(N_{gi} - \hat{d}_i N_{g\cdot})^2}{\hat{d}_i N_{g\cdot}}$
4. Choose $S = \{g \in S | GOF_g$ is in the $(\epsilon, 1-\epsilon)$ quantile of all GOF_g values$\}$.
5. Calculate $\hat{d}_i = \frac{\sum_{g \in S} N_{gi}}{\sum_{g \in S} N_{g\cdot}}$

6. Iterate over steps 3-5 until $\hat{d}_i$ converges.

Then counts in sample i can be scaled by $\hat{d}_i$ to form a dataset that is comparable between samples.

Upper Quartile Normalization (UQ) Upper Quartile Normalization (UQ) is another method that uses a scale factor and it is the default normalization method in the Bioconductor R package baySeq [17]. This method follows a similar procedure to adjusted TC, instead using the upper quartile of counts to scale the library sizes for each sample. The set, S, of transcripts that are used to derive the scaling factor, $\hat{d}_i$, for each sample i are those in the upper quartile of counts for sample i after 0 counts have been removed.

UQ has been shown to be an effective method for correcting for library sizes, exhibiting similar log_2 expression changes to other methods such as those employed in the edgeR, DeSeq, PoissonSeq and SamSeq packages. [10]. Table 5.3 shows a sample of our WEC data after UQ normalization has been applied.

	C1	C2	C3	C4	T1	T2	T3	T4
CCH-0004-C_J3968175	0	0	0	0	0	0	0	0
CCH-0004-C_J3968307	0	0	0	0	0	0	0	5
CCH-0004-C_J3968402	955	1066	1563	1448	953	1428	2037	1724
CCH-0004-C_J3968416	141	118	130	139	152	129	119	118
CCH-0004-C_J3968457	14	20	18	15	22	22	18	17
CCH-0004-C_J3968510	0	0	0	0	2	0	0	0
CCH-0004-C_J3968589	4	4	7	5	2	4	0	2
CCH-0004-C_J3968598	0	0	0	0	0	2	0	2
CCH-0004-C_J3968657	0	0	0	0	2	0	0	2
CCH-0004-C_J3968658	3	9	12	0	3	4	5	0

Table 5.3: Normalized data using Upper Quartile method

Trimmed Method of Means(TMM) Trimmed Method of Means is the default normalization method implemented in the edgeR package [39]. The TMM method estimates scale

factors between samples using a weighted trimmed mean of the log expression ratios.

Let y_i be the vector of counts for the ith sample, $i = 1, \cdots, n$. Choose one sample, y_r, to be the reference sample. For each $y_i, i \neq r$, and transcript g, define:

$$M_{gi}^r = log_2 \frac{(N_{gi}/N_{.i})}{(N_{gr}/N_{.r})} \tag{5.1}$$

$$A_{gi}^r = \frac{1}{2} \log_2 (N_{gi}/N_{.i} \cdot N_{gr}/N_{.r}) \tag{5.2}$$

$$w_{gi} = \frac{N_{.i} - N_{gi}}{N_{.i} N_{gi}} + \frac{N_{.r} - N_{gr}}{N_{.r} N_{gr}} \tag{5.3}$$

Where M_{ig}^r is the transcript-wise log-fold change between samples i and r, for transcript g. A_{gi}^r is the absolute expression level for transcript g between samples i and r. Additionally, w_{gi} is a weight as the inverse of the approximate asymptotic variance calculated using the delta method.

The upper and lower $x\%$ of M and A values are removed from the dataset before computing a trimmed mean for each sample. In the R package EdgeR [39], the M_{gi}^r values are trimmed by 30% and the A_{gi}^r values by 5% by default, however x can be tailored to the dataset.

The TMM normalization factor, $log_2(TMM_i^r)$, is computed as the weighted mean of M_{gi}^r over G^*, the set of transcripts with valid M_g and A_g values

$$log_2(TMM_i^r) = \frac{\sum_{g \in G^*} w_{gi}^r M_{gi}^r}{\sum_{g \in G^*} w_{gi}^r} \tag{5.4}$$

5.2 Filtering

The number of DE hypothesis tests conducted on a typical RNA-Seq dataset can be very large and many truly null hypotheses may give small P-values by chance. Genes with low

counts across all samples provide little evidence for differential expression. RNA-seq data tends to be comprised of many low counts, thus interfering with the statistical calculations for differential expression.

This has motivated development and testing of several corrective methods including control of the **false discovery rate(FDR)** [4] using a Bonferroni-type procedure. According to Rau et al. these methods applied to multiple testing help control false positives, but may miss true positives [36]. To remedy the problem, the use of data filters can be used to remove the transcripts that provide little to no information for detecting differentially expressed transcripts [5].

Bourgon et al. [5] advocated for the use of independent filtering, which has been further developed in work such as from Kim and Schliekelman [19]. Independent filtering correlates a filter criterion with a test statistic, or p-value. Optimization of this filter leads to maximizing the number of true effects discovered. More is discussed about this methodology later in this section.

Several other types of data filters for RNA-seq data are sometimes used, including filtering transcripts with a total read count smaller than a given threshold and filtering transcripts with at least one zero count in each experimental condition. However, selecting an arbitrary threshold needs to account for the overall sequencing depth and variability of a given experiment. Transforming counts to RPKM (Reads per Kilobase per Million) or CPM (Counts per Million) can effectively account for sequencing depth.

According to [36], filters for read counts are routinely used in practice with little attention paid to the type of filter used or cut-off threshold value. These choices can impact downstream analysis and therefore should be considered.

Counts per Million (CPM) Data Filter CPM is the filtering method used in the edgeR analysis pipeline, found in EdgeR User's Guide. We apply it to our real and simulated datasets in this study. Transcripts whose transformed counts are larger than some cut-off value, c, in at least the minimum number of samples in a group, over all groups, are retained for analysis. The method is as follows:

Let N_{gi} be the raw count from transcript g in sample i, and $l_i = \sum_{g=1}^{p} N_{gi}$ be the library size for sample i. Then $CPM_{gi} = 1000000 * N_{gi}/l_i$ is the transformed **counts per million**

(cpm) value for transcript g in sample i.

For a K-group experiment, define m_k to be the number of samples in the k^{th} group. Order the m_k's, $m_{(1)} < \cdots < m_{(K)}$. Select S, the set of transcripts whose $\text{CPM}_{gi} > c$ in at least $m_{(1)}$ samples. Those transcripts not in S are filtered from further analysis.

In R, the code would look like

```
keep <- rowSums(cpm(y)>1) >= 2,
```

and in practice, c is some small, arbitrarily choosen value [13] [14]. Little research has been done on the best choice for c, although it has been shown that choosing a good filter cut-off substantially increased the number of null hypotheses rejected [5].

Jacaard Filter Typically, arbitrary cut-off thresholds are chosen with little consideration placed on the impact on the downstream analysis [36]. To address this issue, Rau, Gallopin, Celeux et al. [36] developed a data-driven threshold based on defining a threshold that maximizes the similarity between replicates. They call this method *Jacaard Filtering*. The proposed filtering method is implemented in the R Bioconductor package, HTSFilter [34] [36].

The value for the filter cut-off, c, is decided by maximizing the filtering similarity among replicates. A series of Jaccard indices is calculated for each group and the maximum of these is selected as the cut-off.

The method is as follows. Let N_{gi} be the observed normalized read count for transcripts g, $g = 1, ..., p$, in sample i, $i = 1, ..., n$, belonging to experimental group $C(k)$. Denote the vector of read counts for sample i as y_i. The data is binarized for a fixed cut-off s such that

$$y_g i = \begin{cases} 1 & \text{if } y_{gi} > s \\ 0 & \text{otherwise} \end{cases}$$

Then for some threshold, s, the Jacaard index for samples i and i' from the same experimental group is defined as:

$$J_s(y_i, y_{i'}) = \frac{a}{a+b+c} \tag{5.5}$$

where the numerical values for a, b, and c are defined in Table 5.4.

	sample i'	
sample i	# Normalized counts > s	# Normalized counts ≤ s
# Normalized counts > s	a	b
# Normalized counts ≤ s	c	d

Table 5.4: Table of frequencies used to compute the Jaccard Index Coefficient. a represents the number of transcripts with normalized counts greater than s in both samples.

$J_s(y_i, y_{i'})$ ranges from 0 (dissimilar) to 1 (similar). The definition of the Jaccard index can be extended to account for multiple samples. A global Jaccard index is produced by taking the mean of all indices calculated over all pairs in each condition (Equation 5.6).

$$J_s^*(y) = \text{mean}\Big\{J_s(y_i, y_{i'}) : i < i' \text{ and } C(i) = C(i')\Big\} \tag{5.6}$$

The cut-off, $s*$ is defined in 5.7 as the value of s that corresponds to the maximum value of the global Jaccard index and represents the greatest similarity possible among replicates:

$$s^* = argmax_s J_s^*(y) \tag{5.7}$$

In practice, the value of the global Jaccard index in Equation 5.6 is calculated for a set of threshold values and fit to a loess curve [7]. s^* is then the maximum of these fitted values.

Independent Filtering In a 2010 study, Bourgon et al. recommended the use of independent data filtering, in which the filter and subsequent test statistic pairs are independent under the null hypothesis. The authors showed that the filter must be positively correlated with the test statistic under the alternative hypothesis in order to be effective. The key requirement for maintaining Type I error control is for the null hypothesis distribution prior to filtering be the same as the null hypothesis distribution after filtering.

According to Kim and Schliekelman, at the time of publication of their study [19] in 2015, little was known about the conditions required for filtering to successfully increase discovery probabilities. They introduced a framework that is able to quantify filter effectiveness in terms of the probability of a feature associated with a true effect being ranked at or above a specific quantile. They examined the conditions under which filtering is successful.

Their study was partially weighted towards datasets in which there were redundancy between hypothesis tests as is found in Quantitative Trait Loci (QTL) mapping studies. In testing differential expression of transcripts in RNA-seq data, there is no redundancy in testing as each transcript requires a single test of DE versus non-DE. However, Kim and Schliekelman found that in cases of low redundancy between hypothesis tests, the benefits of filtering decrease rapidly as the quality of filter information decreases. They found that the gain from filtering was highly dependent on the choice of filter cut-off, increasing discovery probability as much as 2-3 fold. They found that naively choosing a filter cut-off based on the data potentially inflated the Family-Wise Error Rate (FWER).

It was found that filtering could greatly increase the relative discovery probablity for weak effects, but power gains were limited. In practice, some studies search for large numbers of low effects, while others are interested in finding a few dominant genetic variants [19]. They recommended further reasearch be done in the area to determine where good cut-off points tended to be for different types of data.

Comparison of Jacaard and CPM Filtering Methods Rau et al [36] ran a study to compare the performance of their Jacaard filter against that of other filtering methods, including the CPM filtering method discussed above.

Figure 5.2, originally published in [36], shows results from plotting the *log* fold change against *log* base mean of counts filtered using these filtering methods. From the plot it can be seen that only the CPM method and Jaccard successfully filter the transcripts with small log fold changes and low levels of expression.

Figure 5.3, which was originally published in [36], show Receiver Operating Characteristics (ROC) curves for each filter. The ROC curves were averaged over 300 simulated datasets and show the comparative filtering performance of several types of filters. ROC curves measure **filtering sensitivity** against 1-**filtering specificity**. Filtering sensitivity is defined as

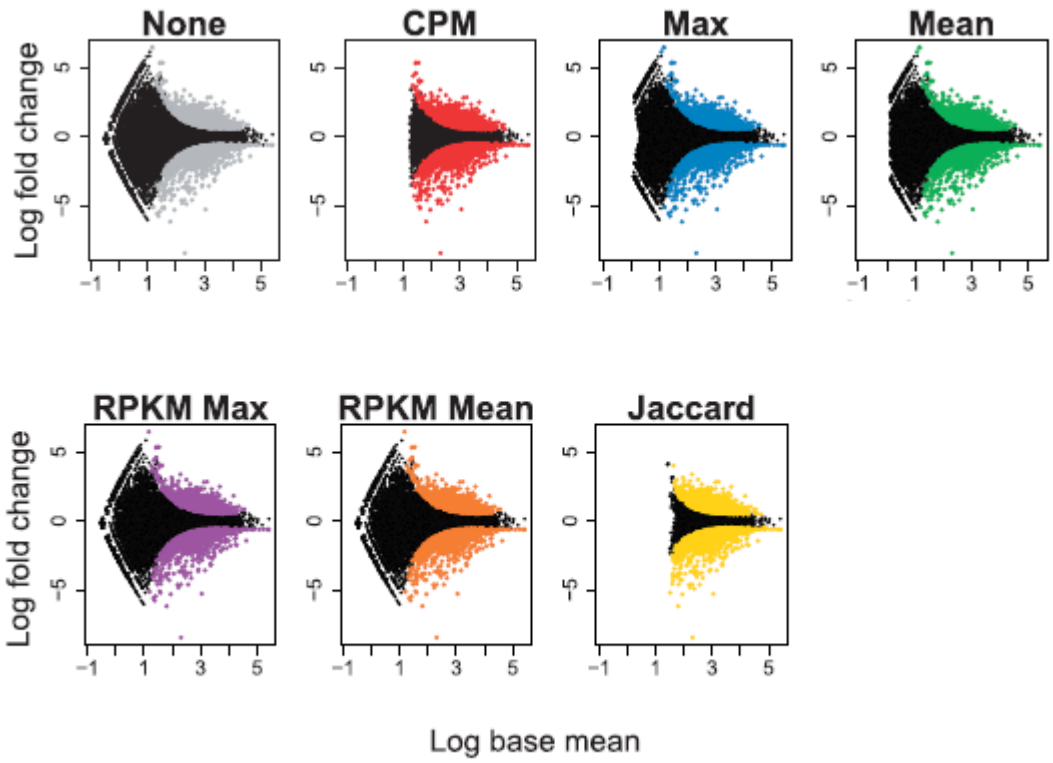

Figure 5.2: Log mean expression versus log fold change values for a real RNA-seq dataset. For each filter, transcripts identified as non-DE are drawn in black, and filtered transcripts are omitted from the plot. From left to right, the filters are: none, CPM, maximum, mean, RPKM maximum, RPKM mean and maximum using the global Jaccard index threshold. This figure originally appeared in [36].

the proportion of correctly unfiltered genes (i.e. DE and unfiltered) among all truly DE genes, and the filtering specificity is defined as the proportion of correctly filtered genes (i.e. non-DE and filtered) among all non-DE genes [36].

Figure 5.3 demonstrates that the maximum-based filters that include either the Jaccard method or the CPM method perform better than the other approaches. ROC curves (averaged over 300 datasets) for the filtering performance on a real RNA-seq dataset for the CPM, maximum, mean, maximum RPKM and mean RPKM filters over a range of cut-offs. The yellow cross labelled with a J corresponds to the filtering sensitivity and specificity for the data-based threshold chosen via the global Jaccard index.

Rau et al. [36] showed that overall, the Jaccard method ranked slightly better than the CPM method, which is the default method used in edgeR. However, compared to the Jaccard maximum-based filtering, CPM filtering is computationally more convenient while its performance is close to that of Jaccard.

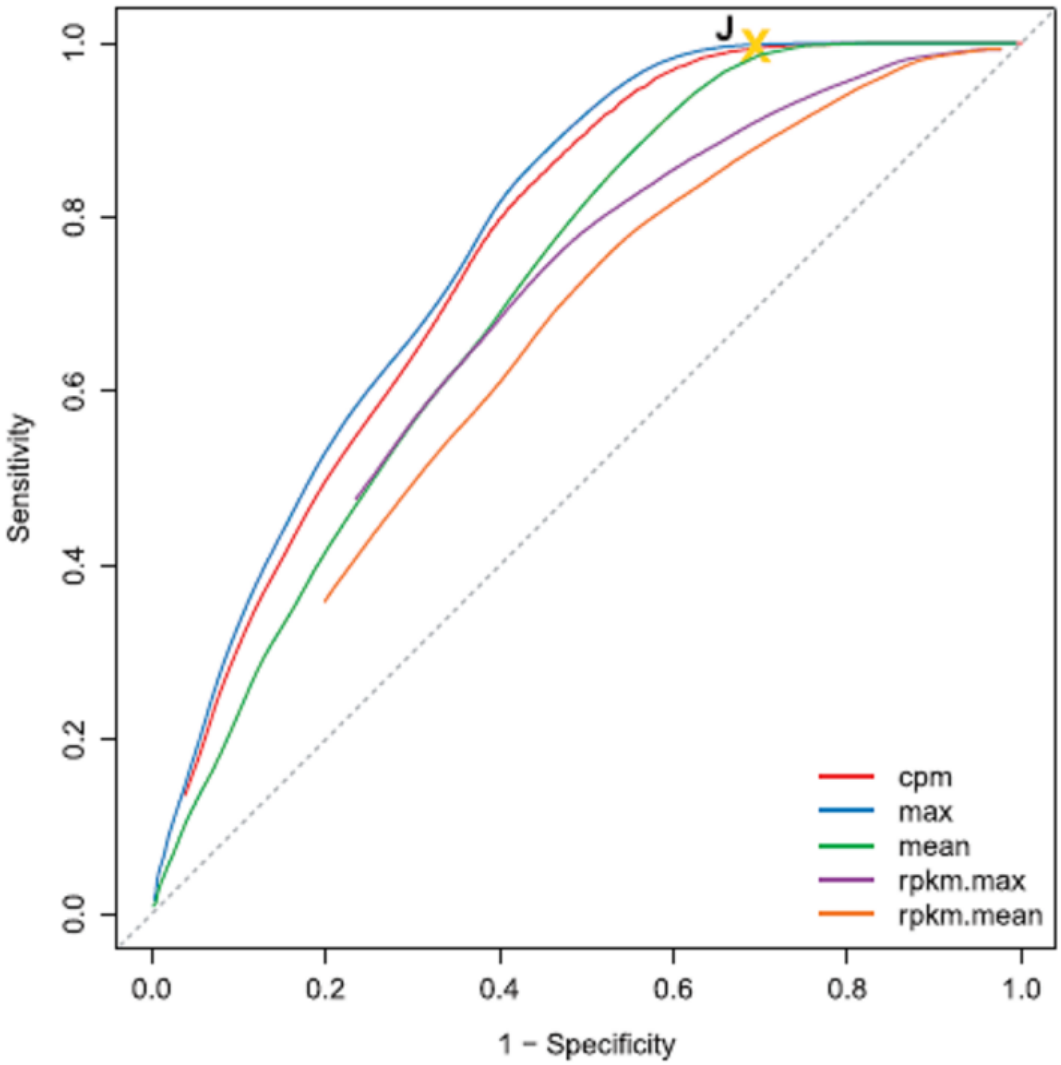

Figure 5.3: ROC curves (averaged over 300 datasets) for the filtering performance on a real RNA-seq dataset for the CPM, maximum, mean, maximum RPKM and mean RPKM filters over a range of cut-offs. The yellow cross labeled with a J corresponds to the filtering sensitivity and specificity for the data-based threshold chosen via the global Jaccard index. This figure originally appeared in [36].

Due to the success shown in [36], CPM filtering was chosen to filter our WEC data prior

to analysis using all packages. As per the CPM method, transcripts with CPM values greater than two in least four samples were kept in the dataset. The cut-off of two was arbitrarily chosen. The resulting dataset was used in the downstream analysis.

Chapter 6

WEC Data Analysis

6.1 Data Quality and Exploration

6.1.1 Distribution of Dataset

The pre-filtered raw counts of the WEC dataset consists of 49166 transcripts and eight replicates over two groups, four in each of control and treatment. Tables 6.1 and 6.2 provide descriptive statistics about the samples. From these tables, it can be seen that the library size, or total read count, for each sample varied between 1.0 and 1.8 million reads. After filtering and normalization, the library size for each sample was approximately 1.26 million reads. Maximum counts in each library varied between 27000 and 40000 reads, with the mean read count being approximately 28. Not surprisingly, the smallest counts for each sample were zero, and in many cases these were filtered out prior to analysis. Filtering methods are discussed in Chapter 5.

Counts were normalized using **upper quantile** (UQ) normalization. Low count transcripts were filtered using the CPM method as described in Chapter 5. As per our CPM method, transcripts with cpm values > 2 in at least four replicates were kept for further analysis, all others were filtered out of the dataset. Normalization was applied before the filtering step for the descriptive analysis, unless otherwise stated.

As seen in Figure 6.1, the rawcount distribution (prior to normalization and filtering)

	C1	C2	C3	C4
library size (raw)	1179381.00	1806953.00	1087840.00	1041466.00
library size	1269548.00	1253664.00	1278234.00	1275453.00
# of transcripts (raw)	65499.00	65499.00	65499.00	65499.00
# of transcripts	44566.00	44566.00	44566.00	44566.00
mean count	28.49	28.13	28.68	28.62
median count	5.00	5.00	6.00	5.00
minimum count	0.00	0.00	0.00	0.00
maximum count	28574.00	40083.00	27258.00	38382.00

Table 6.1: Summary of WEC Data control group. All summary statistics are taken after normalization then filtering, unless specified as raw.

	T1	T2	T3	T4
library size (raw)	1562700.00	1341797.00	1054955.00	1108388.00
library size	1259763.00	1264384.00	1263963.00	1261053.00
# of transcripts (raw)	65499.00	65499.00	65499.00	65499.00
# of transcripts	44566.00	44566.00	44566.00	44566.00
mean count	28.27	28.37	28.36	28.30
median count	5.00	5.00	5.00	5.00
minimum count	0.00	0.00	0.00	0.00
maximum count	40150.00	31449.00	39733.00	34937.00

Table 6.2: Summary of WEC Data treatment group. All summary statistics are taken after normalizating then filtering, unless specified as raw.

is highly skewed due to the large number of zero-count transcripts. This is typical with RNA-seq data, therefore it can be more helpful to work with transformed data. A natural transformation is log base 2 where a difference of one on the log_2 scale represents a fold change of two on the original count scale. To allow zero count reads to be included in the log_2 function, a value of one is added to all counts prior to transformation. From here forward, we refer to the log_2(count+1) values as **pseudo counts**.

Figure 6.2 shows the frequency distribution of the median transcript pseudo counts for each experimental group after normalization and filtering. Most transcripts have mean pseudo counts of 5 or less and there appear to be few transcripts with median pseudo counts higher then 8. This can be seen in Tables 6.1 and 6.2 where normalized and filtered mean counts are, on average, slightly less than 2^5.

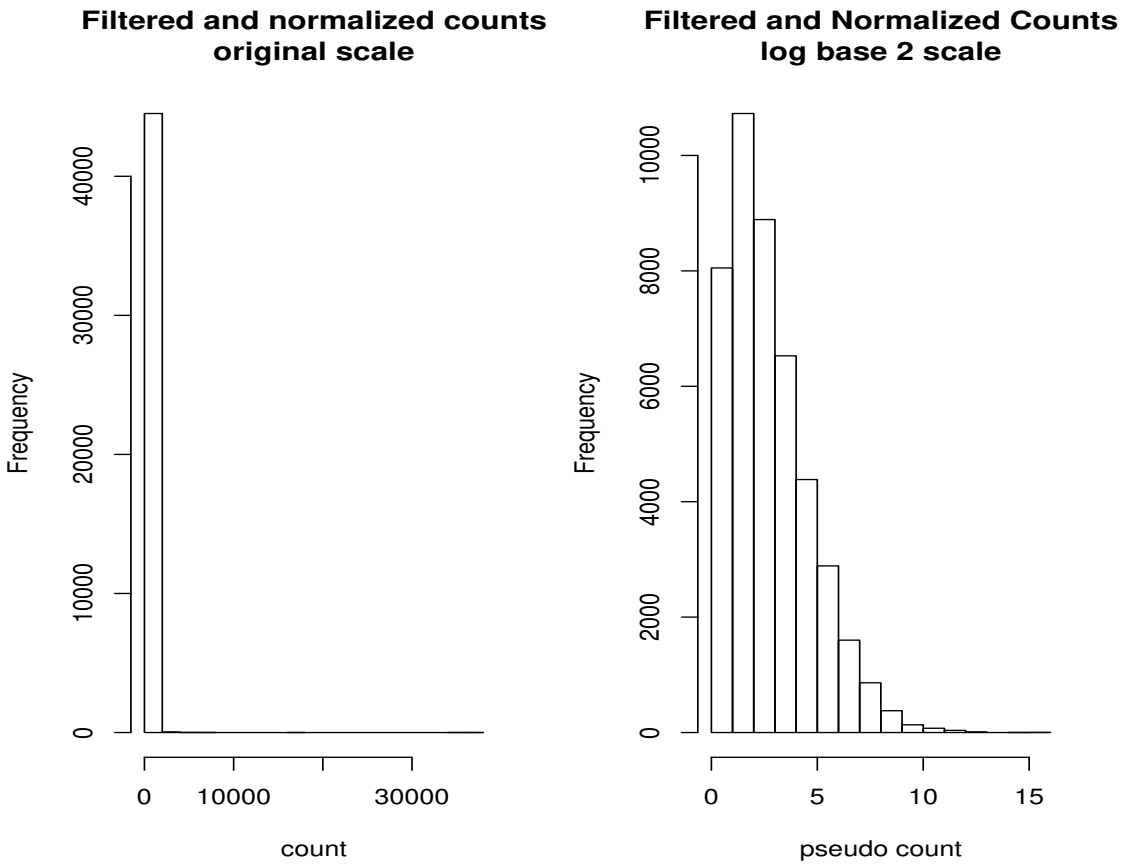

Figure 6.1: Distribution of counts. Data are normalized by UQ normalization and filtered by CPM method.

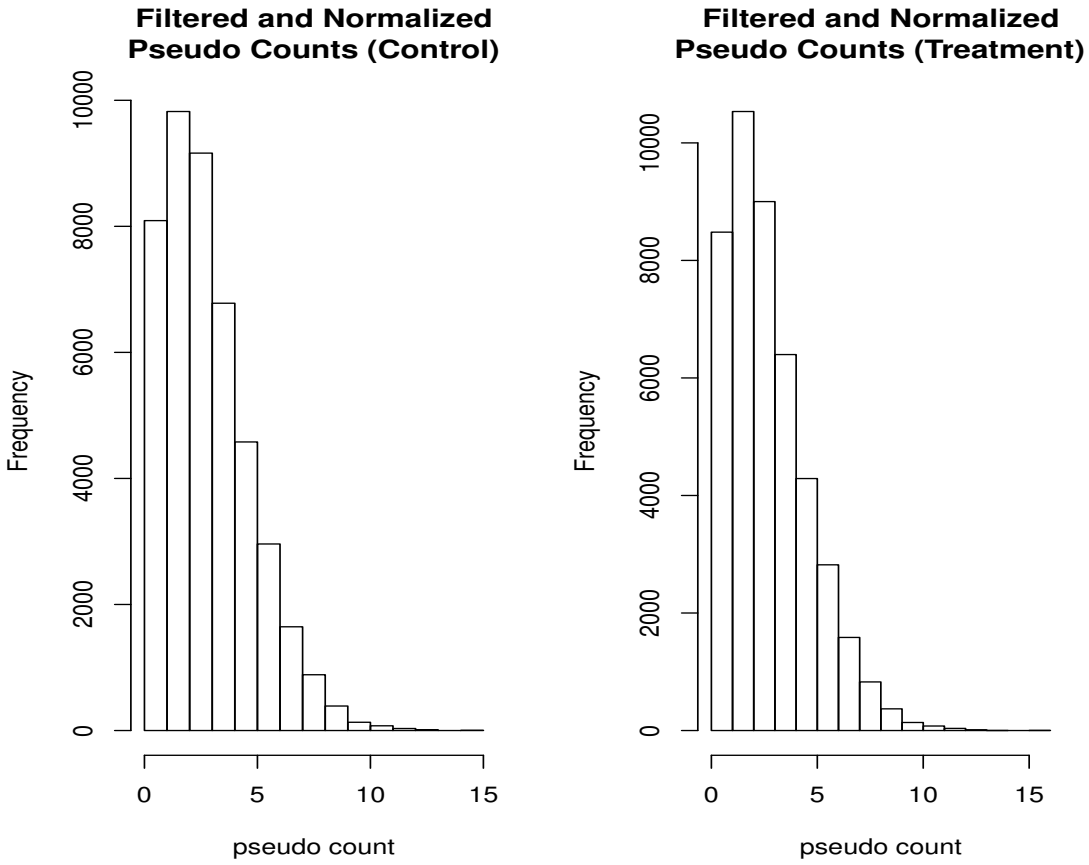

Figure 6.2: Distribution of median psuedo-counts, taken across each treatment group. Data are normalized by UQ normalization and filtered by CPM method.

6.1.2 Boxplots

MA plots, Figure 6.5, and boxplots, Figure 6.3, are used to visualize the between-sample distribution of counts. They highlight contrast in the read count distribution between samples, and are useful in assessing replicate quality and revealing outlier counts. As well, the effects of filtering and/or normalization can be observed. Some datasets can be particularly sensitive to the normalization procedure, especially those with many transcripts exhibiting high abundance.

Rapaport [35] uses boxplots to show how the normalization methods of several R packages produces comparable distributions after transformation. Replicates with high overall expression after normalization should be identified. More details regarding normalization and filtering are in Chapter 5.

Boxplots of the unnormalized and UQ normalized pseudo counts for our WEC data are shown in Figure 6.3. It can be seen that normalization has helped stabilize the samples by removing many of the low count transcripts and scaling those remaining. Thus samples within an experimental group have comparable distributions.

Samples that contain many transcripts with high relative expression rates (after normalization) should be examined further as this can be indicative of outlier samples. Yet, it should be noted that outlier samples will not necessarily be identified through box plots alone if the plots are constructed with normalized data.

In this paper we provide an alternative by examining the frequency of dominant transcripts. We define a dominant transcript to be one whose read count is greater than 50% of the group total. Consider N_{gi1}, the normalized read count from transcript g in sample i, for all i in the first treatment group. Define $N_{g.1}$, the sum of N_{gi1} for all i in treatment group 1. We define the indicator variable I_{gi} for the normalized read counts from samples i and transcript g in group 1 to be

$$I_{gi} = \begin{cases} 0 & \text{if } \frac{N_{gi1}}{N_{g.1}} < .5 \\ 1 & \text{if } \frac{N_{gi1}}{N_{g.1}} \geq .5 \end{cases}$$

Similarly, I_{gi2} can be defined for samples in group 2 and so on.

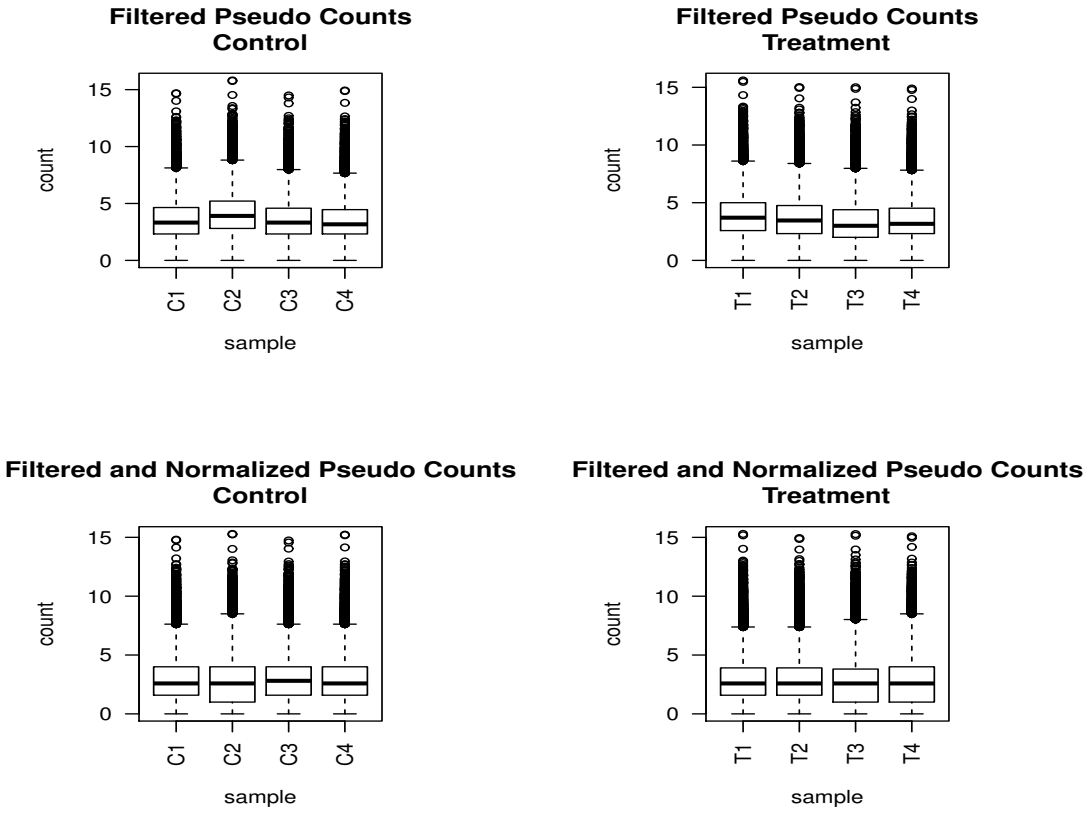

Figure 6.3: Boxplots of CPM filtered pseudo counts (top) and UQ normalized then CPM filtered pseudo counts (bottom).

The proportion of dominant counts, $\frac{\sum_g I_{gi}}{p}$, for each sample is shown in Figure 6.4. Samples tend to exhibit dominant relative expression in 10% to 15% of transcripts, suggesting no replicate appears to be an outlier sample.

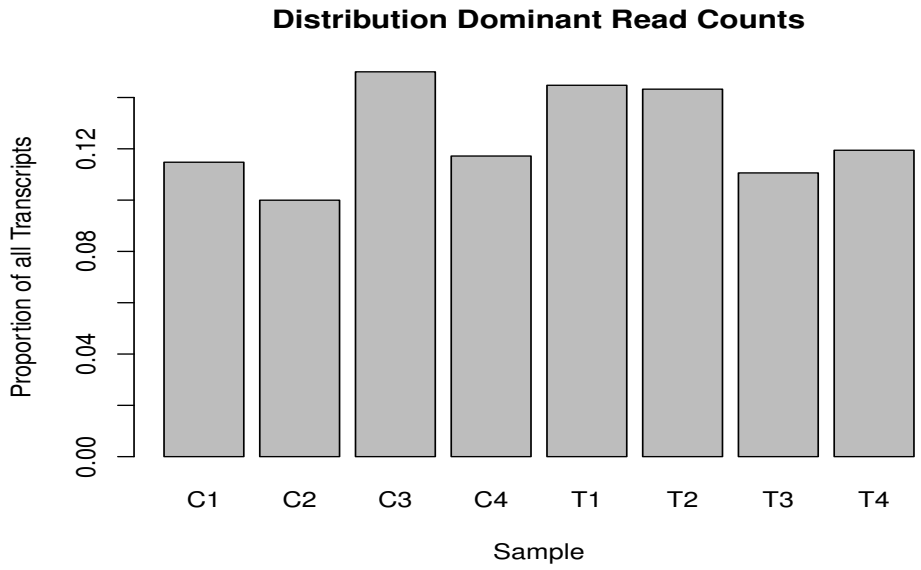

Figure 6.4: Proportion of dominant transcripts by treatment group. A dominant transcript is one whose read count comprises over 50% of the group total. Data is UQ normalized and filtered.

6.1.3 MA Plots

An MA (Mean-Average), or MD (Mean-Difference) plot, is a scatter plot that compares log_2 ratios (M) to log_2 averages (A) for two samples of data, where M and A are defined as follows. For some transcript g, $g = 1, \cdots, p$, let x_g be the count from sample x, and y_g be the count from sample y. Then $M_g = \log_2(x_g/y_g)$ and $A_g = 0.5\log_2(x_g \cdot y_g)$. Figure 6.5 shows a variation of a MA plot using UQ normalized and filtered data, in which each sample is compared to the average of remaining samples. That is, if x_g is the read count for transcript g from sample x, then y_g is the read count for transcript g averaged over all samples excluding sample x.

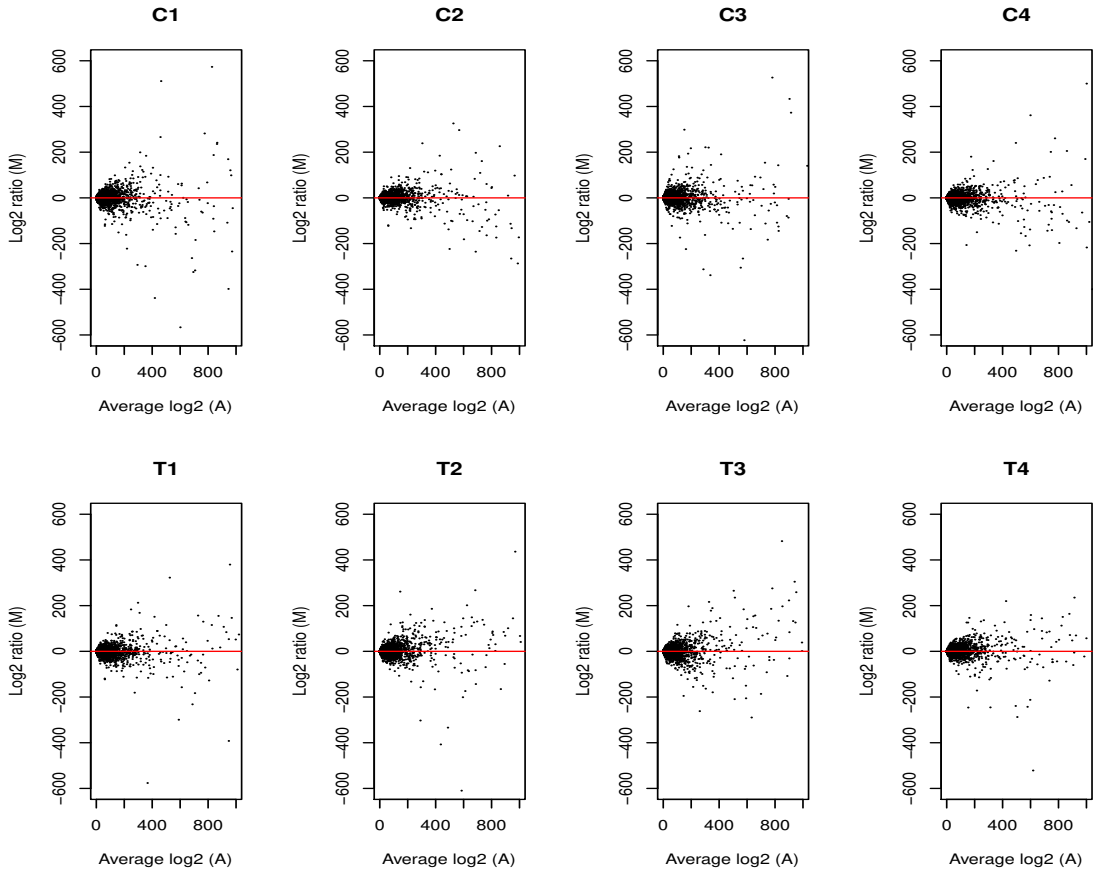

Figure 6.5: MA plots comparing each sample to all other samples. Data is unnormalized, unfiltered counts.

MA plots have been used extensively for analyzing microarray data, and are helpful to visualize the relationship between samples in RNAseq data [20]. Prior to analysis, MA plots can be used to assess sample quality and look for trends in the bias related to the mean expression. If it is assumed that a dataset exhibits equal relative expression rates for up and down regulated transcripts, then one can expect to see the plots exhibit symmetry around the line $y = 0$, and plots for all samples should be roughly the same shape.

The funnel shape is the result of the vastly different log ratios occurring due to read error and abundance. Technical error associated with reads is fairly consistent across transcripts, regardless of their read count, but should still be considered [12]. For example, consider transcripts a and b with average group 1 read counts of $N_a = 1000$ and $N_b = 30000$. Assume the average log ratios between groups 1 and 2 for both transcripts equals 1.5. If there is

read error of ± 100, then the range for the average log ratio for a is $[log\frac{1400}{1100}, log\frac{1600}{900}]$ or $[0.140, 0.249]$. Similarly, the range for transcript b is $[log\frac{44900}{30100}, log\frac{45100}{29900}]$ or $[0.173, 0.178]$ [12].

MA plots show clustering around $y = 0$, as the majority of transcripts in an RNAseq dataset tend not to be differentially expressed. Post-analysis results can be added by marking differentially expressed transcripts in red. Differentially expressed transcripts tend to be seen on the exterior of the funnel. Transcripts with the higher counts, in general, exhibit lower log ratios due to the phenomenon described above.

In Figure 6.5, an MA plot is produced for each sample using the raw count data. On the y-axis is plotted the log_2 count ratios of the sample of interest over the mean of all other samples. On the x-axis is the log fold change between each sample's raw counts and the average raw counts of the others. In Figure 6.5, the shape of the graph is fanned out to the left suggesting the majority of transcripts will have low average counts, which is to be expected. Samples appear well balanced around the line $y = 0$, with the exception of C2 and T1 which appear to be shifted above $y = 0$. This suggests C2 and T1 have higher relative expression rates. However, Tables 6.1 and 6.2 suggest this is due to the larger library sizes of these two samples. Reconstructing the MA plot after normalization shows more balanced plots (Figure 6.6).

When used post analysis, differential expression results can be incorporated into the MA plot and used to examine the quality of differential expression analysis. Deseq2, EdgeR and BaySeq packages have their own versions of this type of MA plot in which transcripts with p-values or likelihoods above a given cut-off are plotted in red.

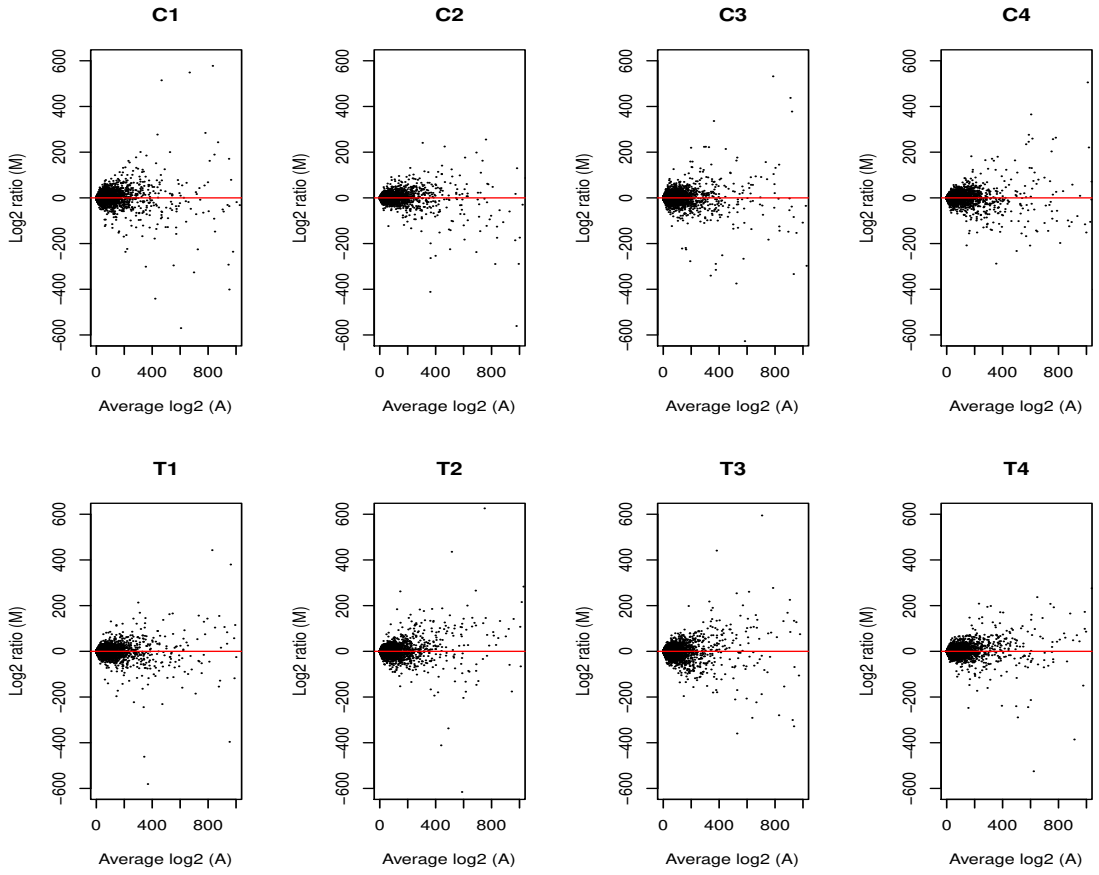

Figure 6.6: MA plots comparing each sample to all other samples. Data is UQ normalized, CPM filtered counts.

6.1.4 Mean-Variance

Plotting the mean variance relationship of transcripts within a group can provide information about the underlying sample distribution as well as indicate how much variation exists between biological replicates. It has been suggested that RNA-seq counts are Poisson-distributed [29], therefore we would expect that the mean and variance will be approximately equal for purely technical replicates.

Figure 6.7 has been constructed after log-transforming the UQ normalized, CPM filtered counts. The $log2$ of the sample variances across treatment groups is shown plotted against the $log2$ of the sample means. It can be seen from this plot that the points tend to lie between the lines $y = x$ and $y = 2x$, suggesting the variance is larger than the mean. This would

likely be caused by the natural variability in expression levels of transcripts that arises due to biological replication. The counts may be more accurately modelled by an over-dispersed Poisson or negative binomial model with parameters μ and dispersion parameter ϕ.

It is shown in Appendix C that Var $(X) = \frac{\mu}{1-\phi}$ where ϕ takes a value between 0 and 1, therefore the variance is larger than the mean.

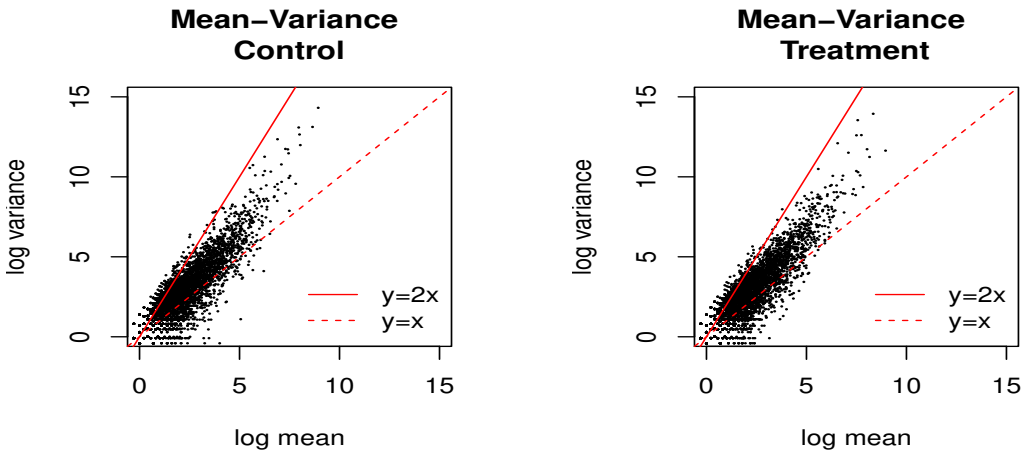

Figure 6.7: Log_2 of the sample variances vs log_2 of the sample means for the control (left) and treatment (right) groups. Data is UQ normalized then CPM filtered before computing the sample variances and means.

6.1.5 Between Group Plots and LogFold Change

Line plots in Figure 6.8 of the UQ normalized and CPM filtered pseudo counts reveal several highly expressed transcripts, which is typical for RNA-seq data. Comparing the control and treatment groups there appear to be few differentially expressed transcripts.

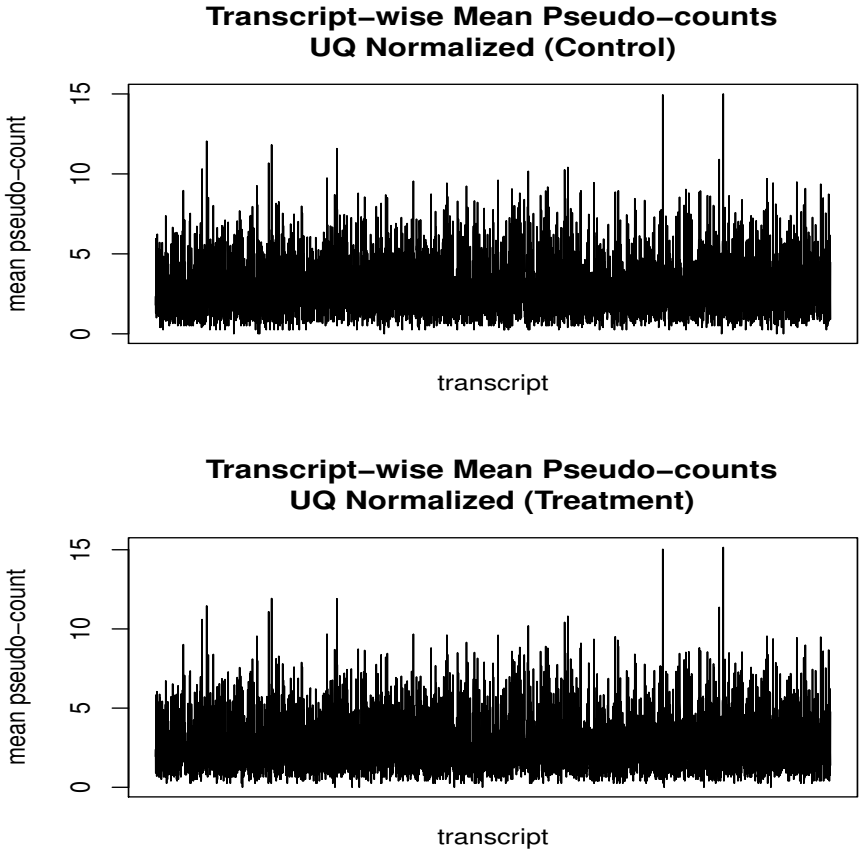

Figure 6.8: Transcript-wise mean pseudo counts of UQ normalized then CPM filtered data.

Figure 6.9 shows the difference in group mean pseudo counts for the UQ normalized then CPM filtered data. Some differences are quite large and can dominate the dataset. Transcripts with high difference in group pseudo counts (>2.7) are shown in Table 6.3.

Another way to visualize the log fold change is by plotting the frequency of **relative log expression (RLE)** values (Figure 6.10). The RLE is the computed log_2 ratio of mean counts between treatment groups. Figure 6.10 is a frequency distribution of the RLE's of

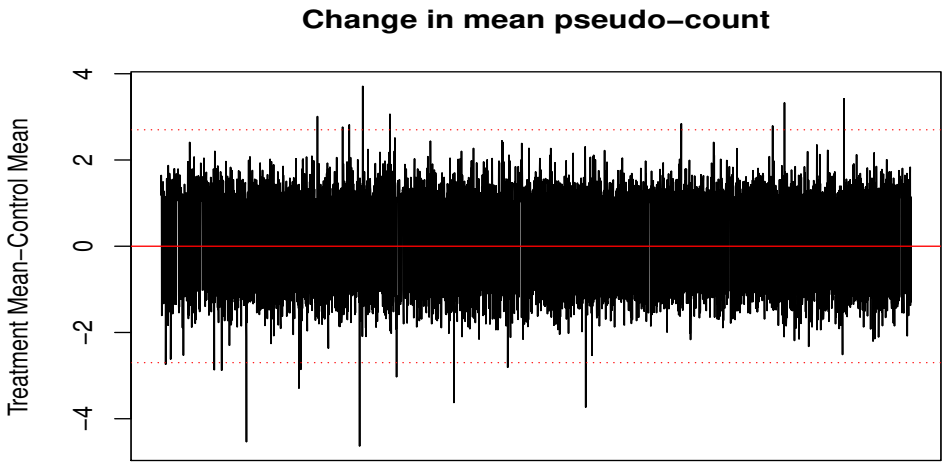

Figure 6.9: Difference in group mean pseudo counts (mean control - mean treatment). Data is UQ normalized then CPM filtered before converting to pseudo counts.

the UQ normalized and CPM filtered WEC data.

$$RLE_g = log_2(\overline{N_{gk_2}}) - log_2(\overline{N_{gk_1}}), \text{where } k_i \text{ consists of all samples in treatment group } i. \tag{6.1}$$

Figure 6.10 shows the balance between up and down-regulated differential expression and is roughly normally distributed. There is a large spike at 0 and cut points [-0.5, 0.5] are drawn as an indicator of approximate differential expression [12]. The ratio of expected values for the two groups should be close to 0 if non-differential expression is suspected. Given this criteria, there appear to be relatively few DE transcripts.

	Mean_Class_1	Mean_Class_2	Difference
CCH-0004-C_S4122880	2.58	7.22	-4.63
CCH-0004-C_S3895400	1.88	6.42	-4.54
CCH-0005-C_S2851150	2.10	5.84	-3.73
CCH-0004-T_S3279015	0.40	4.02	-3.63
CCH-0004-C_S4043614	1.54	4.84	-3.29
CCH-0004-T_J3274710	0.93	3.96	-3.03
CCH-0004-C_R4124368	1.36	4.23	-2.88
CCH-0004-C_R4075426	0.65	3.51	-2.87
CCH-0004-C_S4047390	1.79	4.65	-2.86
CCH-0005-C_J2768708	3.88	6.69	-2.81
CCH-0004-C_J3987520	1.41	4.15	-2.74
CCH-0004-C_S4104120	3.16	0.40	2.76
CCH-0005-T_S2754924	2.79	0.00	2.79
CCH-0004-C_S4112016	3.21	0.40	2.82
CCH-0005-T_R5950008	3.99	1.15	2.84
CCH-0004-C_S4070101	3.01	0.00	3.01
CCH-0004-T_J3241882	6.36	3.30	3.06
CCH-0005-T_S5666812	4.57	1.24	3.33
CCH-0005-T_S5931640	3.82	0.40	3.42
CCH-0004-C_S4127351	6.33	2.62	3.71

Table 6.3: Transcripts with average logfold greater than 2.7 across group samples. Data UQ filtered and normalized with CPM method

Pairs plots or matrix plots are used to help visualize the relationships between samples. Those points that fall to the one side of the diagonal represent transcripts whose read counts are different between samples. To reduce the filesize of Figure 6.11, the pairs plot was constructed using a sample of 15000 counts. The y and x-axis were set to range from $(0, \cdots, 500)$, eliminating more highly expressed transcripts.

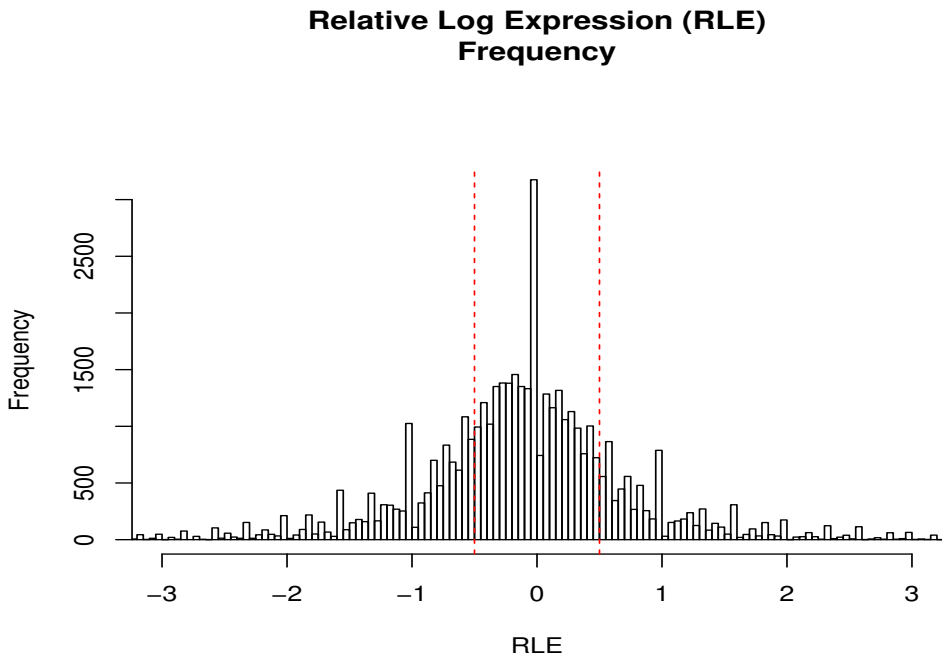

Figure 6.10: Frequency of log_2 mean ratios beween treatment groups (**RLE**). Many transcripts exhibit an RLE of zero. Data is UQ normalized then CPM filtered. Zero rowmeans do not appear in the plot.

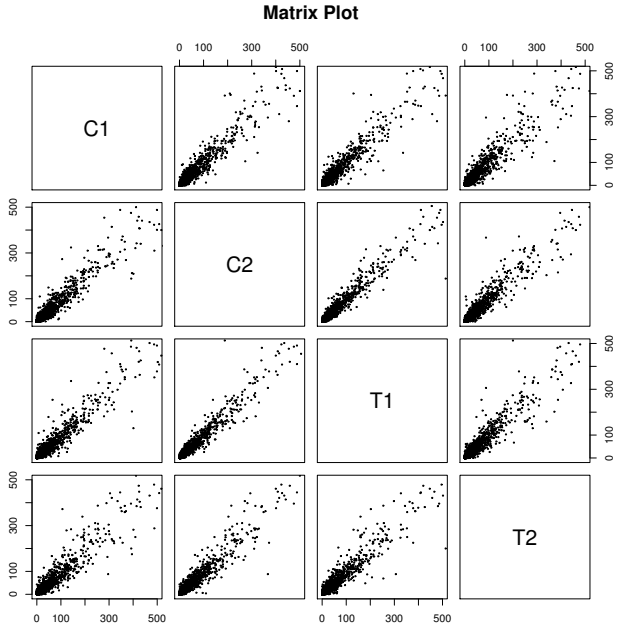

Figure 6.11: Matrix plots of UQ normalized then CPM filtered counts for two samples from each treatment group.

6.1.6 PCA Plots

A significant problem with visualizing RNA sequence data lies in the large number of dimensions that require examination. Our WEC data, for example, has over 40,000 transcripts. A technique called Principal Component Analysis can be applied to reduce the number of dimensions and therefore decrease the complexity of the problem.

The goal of PCA is to calculate a new set of coordinates on which to plot the data. First, the eigenvectors of the covariance or correlation matrix, $\mathbf{A}$, are calculated. An eigenvector, $\mathbf{z}$ of matrix $\mathbf{A}$ is defined to be a vector that satisfies the following: $\mathbf{Az}=\lambda\mathbf{z}$, for some scaler, λ. λ is called the **eigenvalue** and each eigenvector has a related eigenvalue.

If one draws a hypothetical ellipsoid around the cluster of data points, the eigenvectors of the covariance matrix fall along the main axes of this ellipsoid. The absolute values of the eigenvalues indicate how the data is distributed along these axis. Eigenvalues measure the variability retained by each PC. The eigenvalue with the largest value is associated with the eigenvector along which the data exhibits the largest variance, followed by the vector with the second largest eigenvalue, etc. The variance in the data tends to be captured by only a few eigenvectors.

Multiplying the data by the matrix of eigenvectors gives a new projection of the data called **principal components**. The first principal component explains the majority of the variation in the data.

The goal of PCA is to obtain insight into the general structure of a dataset. As RNA-seq data tends to be right-skewed it is good practice to apply some form of transformation to obtain a more normal distribution for the data, prior to performing PCA. Yet another problem inherent to RNA-seq data is that weakly expressed transcripts seem to show much stronger log fold change differences than strongly expressed transcripts. The latter issue is a direct consequence of dealing with count data, in which ratios are inherently noisier when counts are low. This heteroskedasticity (variance of log fold change depending on mean count) complicates PCA [27].

Incorporated into the R Bioconductor [34] package DESeq2 [26] is `rlog`, a variance stabilizing function. For high counts, the resulting transformation mimicks a log_2 transformation. For low counts, the function shrinks the counts towards the transcript's mean using an

empirical Bayesian prior in the form a ridge penalty.

Figure 6.12 shows the result of PCA performed on filtered and normalized WEC transcripts. As per the method in DESEq2's `plotPCA` function, a subset of 500 transcripts exhibiting the highest overall variance were chosen and then transformed using the `rlog` function. The row-wise variances for each transcript were calculated using the `rowVars` function from the R package `genefilter` [16]. The data were then centered and scaled to account for different library sizes.

PCA is a linear transformation that is intended to capture the greatest variability between samples. Using only the 500 most variable transcripts will have a larger effect on the result than using the second 500 most variable transcripts. This method allows for a tradeoff between computational efficiency and getting the most information out of the data.

The first principal component (PC1) plotted along the x-axis explains 27.2% of the variation seen in the data. As significant difference in expression rates between treatment groups is expected, the plot should show clustering of the samples into treatment groups. In Figure 6.12 we do not see the expected clustering in the data. As the subset of data used for this PCA should have captured much of the overall variation seen in the entire dataset, this lack of clustering may be indicative of a larger problem with sample quality.

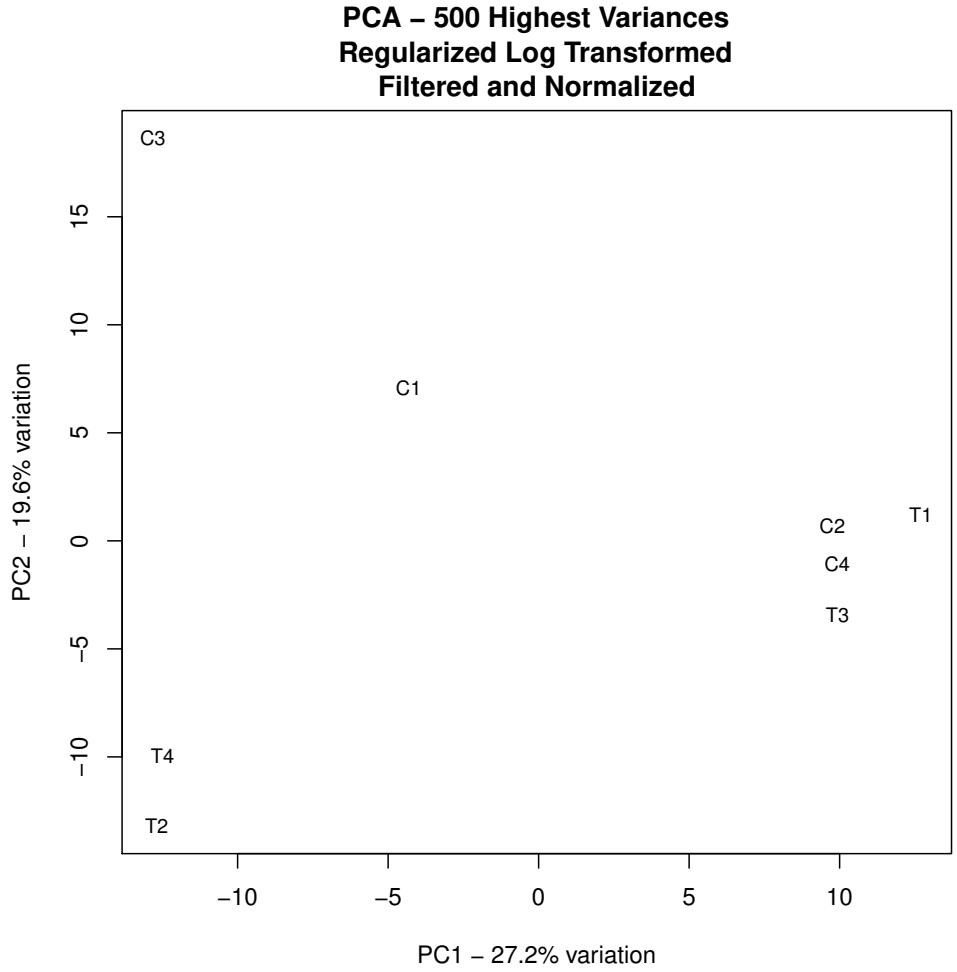

Figure 6.12: PCA analysis using 500 transcripts with the highest row variances. Data is UQ normalized and CPM filtered then transformed using DeSeq2's regularized log transformation method.

6.1.7 Heatmaps

A heatmap is a graphical representation of data where the individual values contained in a data matrix are represented by colours. Any value assigned to a transcript can be plotted; in the case of RNA-seq data this can be counts, p-values or false discovery rates. The data are often plotted with samples as columns and transcripts as the rows, as in Figure 6.13. Alternatively, the distances between samples can be plotted, as in Figure 6.15.

In order to help visualize the clustering effect on the transcripts, a dendogram is placed on

the graph. The tree structure clusters the rows (or columns) based on some distance function. In this paper, heatmaps were created using the `heatmap.2` function from the R Bioconductor [34] package `gplots`. The default clustering method for `heatmap.2` is **complete linkage clustering**.

The method for complete linkage clustering is as follows. Initially, each transcript is in a cluster of its own. The two transcripts separated by the shortest distance are combined into one cluster. In this way, clusters are sequentially combined into larger clusters until all transcripts reside in the same cluster. At each step, the two clusters separated by the shortest distance are combined. Mathematically, the distance between clusters X and Y is defined as $D(X,Y) = \max_{x \in X, y \in Y} d(x,y)$, where

- $d(x,y)$ is the distance between elements $x \in X$ and $y \in Y$; and
- X and Y are two sets of elements (clusters)

To create Figure 6.13 below, the entire dataset of UQ normalized and CPM filtered counts were first ordered by row median. The top thirty transcripts with the highest overall median count across all samples were then selected, scaled and the normalized, filtered counts were plotted. Clustering was performed on the euclidean distance dissimilarity matrix of the Spearman correlation coefficients using the R code:

```
hr <- hclust(as.dist(1-cor(t(mat), method="spearman")), method="complete")
```

In Figure 6.14 the log fold change in count for the same 30 transcripts is plotted. Counts were divided by the median of the control counts, and a small value was added to handle zero's before log_2 transforming. As in Figure 6.13, Clustering was performed on the euclidean distance dissimilarity matrix of the Spearman correlation coefficients.

Figure 6.15 shows the sample to sample euclidean distances for the normalized and filtered counts. Counts were first transformed using `DESeq2`'s variance stabilizing function, `rlog`, which transforms the count data to the log_2 scale in a way that minimizes differences between samples for rows with small counts, and which normalizes with respect to library size. The top 500 transcripts with the highest row variances were selected using the `rowVars` of the

`genefilter` package. Clustering was performed on the Euclidean distance dissimilarity matrix of the samples, calculated using the R [34] functions `as.dist`, `as.matrix` and `dist`, using the following code:

```
#compute raw variances from count matrix (assay(rld)) and select top 500
Pvars<-genefilter::rowVars(assay(rld))
select <- order(Pvars, decreasing = TRUE)[1:500]

#compute euclidean distances of the samples
mat<-as.matrix(dist(t(assay(rld[select,])), method="euclidean"))
rownames(mat) <- colnames(mat)<-colData(rld)$condition

#convert distance matrix to lower triangular and perform clustering
hr<- hclust(as.dist(mat), method = "complete")
```

It is expected to find samples in a treatment group clustered together in Figure 6.15. However samples $C1$ and $T1$ appear to be mis-grouped. This is seen in the PCA plot, Figure 6.12. The lack of clustering may be indicative of a larger problem with sample quality.

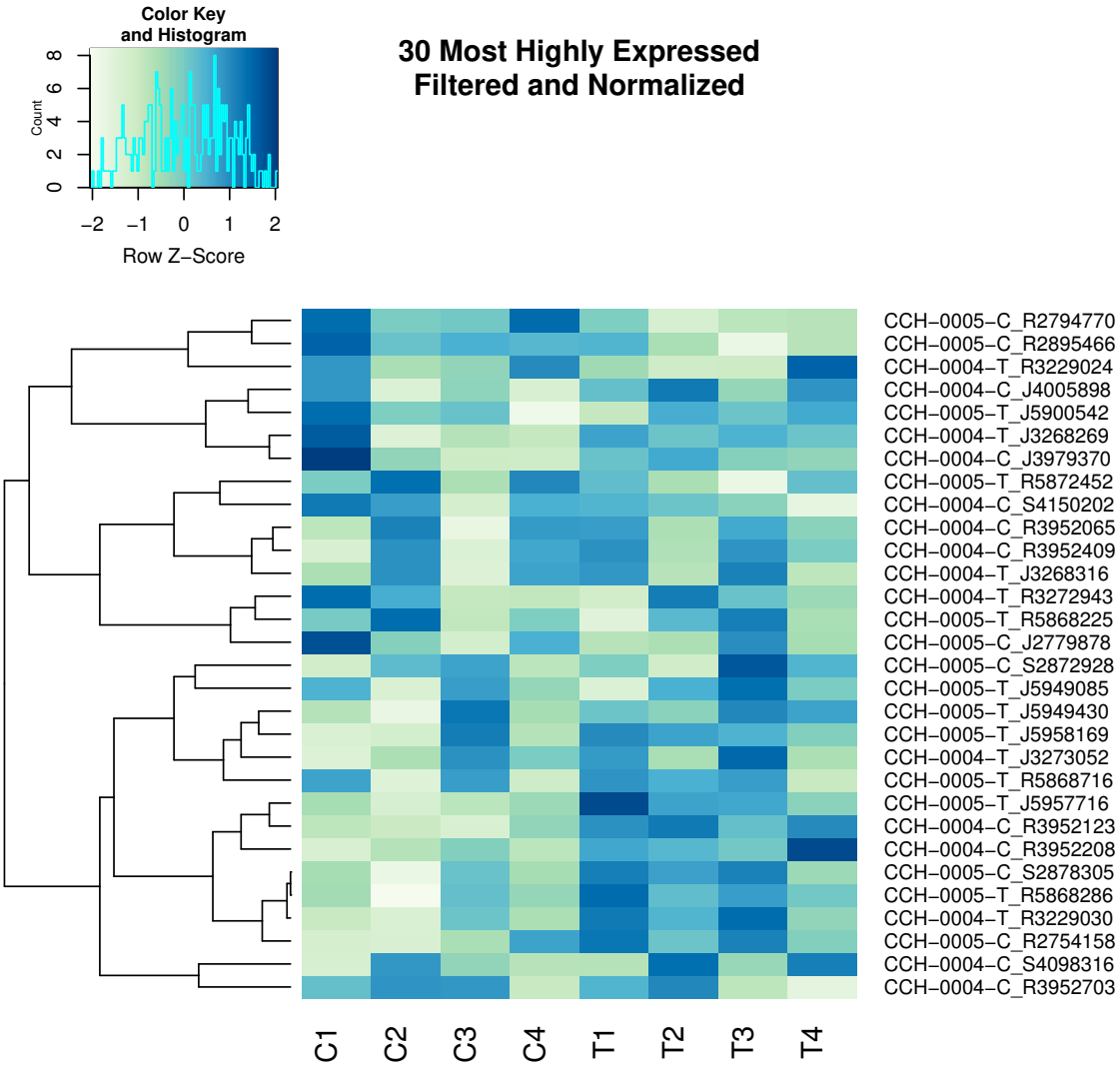

Figure 6.13: Heatmap of the normalized then filtered counts. Data are counts from the 30 transcripts with the highest median value across all samples.

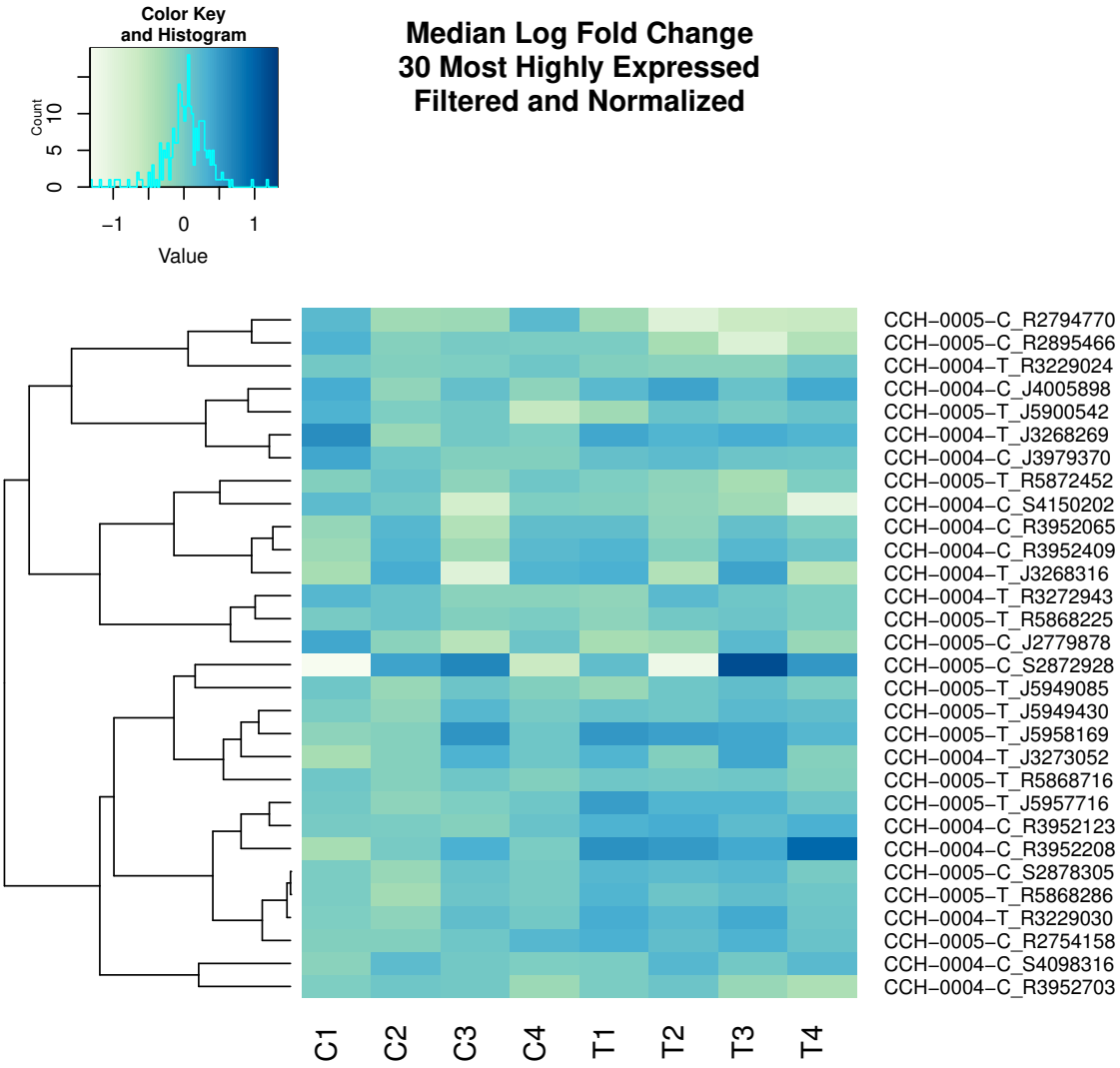

Figure 6.14: Heatmap of the median log fold change of the 30 transcripts with the highest median value across all samples. Data was normalized then filtered before log transforming. Counts were divided by the median of the control counts, and a small value was added to handle zero's.

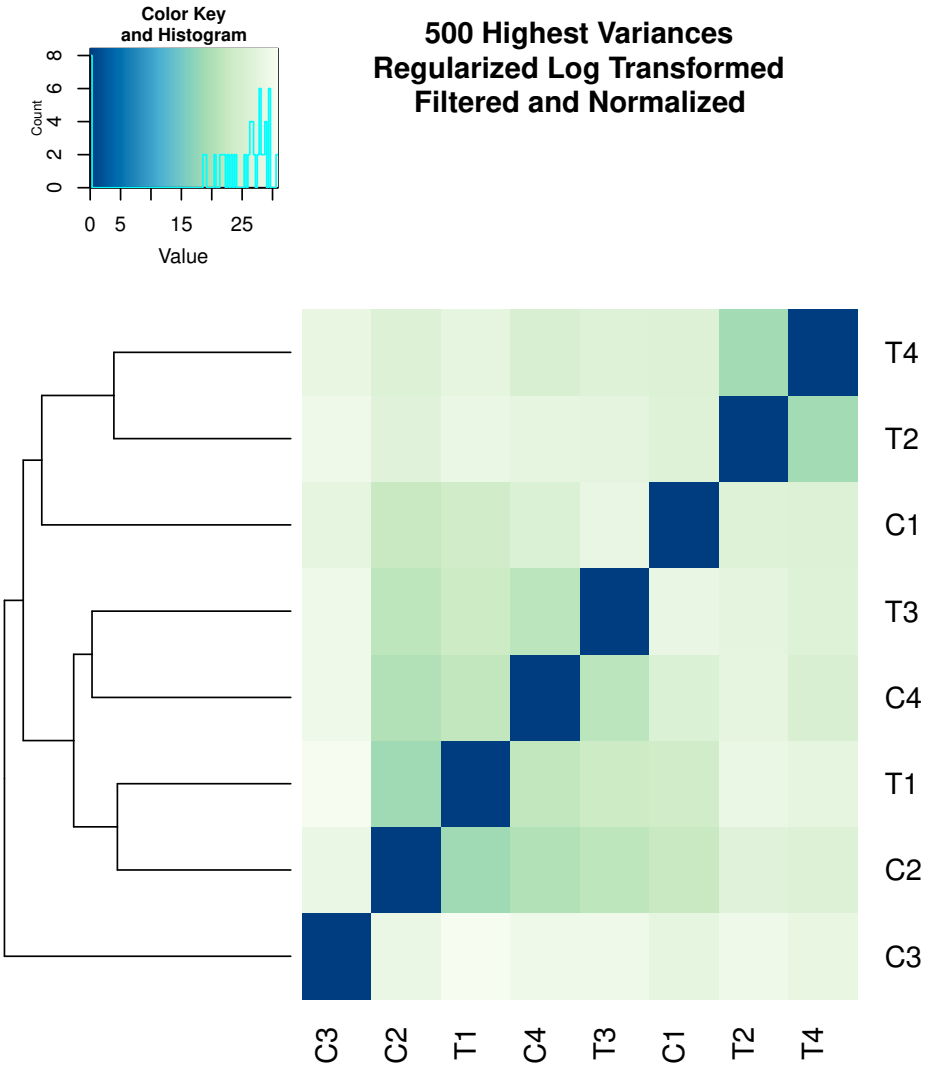

Figure 6.15: Heatmap of euclidean distances between samples using 500 transcripts with the highest row variances. Data is normalized then filtered, then log transformed.

6.2 WEC Data Analysis

6.2.1 Workflow

A workflow was developed to take raw counts and perform a pre-analysis summary of the dataset, followed by a full analysis using five popular statistical packages available through R Bioconductor [34]. The R packages used were SAMseq [45], BaySeq [17], EdgeR [39], Robust EdgeR [39] and DESeq2 [26]. A post-analysis summary was developed that compared the results of the individual packages.

The pre-analysis or descriptive phase examined the dataset for sample or replicate quality, looking for outlier data points and outlier samples, and took a preliminary look at approximate differential expression results. For this, we normalized the data using the UQ method, and subsequently filtered using the CPM method. In many cases, pseudo counts were used to generate the descriptive statistics. Results from running our WEC data through this workflow are found in Section 6.1.

To prepare the data for anaysis using the R packages SAMseq, BaySeq, Robust EdgeR, EdgeR and DESeq2, the raw count data was filtered using the CPM filtering method found in EdgeR [39], with a cut-off of 2 counts per million. Normalization was not necessary prior to filtering as we were not concerned with retaining a normalized dataset; individual pacakges provide their own internal normalization algorithm. More details on these normalization and filtering methods can be found in Chapter 5.

The filtered data was run through the five R statistical packages as mentioned above. Technical and theoretical details for each of these packages are laid out in Chapters 3 and 4, and Appendices A and B . Each package consists of several stand-alone functions through which certain features of the methods can be modified according to user needs. For this workflow we chose the majority of features to be consistent with those used in the corresponding vignettes.

SAMSeq `SAMseq` is a function found in the Bioconductor package `samr` [45]. The function `samr` of the `samr` package was initially written to analyze microarray data; `SAMseq` extends `samr` to handle RNA-seq data, disabling arguments and features that do not apply

to sequencing data.

The SAMSeq function takes a vector of counts, x, a vector of sample groupings, y, a problem type such as 'Two class unpaired' and an FDR rate. In the case of our WEC data, x is the set of filtered (non-normalized) counts, y = $(1, 1, 1, 1, 2, 2, 2, 2)$, where 1 represents the control and 2 represents the treatment samples, and the fdr.output was set to 0.15. Normalization is not necessary, as SAMseq accounts for library sizes internally in the SAMseq function.

```
samfit<-SAMseq(x=fcounts,
               y=samModel,
               resp.type="Two class unpaired",
               fdr.output=cutoff_FDR)
```

```
> samModel
[1] 1 1 1 1 2 2 2 2
```

The outcome of the SAMseq function is a list with the following components.

```
> summary(samfit)
               Length Class  Mode
samr.obj       37     -none- list
del             1     -none- numeric
delta.table     8     -none- numeric
siggenes.table  4     -none- list
call            5     -none- call
```

The most useful component is siggenes.table, which contains a table of significant transcripts that is comprised of components: genes.up, a matrix of significant transcripts having positive correlation with treatment effect ($y = 2$) and genes.lo, a matrix of significant transcripts having negative correlation with treatment effect ($y = 2$).

samr.obj is a list object with elements holding statistics related to the data. Possibly the most useful are tt, the vector of p test statistics for each transcript, foldchange, a p-vector

of mean fold changes between groups for each transcript, and `evo`, a p-vector of expected values for `tt` under permutation sampling.

BaySeq `BaySeq` is an R Bioconductor package [17] designed to identify differential expression in RNA-seq data by calculating estimated posterior likelihoods of differential expression (or more complex hypotheses) via empirical Bayesian methods. In this analysis we follow the recommended series of function calls as documented in the BaySeq vignette. The vignette can be sourced in R by running the command `browseVignettes("baySeq")`.

In preparation for analysis, the models to be tested are defined and stored in the variable `bayGroups`, below). In the current analysis, we are testing models of differential expression (**DE**) and no differential expression (**NDE**), however BaySeq can handle more complex experimental designs. To model no differential expression, it is assumed that all samples belong to the same treatment group. Otherwise, the samples belong to one or more groups, each assumes a treatment effect.

```
bayReplicates <- c("C1", "C1", "C1", "C1", "C2", "C2", "C2", "C2")
bayGroups <- list(NDE = c(1,1,1,1,1,1,1,1), DE = c(1,1,1,1,2,2,2,2))
```

A new `countData` object is created from a combination of the filtered count data and defined models. Normalization is not necessary, as Bayseq accounts for library sizes in the pipeline. The function `getLibsizes` is a helper function which estimates the library scaling factors used to create the `countData` object.

```
bayObj<-new("countData",
            data=as.matrix(fcounts),
            groups=bayGroups,
            replicates=bayReplicates)

libsizes(bayObj)=getLibsizes(bayObj)
```

The function `getPriors.NB` estimates the parameters of the data's underlying Negative Binomial distributiona via quasi-likelihood methods. There is an option to estimate the

parameters via maximum likelihood methods. As our dataset consisted of approximately 45000 transcripts, the default sample size of 100000 used in the quasi-likelihood estimation algorithm was decreased to 1000. `Bayseq` is a computationally heavy package, using a small sample size helped decrease runtime.

After fitting priors, the posterior likelihoods were calculated based on the distributional model specified in the `groups` slot of the `countData` object. The model is automatically used in the `getLikelihoods` function.

```
bayObj<-getPriors.NB(bayObj,
                     samplesize=1000,
                     estimation="QL",
                     cl=NULL)


bayObj<-getLikelihoods(bayObj,
                       pET='BIC',
                       cl=NULL)


results <- topCounts(bayObj,group="DE", number=Inf, normaliseData=TRUE)
```

The prior and posterior likelihood estimates are stored in the `bayObj` object. `topCounts` extracts posterior likelihoods and returns counts with highest (or lowest) likelihood of association with a specified model. Model options for the WEC data are **DE** (model of differential expressed) or **NDE**. Results are returned ordered on likelihoods.

EdgeR and Robust EdgeR The EdgeR package developed by Robinson and Smyth [39] implements exact statistical methods for multigroup experiments. The package comprises a range of statistical methodology based on the negative binomial distributions, including empirical Bayes estimation, exact tests, generalized linear models and quasi-likelihood tests.

EdgeR stores data in a simple list-based data object called a DGEList. This type of object is easy to use because it can be manipulated like any list in R. The DGEList object is created using the `DGEList` function from a matrix or dataframe of counts and is used

downstream in the analysis. As with `SAMSeq` and `BaySeq`, a grouping vector can be defined to differentiate the model of differential expression versus non-differential expression.

In this analysis, the counts passed to the DGEList object were filtered, but not normalized. The `calcNormFactors` function normalizes the count data by finding a set of scaling factors for the library sizes. The default method for computing these scale factors is TMM (trimmed method of means) which is described in more detail in Chapter 5.

```
#data is stored as a pxn matrix in fcounts
group = factor(c( rep("Control",length(class1)), rep("Treated",length(class2)) ))
y_f = DGEList(counts=fcounts, genes=rownames(fcounts), group=group)

y_f=calcNormFactors(y_f)
```

The simplest and most common type of experimental design for RNA-seq data is that in which a number of experimental conditions are compared on the basis of independent biological replicates. The classic edgeR approach is to make pairwise comparisons between groups using the function `exactTest`. The glm approach to multiple groups is similar to the classic approach, but requires a design matrix, allowing for more general comparisons to be made. In the design matrix defined below, the 0+ in the model formula is an instruction not to include an intercept column and instead to include a column for each group.

```
design=model.matrix(~0+group)
colnames(design) = levels(group)

> design
  Control Treated
1       1       0
2       1       0
3       1       0
4       1       0
5       0       1
6       0       1
7       0       1
8       0       1
```

The `estimateDisp` function calculates the likelihood of a study design, conditioning on the total, normalized pseudo counts for each transcript. Common dispersion and transcript-wise dispersion estimates are calculated.

```
y_f=estimateDisp(y_f,design)
```

`glmFit` fits the data to a negative binomial GLM that corresponds to the study design. It produces an object of class `DGEGLM` which is passed to `glmLRT` to carry out the likelihood ratio test for non-differential expression. One can compare any of the treatment groups using the contrast argument of the glmLRT function. Below, a contrast of $c(-1, 1)$ corresponds to a test of **Treated-Control**.

```
fit=glmFit(y_f,design)

lrt=glmLRT(fit, contrast=c(-1,1))
```

`topTags` returns the top `n` differentially expressed transcripts and corresponding p-values.

```
top=topTags(lrt,n=Inf)
```

`EdgeR Robust` is more robust to the presence of outliers than `EdgeR`. The command flow of `EdgeR` and `EdgeR Robust` are identical, with the exception of setting the `robust` option of the `estimateDisp` to `TRUE`, as below.

```
y_f=estimateDisp(y_f,design, robust=TRUE)
```

Deseq2 Before performing an analysis using Deseq2, the data needs to be stored in an object of class `DESeqDataSet`, which is used to store the input values, intermediate calculations and results of final analysis of differential expression. The `DESeqDataSet` class stores the count matrix as a list element, enforcing non-negative count values. In addition, a formula which specifies the design of the experiment must be provided.

```
colData <- data.frame(condition=factor(c(rep("Control",length(class1)),
 rep("Treated",length(class2)))))

dds=DESeqDataSetFromMatrix(countData=fcounts,colData=colData,
         formula(~condition))

dds=DESeq(dds, test="LRT", reduced=~1)
```

The analysis is called through a wrapper function, `DESeq` which calls the following functions:

- `estimateSizeFactors`: This function estimates the size factors using the 'median ratio method' described by Equation 5 in Anders and Huber (2010).
- `estimateDispersions`: This function obtains dispersion estimates for Negative Binomial distributed data.
- `nbinomWaldTest`: This function tests for significance of coefficients in a Negative Binomial GLM, using previously calculated sizeFactors and dispersion estimates.

The `DESeq` function returns a `DESeqDataSet` object from which tables of log_2 fold changes and p-values can be generated using the `results` function.

6.2.2 Results

Results from the analysis for all packages were combined together. Control and treatment medians were calculated for each transcript, along with the median log fold change, $log_2(C_g/T_g)$. C_g and T_g are the median counts for transcript g for the control and treatment sample groups, respectively. Differential expression was based on a false discovery rate of 0.15, the choice of such a high FDR was made to accommodate the small dataset size.

Results were reported from analysis of the filtered data. Results for all transcripts were obtained for `BaySeq`, `DeSeq2`, `EdgeR` and `robust EdgeR`.

Results were obtained for those transcripts found to be differentially expressed by `SAMSeq`. The underlying statistical test for DE involves a Wilcoxon statistic so there is no p-value or likelihood on which to order the results. Results are shown in Table 6.4 for those transcripts with the highest `Score_d` values.

	C1	C2	C3	C4	T1	T2	T3	T4	Score_d	Fold_Change	Q_Val	DE.n
CCH-0004-C_J4006695	742	970	684	621	1039	996	673	773	8.00	1.18	19.46	1.00
CCH-0004-C_J4008334	647	1234	868	824	1308	1111	1153	1144	8.00	1.32	19.46	1.00
CCH-0004-C_J4008372	1217	1782	1087	1003	2135	1669	1244	1236	8.00	1.31	19.46	1.00
CCH-0004-C_R3952123	4859	7373	4151	4580	7972	7051	4954	5674	8.00	1.29	19.46	1.00
CCH-0004-C_R3952208	3493	6874	5005	3780	8856	7163	5130	8139	8.00	1.53	19.46	1.00
CCH-0004-C_R3960750	14	20	13	10	49	44	22	34	8.00	2.81	19.46	1.00
CCH-0004-C_S3895400	0	43	0	4	94	104	60	84	8.00	36.83	19.46	1.00
CCH-0004-C_S4118923	370	625	407	500	827	800	712	627	8.00	1.69	19.46	1.00
CCH-0004-C_S4122880	8	90	1	0	157	184	120	129	8.00	31.98	19.46	1.00
CCH-0004-T_J3242775	1126	1785	1267	1219	2167	1819	1662	1480	8.00	1.33	19.46	1.00
CCH-0004-T_J3270609	1008	1642	1166	1008	1894	1569	1290	1189	8.00	1.31	19.46	1.00
CCH-0004-T_R3387090	577	1020	666	501	1034	990	756	835	8.00	1.47	19.46	1.00
CCH-0004-T_S3079432	153	230	149	135	272	239	187	175	8.00	1.42	19.46	1.00
CCH-0004-T_S3271084	1532	2480	1473	1614	2538	2204	1758	1770	8.00	1.25	19.46	1.00
CCH-0004-T_S3327536	22	46	9	9	74	54	35	44	8.00	3.06	19.46	1.00
CCH-0005-C_J2781937	795	1200	730	667	1347	1075	852	855	8.00	1.29	19.46	1.00
CCH-0005-C_S2863416	0	0	0	0	4	4	5	6	8.00	515599226.46	19.46	1.00
CCH-0005-C_S2888315	1482	2168	1191	1163	2054	1913	1698	1465	8.00	1.23	19.46	1.00
CCH-0005-T_J5953554	719	1262	834	787	1402	1239	970	968	8.00	1.30	19.46	1.00
CCH-0005-T_J5957716	3300	4490	2881	2959	6054	4486	3503	3273	8.00	1.32	19.46	1.00
CCH-0005-T_J5958155	886	1408	1054	905	1644	1439	1104	1205	8.00	1.36	19.46	1.00
CCH-0005-T_R5868286	3357	4280	3316	3008	5499	4174	3443	3365	8.00	1.19	19.46	1.00
CCH-0005-T_S5943284	909	1350	974	846	1520	1376	1065	1064	8.00	1.31	19.46	1.00

Table 6.4: Results from SAMSeq analysis including filtered and normalized counts. Normalized by applying SAMSeq's scaling factor to column data. Only top DE transcripts shown.

Table 6.5 shows the top 10 transcripts found by BaySeq, along with their counts and associated likelihood of differential expression. The posterior log likelihood that a transcript exhibits differential expression under the treatment are calculated using the function `getLikelihoods` (called with default parameters). For this data, transcripts with a likelihood value of less than 0.15 were considered differentially expressed.

The `topCounts` function takes a `countData` object containing posterior likelihoods and returns the counts with highest (or lowest) likelihood of association with a given group. The user can specify how many transcripts are returned and place a restriction on the group ordering (i.e. up-regulated, down-regulated or both). Alternatively the function will return a subset of transcripts with FDR or FWER above a specified certain value.

	C1	C2	C3	C4	T1	T2	T3	T4	Likelihood	ordering	FDR.DE	FWER.DE	DE.n
CCH-0004-C_S3895400	0	30	0	5	79	101	74	96	1.00	2>1	0.00	0.00	1
CCH-0004-C_S4122880	8	63	1	0	131	179	148	148	0.99	2>1	0.01	0.01	1
CCH-0004-C_R3960750	15	14	14	11	41	43	27	39	0.96	2>1	0.02	0.05	1
CCH-0004-C_S4093449	12	7	8	7	23	26	28	24	0.94	2>1	0.03	0.10	1
CCH-0005-C_S2876299	8	6	7	2	17	21	21	18	0.90	2>1	0.04	0.20	1
CCH-0004-T_R3298384	9	17	14	19	3	3	6	2	0.88	1>2	0.05	0.29	1
CCH-0004-T_S3304335	21	18	22	17	8	6	7	7	0.87	1>2	0.07	0.39	1
CCH-0004-C_S4131948	4	8	49	5	53	51	49	39	0.85	2>1	0.08	0.48	1
CCH-0005-T_J5904561	21	19	13	30	43	61	48	53	0.84	2>1	0.09	0.56	1
CCH-0005-T_R5950008	17	7	19	17	3	0	1	2	0.83	1>2	0.09	0.63	1

Table 6.5: Results from BaySeq analysis including filtered and normalized counts. Normalized by by BaySeq. Only top 10 DE transcripts shown.

Tables 6.6 and 6.7 show the top 10 transcripts, ordered by FDR, found to be differentially expressed using `edgeR` and `robustEdgeR` respectively. The function `fitGLM` fits a negative binomial glm to the count data, using the same design matrix for both packages. Different dispersions, offsets and weights can be passed as arguments, or taken from prior estimation. Tests for differential expression produce p-values, which are adjusted for multiple comparisons using a Bonferronni-type procedure. For this data, transcripts with an adjusted p-value less than 0.15 were considered differentially expressed.

	C1	C2	C3	C4	T1	T2	T3	T4	logFC	logCPM	LR	PValue	FDR	DE.n
CCH-0004-T_S3279015	0	0	2	0	3	35	14	17	5.12	3.38	31.96	0.00	0.00	1
CCH-0005-T_S2754924	5	6	4	4	0	0	0	0	-5.56	1.99	27.50	0.00	0.00	1
CCH-0005-T_S5931640	4	9	54	6	0	0	0	2	-5.04	3.39	25.35	0.00	0.00	1
CCH-0004-C_S4104120	7	6	8	6	1	0	0	0	-3.64	2.32	24.21	0.00	0.01	1
CCH-0005-C_S2863416	0	0	0	0	3	3	5	6	5.35	1.84	23.44	0.00	0.01	1
CCH-0005-T_R5950008	14	6	17	15	3	0	1	2	-2.99	3.07	22.67	0.00	0.01	1
CCH-0004-C_S4043614	0	1	4	4	4	295	1	122	5.73	5.81	22.59	0.00	0.01	1
CCH-0004-C_J3968809	77	243	206	355	50	74	47	81	-1.73	7.14	22.47	0.00	0.01	1
CCH-0004-C_S4112016	8	5	9	6	0	0	0	2	-3.64	2.34	21.86	0.00	0.01	1
CCH-0004-C_R4075426	2	1	0	0	7	21	3	10	3.62	2.86	20.61	0.00	0.02	1

Table 6.6: Results from EdgeR analysis including filtered and normalized counts. Normalized by transforming to CPM using EdgeR's extttcpm function. Only top 10 DE transcripts shown.

The top transcripts are found by calling the function `topTags`. The function takes a `DGE` object, a data frame containing information on the individual transcripts including the log-fold change in expression between groups and the p-value for differential expression. The function extracts the top differentially expressed transcripts ranked by p-value or absolute log-fold change.

	C1	C2	C3	C4	T1	T2	T3	T4	logFC	logCPM	LR	PValue	FDR	DE.n
CCH-0004-T_S3279015	0	0	2	0	3	35	14	17	5.12	3.38	31.75	0.00	0.00	1
CCH-0005-T_S2754924	5	6	4	4	0	0	0	0	-5.55	1.99	26.43	0.00	0.00	1
CCH-0005-C_S2863416	0	0	0	0	3	3	5	6	5.35	1.84	22.92	0.00	0.01	1
CCH-0004-C_S4104120	7	6	8	6	1	0	0	0	-3.64	2.32	22.84	0.00	0.01	1
CCH-0005-T_R5950008	14	6	17	15	3	0	1	2	-2.99	3.07	22.57	0.00	0.01	1
CCH-0004-C_R4075426	2	1	0	0	7	21	3	10	3.62	2.86	21.16	0.00	0.02	1
CCH-0005-T_S5931640	4	9	54	6	0	0	0	2	-5.03	3.39	21.14	0.00	0.02	1
CCH-0004-C_S4112016	8	5	9	6	0	0	0	2	-3.64	2.34	21.07	0.00	0.02	1
CCH-0004-C_R3960750	12	11	12	10	32	34	21	31	1.45	4.49	19.06	0.00	0.04	1
CCH-0004-T_S3382104	24	28	20	8	3	7	0	5	-2.32	3.76	19.00	0.00	0.04	1

Table 6.7: Results from Robust EdgeR analysis including filtered and normalized counts. Normalized by transforming to CPM using EdgeR's extttcpm function. Only top 10 DE transcripts shown.

Using the R package `DeSeq2`, we ran the `results` function on a `DESeqDataSet` object extracting a results table from a DESeq analysis.

The resulting p-values were adjusted for multiple comparisons using a Bonferroni-type procedure that controls the false discovery rate. The adjusted p-values found in column `padj` are returned through the `results` function along with base means across samples, log_2 fold changes, standard errors and test statistics. The criteria for selecting which transcripts are differentially expressed are based on these adjusted p-values. Transcripts with an adjusted p-value less than 0.15 are considered differentially expressed. After ordering the results by decreasing adjusted p-value, the top DE transcripts, based on lowest adjusted p-value, are shown in Table 6.8. Only four transcripts were found to be differentially expressed by this package.

	C1	C2	C3	C4	T1	T2	T3	T4	padj	logfold	DE.n
CCH-0004-T_S3279015	0	0	2	0	3	42	17	20	0.003	5.441	1
CCH-0004-C_J3968809	89	283	229	407	60	89	59	96	0.028	-1.732	1
CCH-0004-C_R3960750	14	13	14	11	39	41	27	37	0.14	1.462	1
CCH-0005-T_R5950008	16	7	19	17	3	0	1	2	0.14	-3.094	1

Table 6.8: Results from DESeq2 analysis including filtered and normalized counts. Normalized by applying DESeq2's normalization factors to column data. Only top DE transcripts are shown.

Transcripts found to be differentially expressed (adjusted p-value < 0.15) were chosen for further PCA analysis. Counts were regularized log transformed using Deseq2's `rlog`

function, then scaled and centered before being passed through `prcomp`. Results are shown below in Figure 6.16.

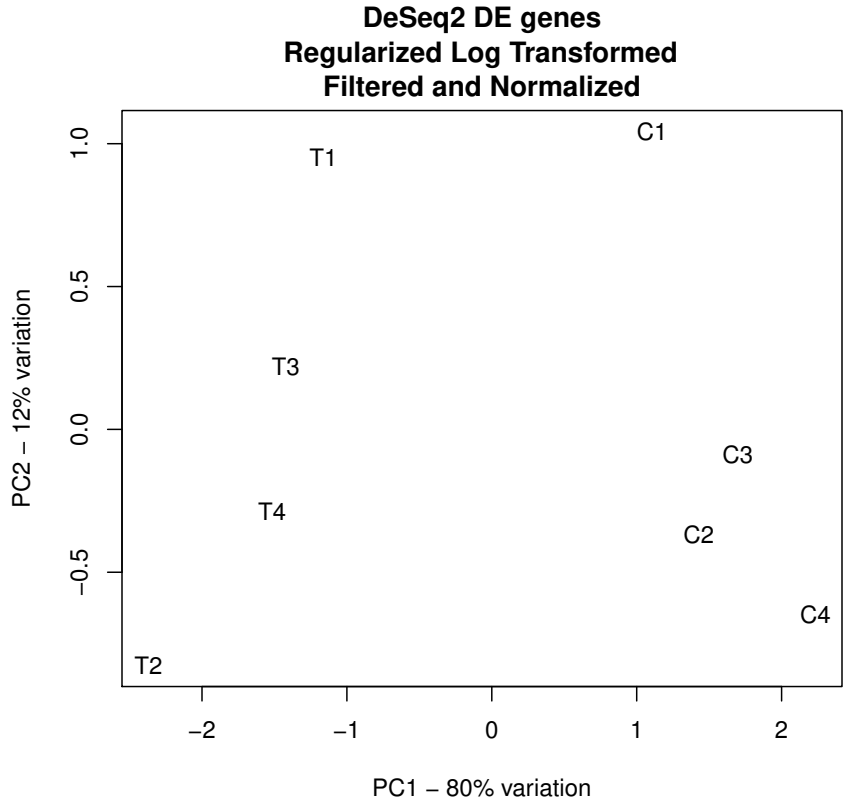

Figure 6.16: PCA analysis plot using top DE transcripts found using DeSeq2. Data is normalized and filtered before log transforming using DeSeq2's `rlog` method.

The first four principal components are shown in Figure 6.17, the first component is seen to carry the large majority (80%) of the variation. The samples show a strong pattern of clustering due to the first principal component (see Figure 6.16). This is expected as the transcripts chosen were differentially expressed.

Figure 6.18 shows the log_2 median fold change for the filtered transcripts found to be differentially expressed by `DESeq2`. Counts were divided by the row median of the control

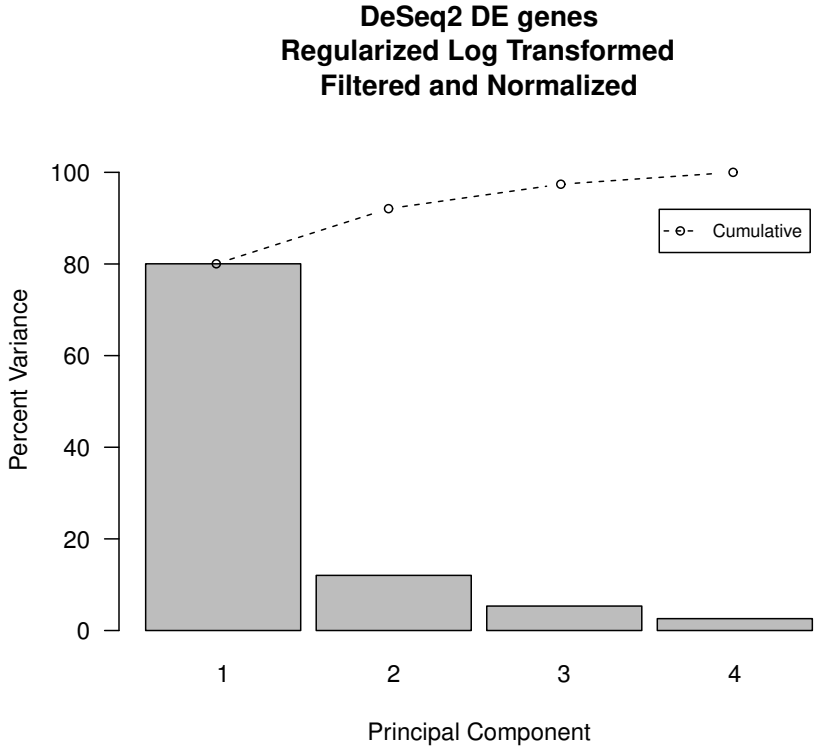

Figure 6.17: PCA analysis scree plot using top DE transcripts found using DeSeq2. Data is normalized and filtered before log transforming using DeSeq2's `rlog` method.

group, then log_2 transformed and plotted. Clustering was performed on the euclidean distances of the dissimilarity matrix of the Spearman correlation coefficients, using the method 'complete linkage clustering'.

```
hr <- hclust(as.dist(1-cor(t(mat), method="spearman")),
 method="complete")
```

Figure 6.19 shows the log_2 median fold change for the filtered transcripts found to be differentially expressed by `DESeq2`. Transcripts were ordered by adjusted p-value and the

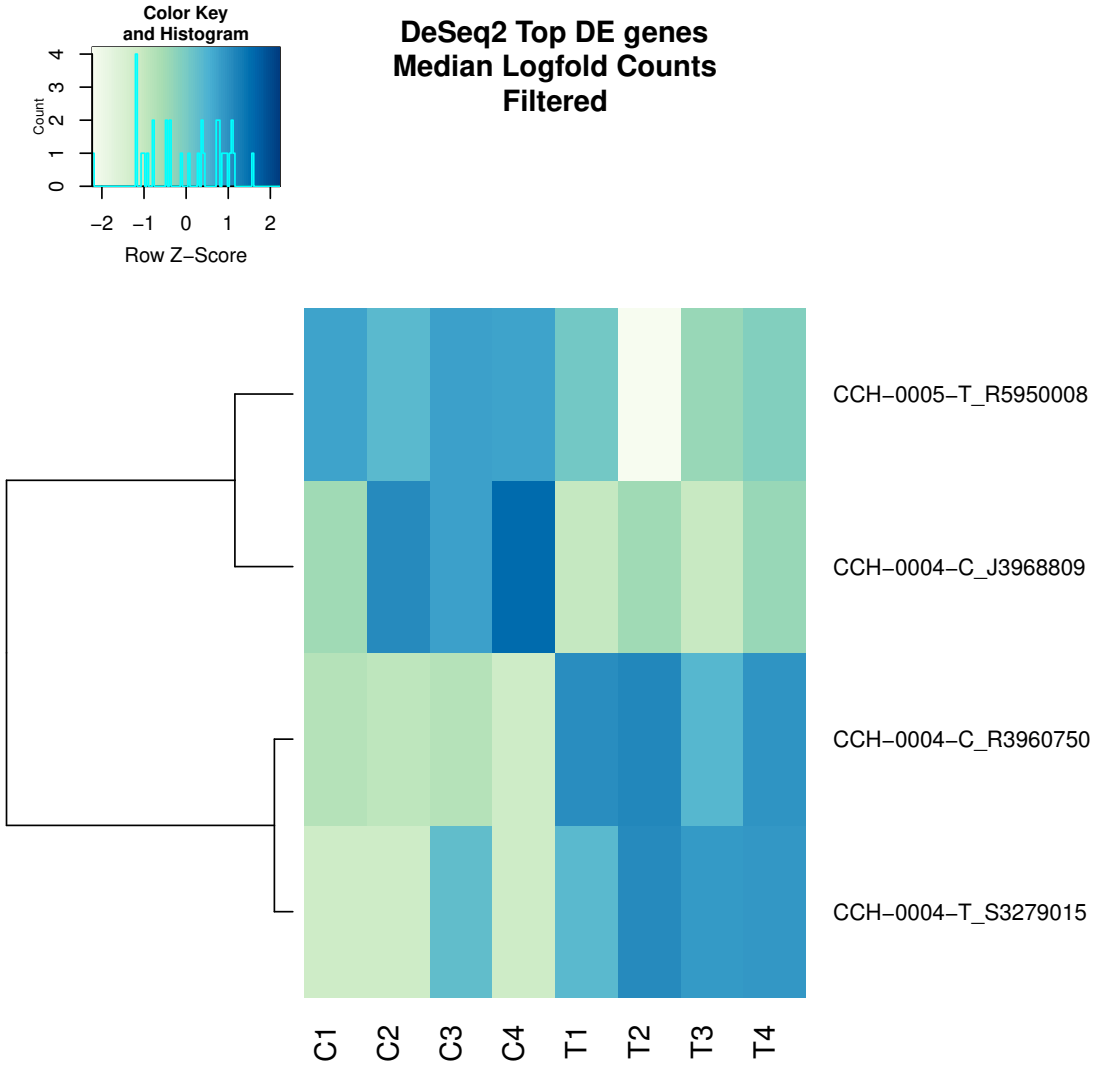

Figure 6.18: Heatmap of log_2 median fold change for DE transcripts found using DeSeq2. Complete linkage clustering for the rows based on the Spearman correlation coefficients of the filtered data.

top transcripts were chosen. As there were only four transcripts found by `DESeq2` to be differentially expressed in the WEC data, only these four were selected. For this reason, Figures 6.19 and 6.18 are identical.

As in Figure 6.18, clustering was performed on the euclidean distances of the dissimilarity matrix of the Spearman correlation coefficients, using the method 'complete linkage clustering'. Counts were divided by the row median of the control group, then log_2 transformed

and plotted.

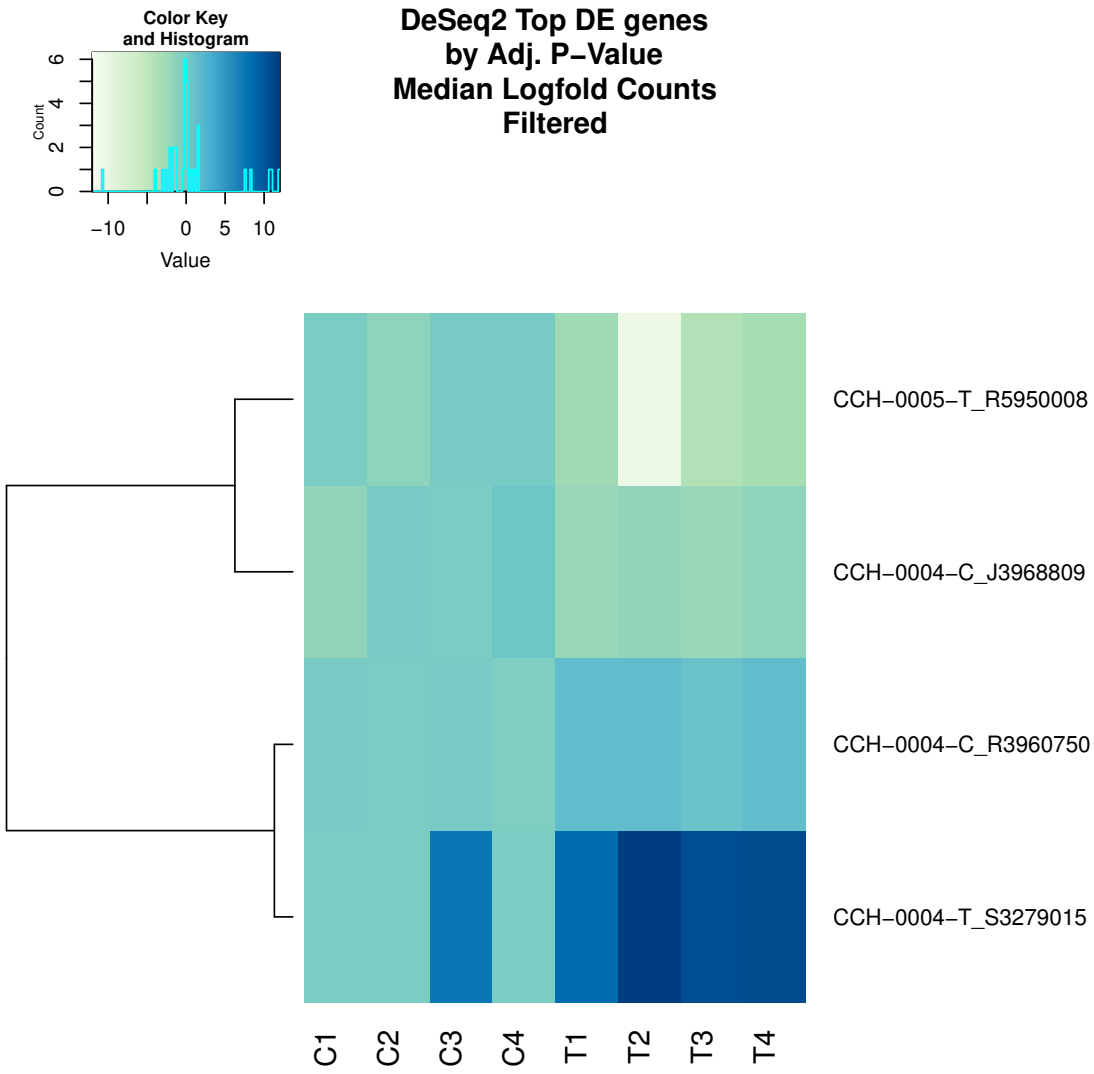

Figure 6.19: Heatmap of log_2 fold change for DE transcripts found using DeSeq2. Results were ordered by adjusted p-value and the top transcripts were chosen. Complete linkage clustering for the rows based on the Spearman correlation coefficients of the filtered data.

Figure 6.20 shows the per-transcript dispersion estimates plotted against the fitted mean-dispersion relationship for the filtered data. DESeq2 [26] automatically normalizes the data, see Appendix A for details on the procedure.

Transcript-wise MLE's, obtained using the transcript count data, are represented by the black dots. A smooth curve is fit to the MLE's that captures the overall dispersion-mean trend. This fit is used as a prior mean for a second estimation round, which results in the final estimates of dispersion shown in blue. Black points circled in blue are the detected dispersion outliers and not shrunk toward the prior. This dispersion plot is typical, with the final estimates shrunk from the transcript-wise estimates towards the fitted estimates.

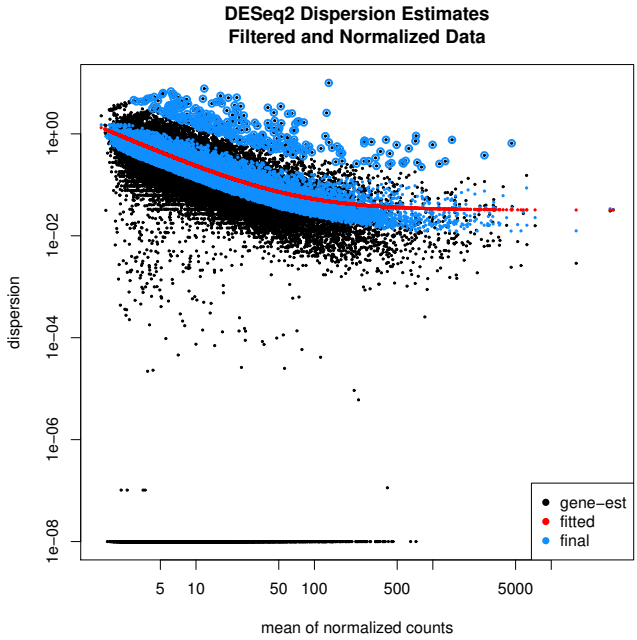

Figure 6.20: Dispersion estimates of the filtered and normalized counts. Counts were normalized using DESeq2's normalization method, found in Appendix A

The volcano plot in Figure 6.21 summarizes both the average expression values and p-values for the WEC dataset. Each point on the graph represents a single transcript, with those found to be differentially expressed by DESeq2 coloured red. It is a scatter-plot of the negative log_{10} transformed p-values from the transcript-specific test (y-axis) against the log_2 fold change (x-axis) of the treatment group over the control group.

Data points with low p-values (highly significant) appearing towards the top of the plot.

The log_2 of the fold change is used so that changes in regulation (both up and down) appear equidistant from the centre. Points that are found towards the top of the plot that are far to either side are likely to be transcripts of interest. These represent values that display large magnitude fold changes and have high statistical significance.

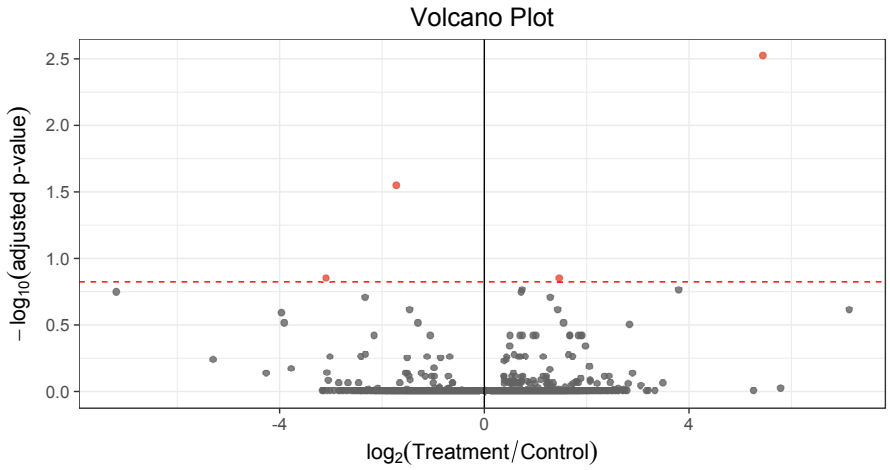

Figure 6.21: Volcano plot of the negative log_{10} adjusted p-values against the log fold change for filtered counts. The red dots represent transcripts found to be differentially expressed.

From the final results table, a Venn diagram was created using the R package `VennDiagram`. Figure 6.22 shows the overlap of transcripts found to be differentially expressed by all packages. Transcripts found in common between all packages are listed in Table 6.9.

	Transcript
1	CCH-0004-C_R3960750

Table 6.9: DE transcript(s) found in common by DESeq2, EdgeR, BaySeq, Robust EdgeR and SAMSeq

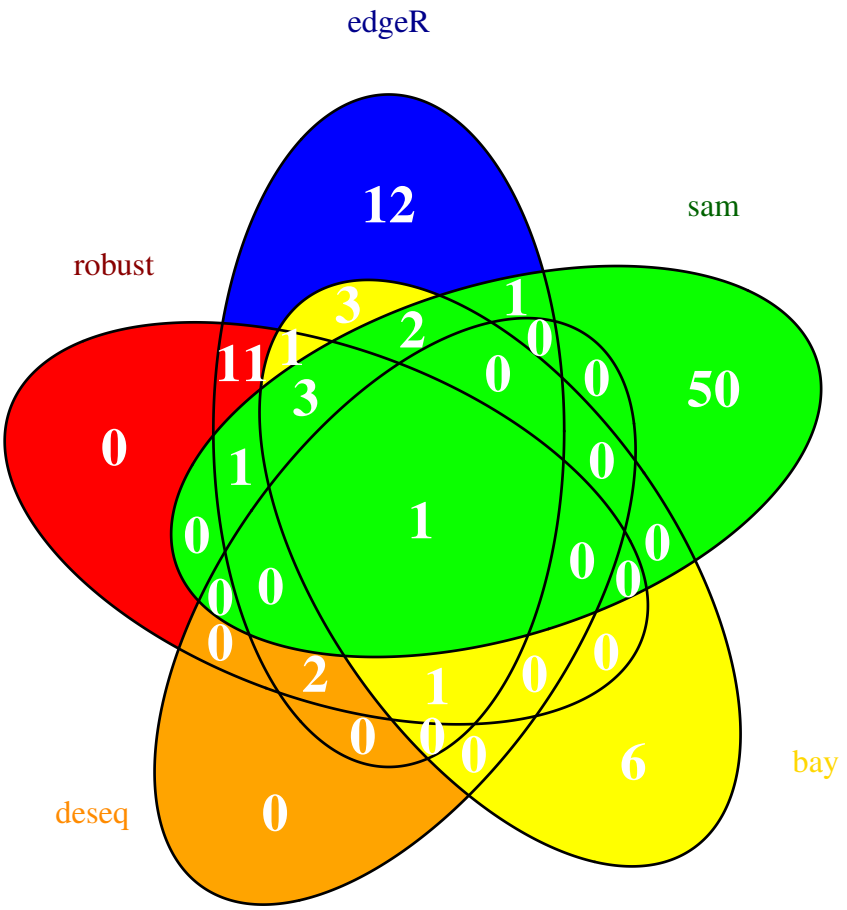

Figure 6.22: Venn diagram of the overlapping DE transcripts found by SAMSeq, BaySeq, DeSeq2, EdgeR and robust EdgeR.

Chapter 7

Simulations

Several simulation algorithms were considered as candidates for creating our test dataset. Three of these are described below. All three were taken from previous publications and have been used to test the performance of novel methods for differential expression analysis of RNA-seq data [24], [47], [41]. The final choice for the simulation scheme was based on ease of implementation and how accurately the associated model was reported to identify true transcript abundance estimates.

7.1 Data Simulation 1 Theory

The following simulation was proposed by Li et al. in their paper introducing the R Bioconductor [34] package PoissonSeq [24]. Counts are assumed to follow the Poisson distribution so that $N_{gi} \sim \text{Poisson}(\mu_{gi})$, where the form of μ_{gi} is given by the log-linear model.

$$\log \mu_{gi} = \log \beta_g + \log d_i + \gamma_g I_{(i \epsilon C_2)} \tag{7.1}$$

Here, N_{ig} is the read count for transcript g from sample i and d_i is the relative sequencing depth for sample i as defined in Chapter 5. β_g is the expression estimate for transcript g and $I_{(i \epsilon C_2)}$ is an indicator variable for such that $I_{(i \epsilon C_2)} = 1$ if sample i belongs to group 2, otherwise it is 0. Then γ_g is a small random value drawn from $N(0, 1)$, whose sign($\pm$)

is chosen to reflect the desired composition for the data (up-regulated or down-regulated); $\gamma_g = 0$ indicates no differential expression.

As an example simulation scenario, consider a dataset derived from a single sample of real RNA-seq counts, such as the one sample, R1L2Liver, of the Marioni dataset in [29]. For counts N_g, $g = 1, \cdots, p$, the β_g's are estimated as $\hat{\beta}_g = N_g / \sum_{k=1}^{p} N_k$. Then $l_i \sim$Unif$(4, 6)$ generates library sizes between one and eight million for the i samples, from which we can estimate the relative sequencing depths, d_i. To create a dataset of $p = 20,000$ transcripts with 10% up-regulated and 5% down-regulated, a subsample, p', of size 3000 is randomly selected to be differentially expressed. For 2000 of the p' DE transcripts, $\gamma_g \sim$ N$(0, 1)$ and for the other 1000, $-\gamma_g \sim$N$(0, 1)$. For those transcripts not included in the DE set, $\gamma_g = 0$.

7.2 Data Simulation 2 Theory

Zhou et al [47] developed a flexible simulator and implemented it as a R Bioconductor [34] package that has the capability to simulate data from preloaded or user-defined RNA-seq datasets. Various features of the dataset can be manipulated, including the percentage of DE transcripts, fold difference and percentage of outliers. The source code is available online [38].

It is assumed that the counts are drawn from a negative binomial distribution, with mean μ and dispersion ϕ. Then the mean is defined with a log function as

$$\log \mu_g = X\beta_g + \log d_i \tag{7.2}$$

where d_i is the sequencing depth, β_g is the abundance estimate of transcript g, $g = 1, ..., p$.

The $\hat{\beta}_g$ are estimated by maximum likelihood estimation, and the following adjusted profile likelihood is constructed.

$$APL_j(\phi_j) = l(\phi_g; y_j, \hat{\beta}_j) - \frac{1}{2} \log |I| \tag{7.3}$$

where $|I|$ is the determinent of the information matrix of β. The average APL for a set, C, of transcripts is defined for those features whose average log counts per million are similar

to transcript j. Then

$$\hat{\phi_j} = \text{argmax}[APL_j(\phi_j) + \gamma\ APL_g^C(\phi_j)] \tag{7.4}$$

Here, γ is a weight attached to the average APL. At each iteration of the dispersion estimation procedure, a weight is attached to the log likelihood, dampening the effect of outliers. A more detailed outline of this process can be found in [47]. Estimation of $\hat{\mu_j}$ is then the fitted value of the linear component of the GLM, considering $\hat{\beta}_j$.

With estimated values for $\hat{\mu}$ and $\hat{\phi}$, counts can be generated with a negative binomial so that $N_{ij} \sim NB(\hat{\mu_{ij}}, \hat{\phi_{ij}})$ where $Var(N_{ij}) = \hat{\mu_{ij}}(1 + \hat{\phi}_j\hat{\mu_{ij}})$.

The following code generates a dataset with 8 samples of 100,000 transcripts, divided into two groups. Mean and dispersion estimates are based on those of the Pickrell Hap Map data [31], filtered using the CPM method. CPM filtering is automatically performed by the simulator. The minumum sample size is assumed to be two, and the cut-off value is set at one. Library sizes of between 1 and 8 million are randomly chosen. Differential expression is set at 5%, with 3/4 of the transcripts up-regulated and 1/4 down-regulated. Those transcripts that are differentially expressed will exhibit a fold change of 2. Outliers were not introduced into this dataset.

```
#source simulator
source("http://130.60.190.4/robinson_lab/edgeR_robust/robust_simulation.R")

#generate library sizes 1..8 million
librSz=exp(runif(min=13.8, max=16, n=(reps_group1+reps_group2)))

#simulate data
sim <- NBsim(foldDiff=2, pickrell, group=c(0,0,0,0,1,1,1,1), nTags=100000,
             lib.size=librSz, add.outlier = FALSE,
             outlierMech="S", pDiff=0.05, pUp=0.75)
```

7.3 Data Simulation 3 Theory

The following simulation scheme was developed by Robles et al.[41] as part of an experiment designed to test the performance of several different methods of differential expression analysis. It is based on the assumption that counts from purely technical replicates follow a Poisson distribution [29]. Sources of variability in the data can be technical such as introduced during library preparation, or biological.

Let the molar concentration, or abundance, of some transcript of interest from some lane of the illumina sequencer be the random variable R. Define $E(R) = q$ and $Var(R) = v$. Let the number of reads mapped onto this transcript be the random variable K. It has been shown [29] that the number of reads can be modelled using $K|(R = r) \sim \text{Pois}\ (\lambda r)$, where λ is some normalisation factor for the sample.

Then the expected value and variance of K, conditional on R, is $E(K|R = r) = \lambda r$ and $Var(K|R = r) = \lambda r$.

It follows using the law of total expectation and the law of total variance that the mean and variance of K are:

$$\begin{aligned} E(K) &= \mu \\ Var(K) &= \mu(1 + \phi\mu) \end{aligned}$$

where $\mu = q\lambda$ and $\phi = \frac{v}{q^2}$.

Further, assume R to be a Gamma random variable. Then it can be shown that K is drawn from a Negative Binomial distribution. Then

$$\begin{aligned} R &\sim \text{Gamma(mean=q, var=v)} \\ R\lambda &\sim \text{Gamma(mean=}\mu\text{, var=}\phi\mu^2\text{)} \\ K &\sim \text{NB(mean=}\mu\text{, var=}\mu(1 + \phi\mu)\text{)} \end{aligned}$$

and the probability mass function is given by

$$Pr(K = k|\mu, \phi) = \frac{\Gamma(k + \phi^{-1})}{\Gamma(k+1)\Gamma(\phi^{-1})} \frac{\mu\phi^k}{(1+\mu\phi)^{k+\phi^{-1}}} \tag{7.5}$$

where ϕ is the dispersion parameter. As $\phi \to 0$, the NB distribution becomes a Poisson distribution with mean μ.

The log likelihood for n counts, y_g, $g = 1, ..., n$ from the NB distribution is

$$\begin{aligned} l(\mu, \phi|y_1, ..., y_n) = \sum_{g=1}^{n} \log\Gamma(y_g + \frac{1}{\phi}) - \\ n\log\Gamma(\frac{1}{\phi}) + \sum_{g=1}^{n} \log\Gamma(y_g + 1) + \\ \sum_{g=1}^{n} y_g \log(\frac{\mu\phi}{1+\mu\phi}) - \frac{n}{\phi}\log(1+\mu\phi) \end{aligned} \tag{7.6}$$

Estimates for $\hat{\mu}$ and $\hat{\phi}$ can be obtained from a representitive RNA-seq data sample. The MLE estimate $\hat{\mu} = \sum_{j=1}^{n} y_j$ is directly obtained from the above log likelihood by setting $\frac{\partial l}{\partial \mu_j} = 0$ and solving for $\hat{\mu_j}$. Using the parameter estimate for $\hat{\mu}_j$, $\hat{\phi}$ can be numerically computed. These estimates describe an approximate null distribution for a single sample from an RNA dataset. The full set of null counts can be obtained by repeating the above for several different samples.

To simulate group 2 counts, Robles et al. [41] proposed sampling from $N_g \sim NB(\theta_g\hat{\mu_g}, \phi_g)$, where θ_g is called the **regulating factor** for μ_g and is responsible for regulating the expression amount and composition for transcript g. Here, we follow the scheme with some modifications.

For null expression, $\theta_g = 1$, which we randomly allocated to 85% of the transcripts. The remaining 15% were evenly divided between up and down regulated and

$$\theta_g = \begin{cases} (1 + X_g), & \text{for up-regulated transcripts} \\ (1 + X_g)^{-1} & \text{for down-regulated transcripts} \end{cases} \tag{7.7}$$

where $X_g \sim Exp(1)$.

A full dataset can be produced by repeating the above for all samples from the null distribution.

Chapter 8

Experiment

8.1 Simulated Data

8.1.1 Background

An issue that complicates differential expression analysis is the large number of hypothesis tests that are performed which decrease the power of the statistical tests. When testing multiple hypotheses simultaneously, many truly null hypotheses will produce small p-values by chance. Consequently, numerous false positives, (Type I Errors) are detected. Yet for RNA-seq data, typically only a small percentage of the transcripts are truly differentially expressed and thus few statistical tests should produce small p-values.

Bourgon et al. [5] show that pruning the dataset of transcripts that have little or no chance of producing low p-values can reduce the loss of power that occurs with multiple testing [27]. Transcripts can be filtered out either prior to or following testing, and in practice there are several methods can be used to achieve this. Still, Rau et al. [36] state little attention has been paid to the choice of the type of filter or threshold used or its impact on the downstream analysis.

The lack of clearly defined filter cut-off for threshold-based filtering methods in general provided motivation for this experiment. In particular we want to determine if the filter cut-off has any effect on a statistical method's ability to detect differential expression. We

use **sensitivity** and **specificity** as the performance measures for assessing this criteria. We define sensitivity as the proportion of correctly unfiltered genes (i.e. those tested DE) among all truly DE genes, and specificity as the proportion of correctly filtered genes (i.e. those tested non-DE or filtered from the dataset) among all non-DE genes, as did Rau et al [36].

The statistical methods used to analyze the data were DESeq2, EdgeR, EdgeR Robust and SAMSeq. BaySeq was initially included but then subsequently removed from the study due to the high computation requirements. The BaySeq pipeline is included in Subsection 8.1.2 below purely for interest.

8.1.2 Methods

The data were artificially produced using a simulator created by Mark Robinson, author of the R package, EdgeR [39]. This simulation method is the second method outlined in Chapter 7. We simulated 18000 datasets consisting of either 50000, 100000 or 150000 transcripts, with either 0 or 10% outliers over six control and six treatment replicates. The relative expression level of truly DE features varied by either two, three or six fold change. Ten percent of transcripts were truly differentially expressed, half of them up-regulated. Simulations were repeated one hundred times for each simulation setting. Examination of the dataset for replicate quality was considered unnecessary.

The independent variable of interest for this experiment is the CPM filter cut-off, x, used in the pre-analysis and post-analysis filtering stage. Ten evenly spaced cut points were used with $x \in [0, 4]$. Read counts were scaled as *parts per million* over the library size for each sample. The dataset was subsequently pruned by keeping only those transcripts with read counts greater than the filter cut-off, x, in at least six samples. As dispersion and mean estimates for the simulated data were based on those of a pre-filtered RNA-seq dataset, the simulated data contained less transcripts exhibiting low levels of abundance than would typically be seen.

Differential expression was then run on all datasets produced under the varying simulation parameters for each cut-off value. Results were compiled for each package and filtering stage (pre- and post-analysis) and performance measures (sensitivity and specificity) were calculated.

Filtering and Normalization

At the time of simulation, library sizes between one and eight million were randomly assigned to replicates within a dataset. The data was not subjected to further normalization for several reasons. First, all five methods include procedures to normalize the data prior to analysis. Normalization would have been recommended if the dataset were being examined for replicate quality, however it was assumed that the simulator produced a quality dataset. As pre-analysis and post-analysis filtering was performed using the CPM method, filtering was effectively based on normalized counts.

Neither of the R packages, Bayseq [17] nor SAMSeq [45] provided an option to apply filtering within the methods, instead the analysis is performed on the full dataset. An option is available within Bayseq to provide a subset of data upon which the prior estimate calculations can be based, however this was not used.

Dispersion estimates in the packages EdgeR [39] and Robust EdgeR [39] are calculated based on a subset of transcripts which meet some cut-off criteria. Transcripts whose sum of normalized counts across all replicates are greater than or equal to five were not used to perform these calculations. The full dataset was fit using the calculated dispersion estimates.

The R package `genefilter` is included in DESeq2 to provide optional independent filtering during analysis. However, the option was set to `FALSE` for this experiment. DESeq2 internally pre-filters transcripts whose sum of counts across all replicates is zero. Additionally, by default, a test for genes containing count outliers, as identified using Cook's distance, is performed post-analysis. The $.99th$ quantile of the $F(p, m-p)$ distribution is used as the default cut-off, where p is the number of coefficients being fitted and m is the number of samples. The p-values for outlier transcripts are set to `NA` within the function `results`.

Workflow

BaySeq The Bayseq pipeline was run on the filtered counts. Differential expression criteria was determined to be those transcripts whose FDR was less than 0.15, as with the WEC data. The model tested was that of differential expression.

```
bayReplicates = c(rep("C",reps_group1), rep("T",reps_group2))
```

```
bayGroups = list(NDE=c(rep(1,reps_group1+reps_group2)),
            DE=c(rep(1,reps_group1), (2,reps_group2)))

bayObj = new("countData", data=as.matrix(fcounts), groups=bayGroups,
            replicates=bayReplicates)

libsizes(bayObj)=getLibsizes(bayObj)
```

The `getPriors.NB` function was run using a quasi-likelihood estimation of priors. The sample size taken from the dataset for estimation varied depending on the size of the dataset. A sample of size 100 was taken from the smaller dataset. A sample of size 10000 was used.

```
bayObj = getPriors.NB(bayObj, samplesize=bay_samp_sz, estimation="QL", cl=NULL)
```

The posteriors likelihoods were estimated using **Bayesian Information Criterion (BIC)** method of estimation. The function `getLikelihoods` attempts to estimate the prior likelihoods by using the BIC to identify the proportion of the data best explained by each model and taking these proportions as prior.

```
bayObj<-getLikelihoods(bayObj, pET='BIC', cl=NULL)
```

Differential expressed transcripts were found using the function `topCounts` from the BaySeq package. `topCounts` takes a `countData` object containing posterior likelihoods and returns the counts with highest (or lowest) likelihood of association with a given group, along with their counts and associated likelihood of differential expression. The user can specify how many transcripts are returned and place a restriction on the group ordering (i.e. up-regulated, down-regulated or both). Alternatively, if specified, the function can return a subset of transcripts with FDR or FWER above a specified certain value.

EdgeR and Robust EdgeR edge R and edgeR robust pipelines were run on the filtered counts. Differential expression criteria was determined to be those transcripts whose FDR was less than 0.15, as with the WEC data. The model tested was that of differential expression.

```
   group = factor(c( rep("Control",reps_group1), rep("Treated",reps_group2)))
y_f = DGEList(counts=fcounts, genes=rownames(fcounts), group=group
```

The negative binomial generalized log-linear models were fit to the transcripts, and tests for differential expression were conducted. DE transcripts were those with FDR > 0.15.

```
fit=glmFit(y_f,design)

lrt=glmLRT(fit, contrast=c(-1,1))
```

Differentially expressed transcripts were found by edgeR and robust edgeR by calling the `topTags` function. This function extracts the top differentially expressed transcripts in a data frame for a given pair of groups, ranked by p-value or absolute log-fold change. The function takes a DGE object, which is a data frame containing the log-concentration (i.e. expression level), the log-fold change in expression between the two groups and the p-value for differential expression for each transcript.

DeSeq2 Deseq2 pipeline was run on the filtered counts. Differential expression criteria was determined to be those transcripts whose adjusted p-value was less than 0.15, as with the WEC data. Independent filtering was applied prior to analysis using the wrapper function `results` which calls the `filtered_p` function of the `genefilter` package. The model tested was that of differential expression.

```
colData<- data.frame(condition=factor(c(rep("Control",reps_group1),
            ("Treated", reps_group2))))

dds=DESeqDataSetFromMatrix(countData=fcounts, colData=colData,
            formula( condition))

res=results(dds, independentFiltering = FALSE)
```

After analysis, we ran the `results` function on the `DESeqDataSet` object of transcripts counts. This extracts a table of analysis results which include the base means across samples, $log2$ fold changes, standard errors, test statistics, p-values and adjusted p-values for each transcript. After ordering the results by decreasing p-value, the top 10 transcripts with the smallest p-values were selected.

SAMSeq Differential expression criteria for SAMseq analysis was determined to be those transcripts whose FDR was less than 0.15, as with the WEC data. The model tested was

that of differential expression.

```
samModel=c(rep(1,reps_group1), rep(2,reps_group2))

samfit<-SAMseq(x=fcounts, y=samModel, resp.type="Two class unpaired",
            fdr.output=cut-off_FDR)
```

Results from the analysis are returned directly from the call to `SAMseq`. Differentially expressed genes are listed in the return object's elements `genes.lo` and `genes.up`.

```
siggenes<-rbind(samfit$siggenes.table[1]$genes.lo, samfit$siggenes.table[1]$genes.up)
```

Because the statistical test for DE is non-parametric and uses a Wilcoxon statistic, there is no p-value or likelihood on which to order the results.

Performance Measure Calculations The following steps were performed for each simulation trial and cut-off value:

1. Analysis results for all filtered and non-filtered transcripts were loaded into tables `samseq`, `edger`, `robust` and `bayseq`. Row entries were set to `NA` for those transcripts that had been filtered out of the dataset.
2. The true differential expression status for each transcript, DE $\in [0, 1]$, was stored in vector `true`.
3. The differential expression results status for each transcript, DE $\in [0, 1]$, was added into a new column. DE $= 0$ for filtered transcripts.
4. Sensitivity and specificity were calculated using functions `sensitivity` and `specificity`, available from the R package, `caret`.
5. Results were compiled into a single file; one set of performance measures for each simulation trial.

8.1.3 Results and Discussion

Performance Measures Plots - Pre-Filtered Data

Effect of CPM cut-off on performance measures The effect of filter cut-off on package performance is seen in Figures 8.1 and 8.2. Ten CPM cut-off values, $x|x \in [0, 4]$, are located along the x-axis, and the calculated performance measure along the y-axis.

In Figure 8.1 it can be seen that sensitivity decreases as the CPM cut-off increases for EdgeR, Robust EdgeR and DESeq2. Sensitivity ranges from approximately 0.7 to 0.3 over the domain of the CPM cut-off for these packages.

SAMSeq is less affected by the cut-off, however the sensitivity for all simulations analyzed with SAMSeq is quite low. Truly differentially expressed are much less likely to be found by SAMSeq.

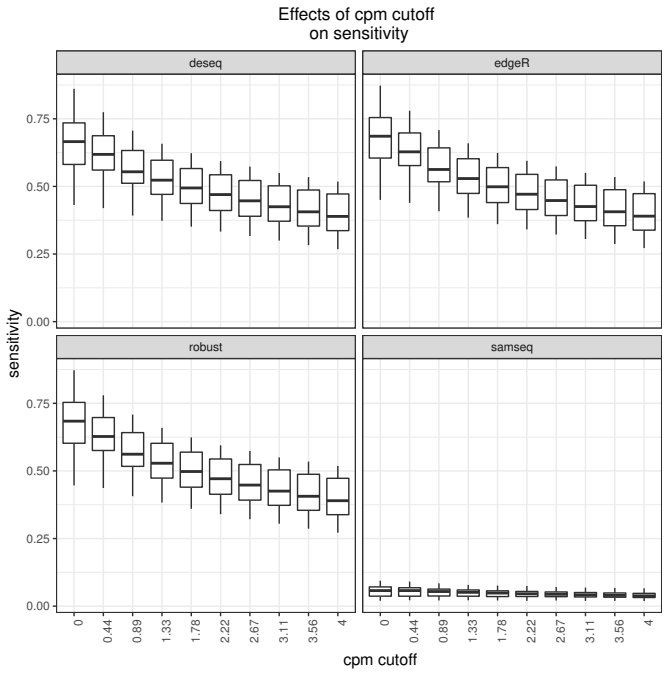

Figure 8.1: Box plots of various filter cut-offs on sensitivity results for four R packages. The filtering step was performed before analysis using the CPM method detailed in Chapter 5. Sensitivity decreases with cut-off value for all four packages.

Specificity increases with the CPM cut-off for all four packages, ranging from approxim-

ately 0.88 to 0.94 for three of the four packages (Figure 8.2). SAMSeq results appear less affected by the cut-off value, ranging from approximatley 0.94 to 0.96. All curves appear quadratic in nature, flattening out as the cut-off approaches 4.

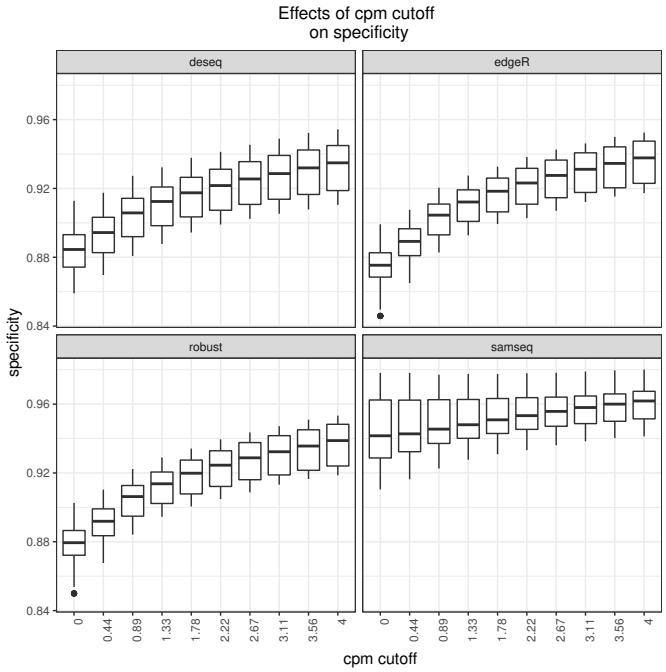

Figure 8.2: Box plots of various filter cut-offs on specificity results for four R packages. The filtering step was performed before analysis using the CPM method detailed in Chapter 5. Specificity increases with cut-off value for all four packages.

Effect of size of dataset on performance measures The effect of the size of the dataset on package performance is shown in Figures 8.3 and 8.4. Three sizes of datasets were simulated (50000, 10000 and 150000). These values are located along the x-axis with the calculated performance measure along the y-axis.

In Figure 8.3 sensitivity appears to decrease as the dataset size increases for EdgeR, Robust EdgeR and DESeq2. Sensitivity ranges from just above 0.5 to approximately 0.4 over the domain of dataset size for these packages. The filter cut-off may have a small negative effect on sensitivity for the package SAMSeq, however the sensitivity for all simulations analyzed with SAMSeq is quite low so this is difficult to determine from the plot.

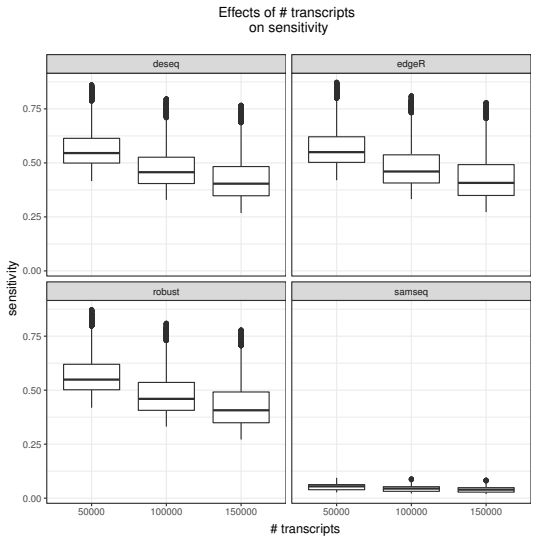

Figure 8.3: Box plots of the effect of dataset size on sensitivity results for four R packages. Datasets of 50000, 100000 and 150000 were analyzed for performance. Sensitivity decreases for three of the four packages.

Specificity increases with the number of transcripts analyzed for all four packages (Figure 8.4). The specificity ranges from approximately 0.90 to 0.93 for three of the four packages. SAMSeq specificity appears the least affected by dataset size, ranging from approximately 0.94 to 0.96.

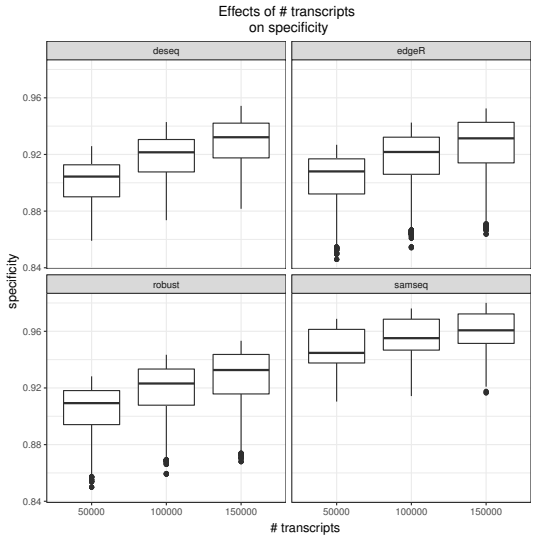

Figure 8.4: Box plots of the effect of dataset size on specificity for four R packages. Datasets of 50000, 100000 and 150000 were analyzed for performance. Specificity increases for all four packages.

Effect of fold change on performance measures The effect of the fold change on package performance is seen in Figures 8.5 and 8.6. Fold changes two, three or six refer to the approximate fold change for genes simulated as DE. The fold changes are located along the x-axis, and the calculated performance measure along the y-axis.

In Figure 8.5 sensitivity increases slightly with the fold change for EdgeR, Robust EdgeR and DESeq2. Sensitivity ranges from just above to just below 0.5 over the domain of fold change values for these packages. Fold change appears to have a small positive effect on sensitivity for the package SAMSeq, however the sensitivity for all simulations analyzed with SAMSeq is quite low so this is difficult to determine from the plot.

Figure 8.6 shows specificity appears fairly unaffected by changes in relative expression EdgeR, Robust EdgeR and DESeq2. Specificity decreases with fold change for SAMSeq, ranging from approximately 0.97 to 0.95.

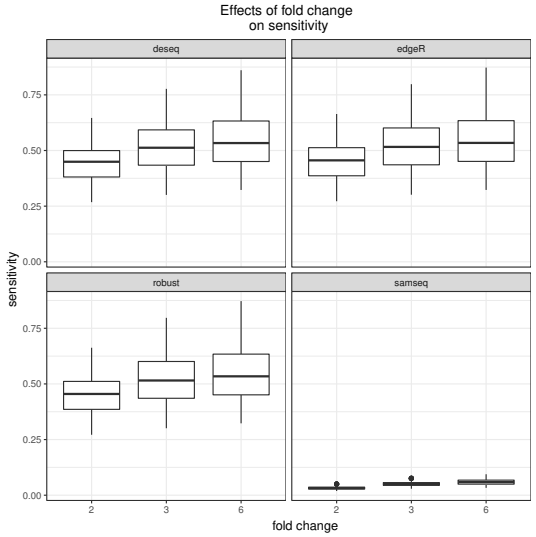

Figure 8.5: Box plots of fold change on the sensitivity of all four R packages. Fold change for DE transcripts is either two, three or six. Sensitivity increases slightly for all four packages.

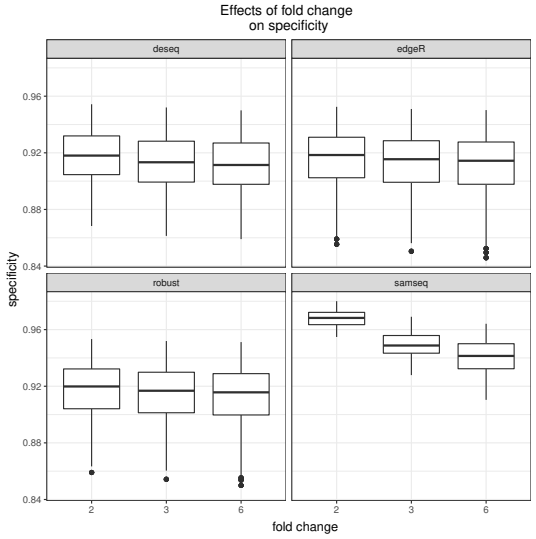

Figure 8.6: Box plots of fold change on specificity results for four R packages. Fold change for DE transcripts is either two, three or six. Specificity remains stable (constant) for three of the four packages, and appears to have a negative effect on the specificity of SAMSeq.

Effect of outliers on performance measures Outliers were added to some of the datasets to simulate real RNA-seq data. The percentage of outliers for each dataset was either 0 or 10%. The effect of the presence of outliers on sensitivity and specificity can be seen in Figures 8.7 and 8.8. The percentage of outliers is located along the x-axis, and the calculated performance measure along the y-axis.

From Figures 8.7 and 8.8 the percentage of outliers in the dataset appears to have no effect on either sensitivity or specificity for any of the four packages.

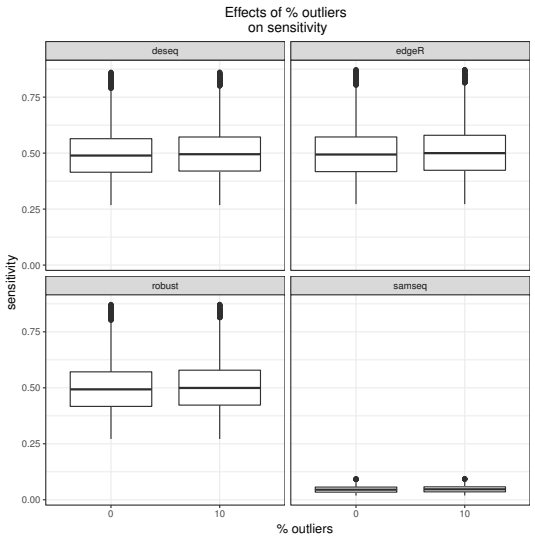

Figure 8.7: Box plots of % outliers on sensitivity results for four R packages. Datasets with 0 and 10% outliers were analyzed for performance. Sensitivity remains stable (constant) for all 4 packages.

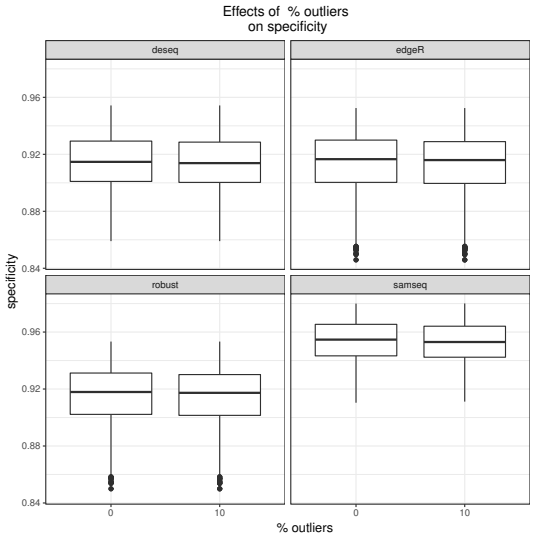

Figure 8.8: Box plots of % outliers on specificity results for four R packages. Datasets with 0 and 10% outliers were analyzed for performance. Specificity remains stable (constant) for all 4 packages.

Interaction effects on performance measures A 2-way interaction effect exists when there is a change in the effect of one variable over the levels of another variable. When interaction effects exist, the means of each variable cannot be modeled simply by knowing the size of the main effects, instead, another parameter is required to explain the differences in means. Interaction effects can be visualized in an **interaction plot**, where the geometric relationship between lines of a factor level indicate presence of an interaction effect. If the lines describing the main effects are not parallel, then the presence of an interaction effect is possible.

Figures 8.9 and 8.10 show the effect of fold change on sensitivity and specificity, respectively, over the levels of CPM cut-off. The lines in both plots are non-parallel for all packages. This suggests there is an interaction effect between fold change and the cut-off value for both performance measures.

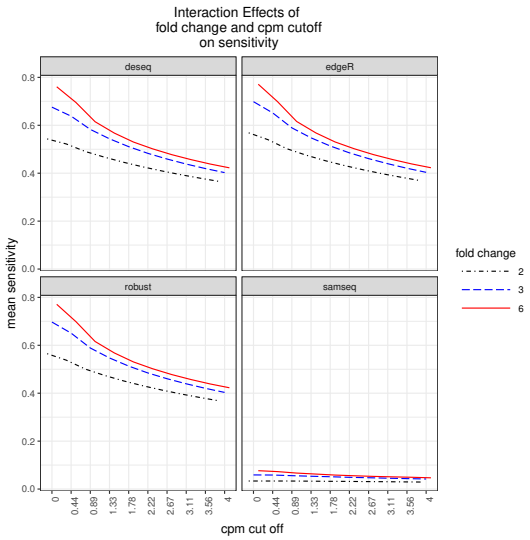

Figure 8.9: Interaction of effects of fold change over the levels of cut-off for four R packages. Non-parallel lines suggest there is an interaction effect between these variables.

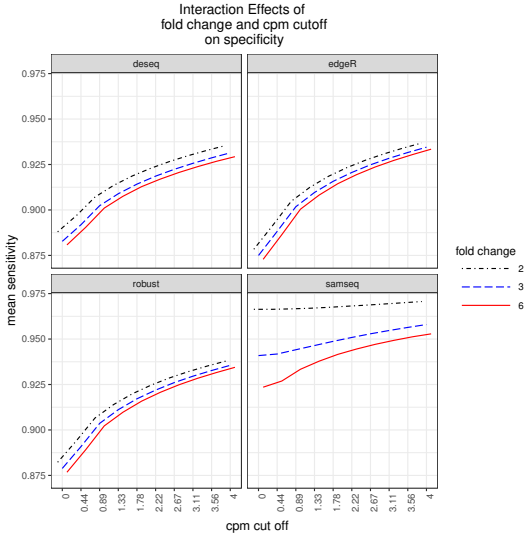

Figure 8.10: Interaction of effects of fold change over the levels of cut-off for four R packages. Non-parallel lines suggest there is an interaction effect between these variables.

Figures 8.11 and 8.12 show the interaction effect of the number of transcripts analyzed on sensitivity and specificity, respectively, over the levels of fold change. The lines in both plots are fairly parallel for all packages. This suggests there is minimal interaction effect between fold change and the dataset size for both performance measures.

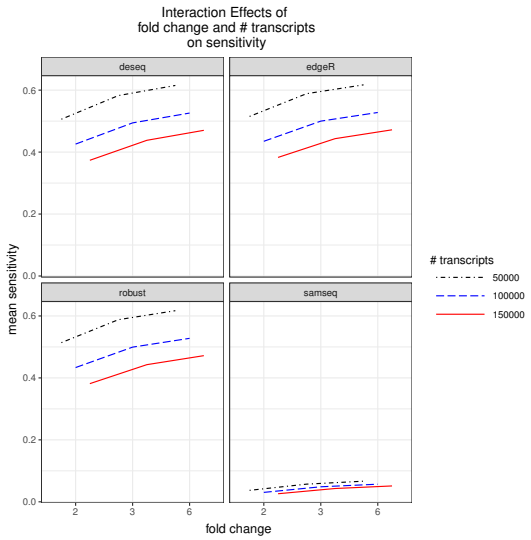

Figure 8.11: Interaction of effects of dataset size over the levels of fold change for four R packages. Parallel lines suggest only minimal interaction effect exists.

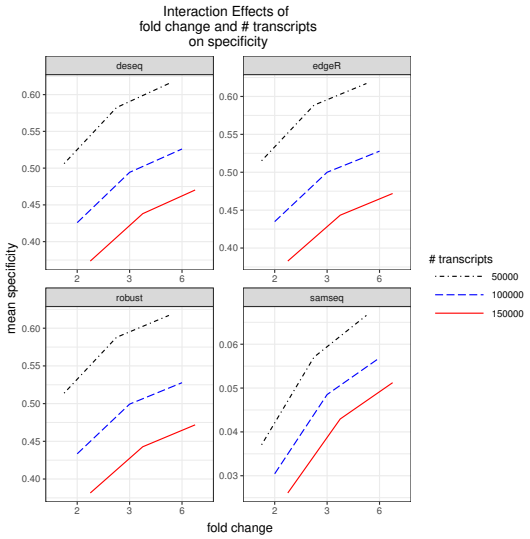

Figure 8.12: Interaction of effects of dataset size over the levels of fold change for four R packages. Parallel lines suggest only minimal interaction effect exists.

Figures 8.13 and 8.14 show the interaction effect of the number of transcripts analyzed on sensitivity and specificity, respectively, over the levels of CPM cut-off. The lines in both plots are non-parallel for all packages. This suggests there is an interaction effect between the number of transcripts analyzed and the cut-off value on both sensitivity and specificity. The interaction effect of the number of transcripts analyzed and the CPM cut-off on sensitivity is hard to determine for SAMseq, due to the extremely low mean sensitivity.

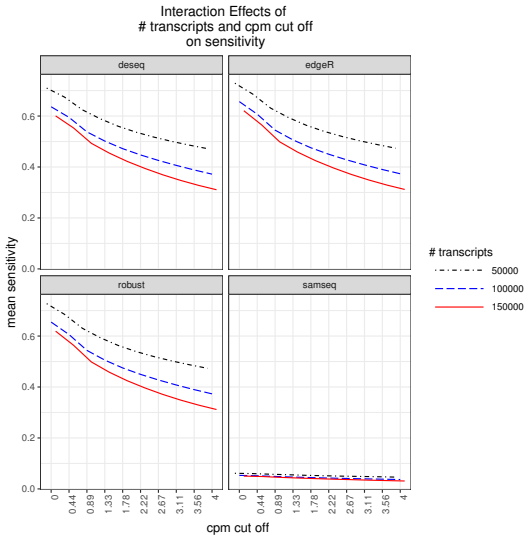

Figure 8.13: Interaction of effects of dataset size over the levels of cut-off for four R packages. Non-parallel lines suggest there is an interaction effect between these variables.

Performance Measures Plots - Post-Filtered Data

Effect of CPM cut-off on performance measures The effect of filter cut-off on package performance is seen in Figures 8.15 and 8.16. Ten CPM cut-off values, $x|x \in [0, 4]$, are located along the x-axis, and the calculated performance measure along the y-axis.

In Figure 8.15 it can be seen that sensitivity decreases as the CPM cut-off increases for all four packages. Sensitivity ranges from approximately 0.7 to 0.3 over the domain of the cut-off for EdgeR, Robust EdgeR and DESeq2. SAMSeq is less affected by the cut-off; the sensitivity for this package is in the range of approximately 0.5 to 0.3. The spread of sensitivity for SAMSeq is relatively large for lower values of the cut-off.

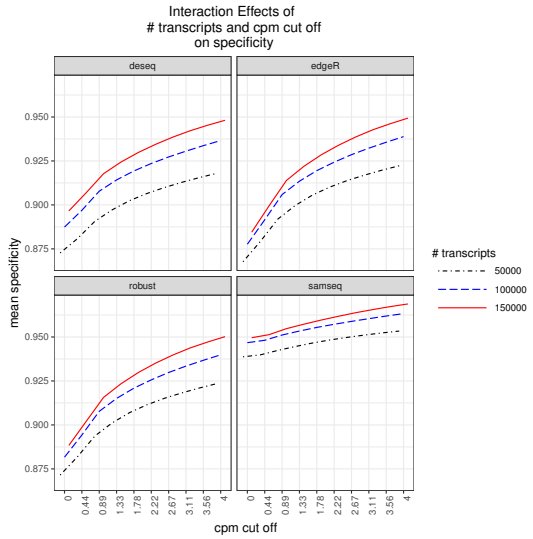

Figure 8.14: Interaction of effects of dataset size over the levels of cut-off for four R packages. Non-parallel lines suggest there is an interaction effect between these variables.

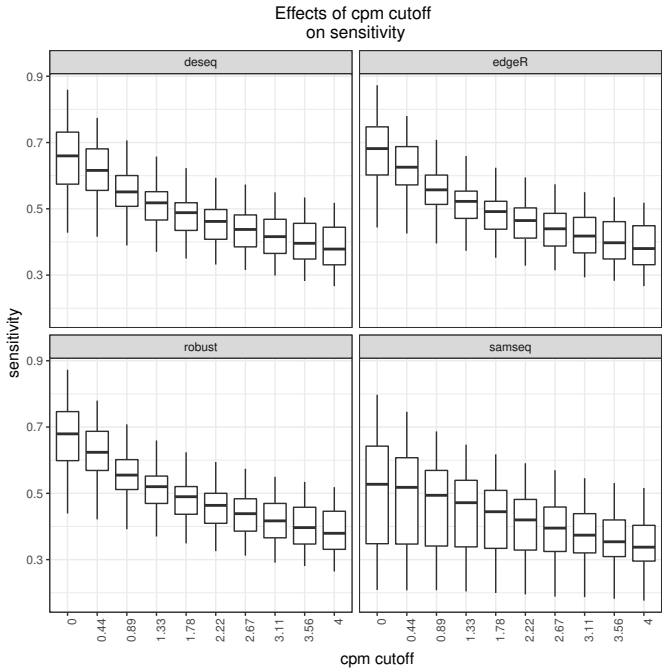

Figure 8.15: Box plots of various filter cut-off values on sensitivity results for four R packages. The filtering step was performed after analysis using the CPM method detailed in Chapter 5. Sensitivity decreases with cut-off value for all four packages.

Specificity increases as the cut-off increases for EdgeR, Robust EdgeR and DESeq2 (Fig-

ure 8.16), ranging from approximately 0.88 to 0.95. The specificity for all simulations analyzed with SAMSeq is extremely high, therefore a change in cut-off value can have little effect on specificity.

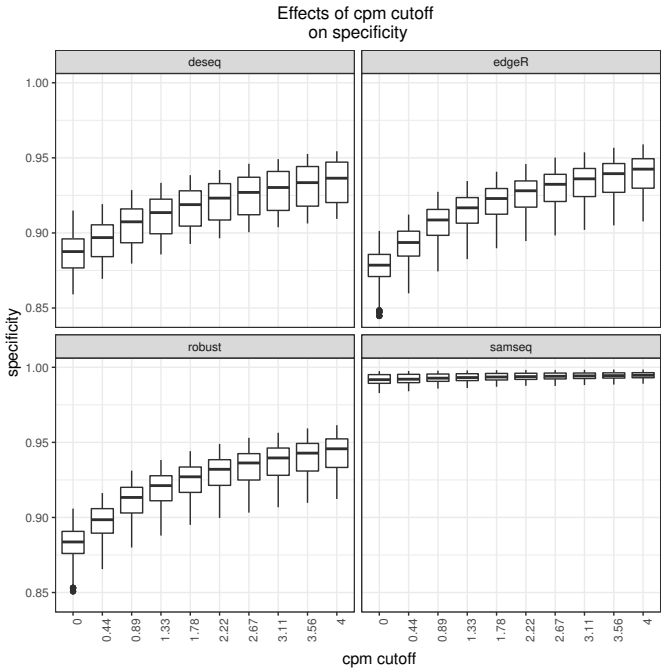

Figure 8.16: Box plots of various filter cut-off values on specificity results for four R packages. The filtering step was performed after analysis using the CPM method detailed in Chapter 5. Specificity increases with cut-off value for all four packages.

Effect of size of dataset on performance measures The effect of the size of the dataset on package performance is shown in Figures 8.17 and 8.18. Three sizes of datasets were simulated (50000, 10000 and 150000). These values are located along the x-axis with the calculated performance measure along the y-axis.

In Figure 8.17 sensitivity appears to decrease as the dataset size increases for all four packages. Sensitivity ranges from just above 0.5 to approximately 0.4 over the domain of dataset size for three of the packages. For SAMSeq, the range is slightly lower, and the spread for sensitivity is wider.

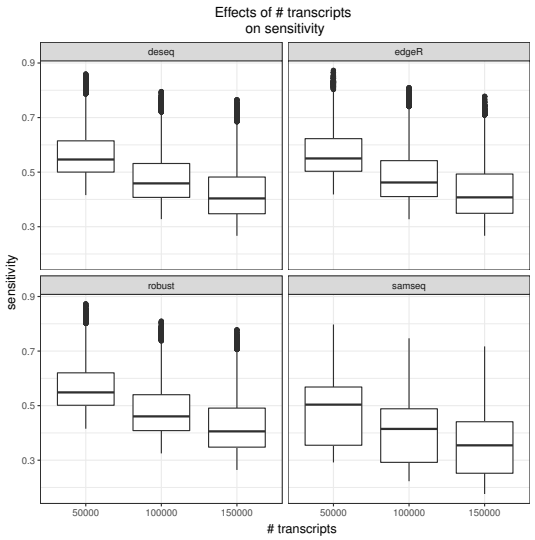

Figure 8.17: Box plots of various dataset sizes on sensitivity results for four R packages. Datasets of 50000, 100000 and 150000 were analyzed for performance. Sensitivity decreases with the number of transcripts analyzed for all four packages.

Specificity appears to increase as the dataset size increases for EdgeR, Robust EdgeR and DESeq2 (Figure 8.18). For these packages, specificity ranges from just above 0.90 to approximately 0.94 over the domain of dataset size. The filter cut-off may have a small effect on specificity for the package SAMSeq, however this is difficult to determine from the plots as specificity for all simulations analyzed with SAMSeq is quite high, and the spread is very small.

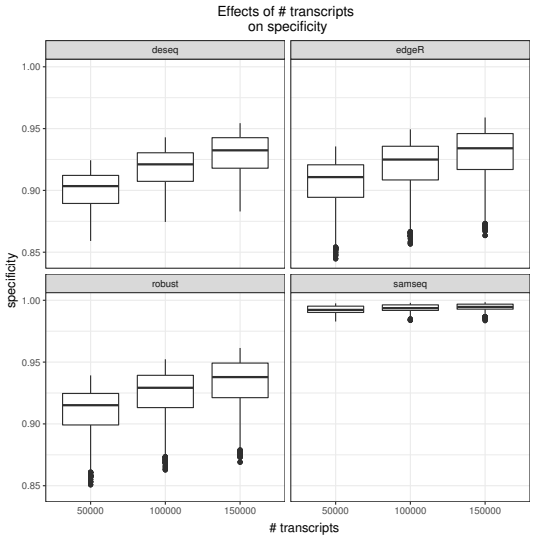

Figure 8.18: Box plots of various dataset sizes on specificity results for four R packages. Datasets of 50000, 100000 and 150000 were analyzed for performance. Specificity increases with the number of transcripts analyzed for all four packages.

Effect of fold change on performance measures The effect of the fold change on package performance is shown in Figures 8.19 and 8.20. Fold changes of two, three and six refer to the approximate fold change for genes simulated as DE. The fold changes are located along the x-axis, and the calculated performance measure along the y-axis.

Sensitivity appears to increase as the dataset size increases for all four packages (Figure 8.17). For EdgeR, Robust EdgeR and DESeq2, sensitivity ranges from approximately 0.45 to just above 0.50 over the domain. Sensitivity ranges over a higher interval for the package SAMSeq.

From Figure 8.20, the fold change for truly DE transcripts appears to have little effect on either sensitivity or specificity for EdgeR, Robust EdgeR and DESeq2. Increasing fold change appears to have a slight negative effect on specificity for the package SAMSeq. Although the specificity for all simulations analyzed with SAMSeq is quite high, the spread of the data is very small. A small change in specificity may be significant.

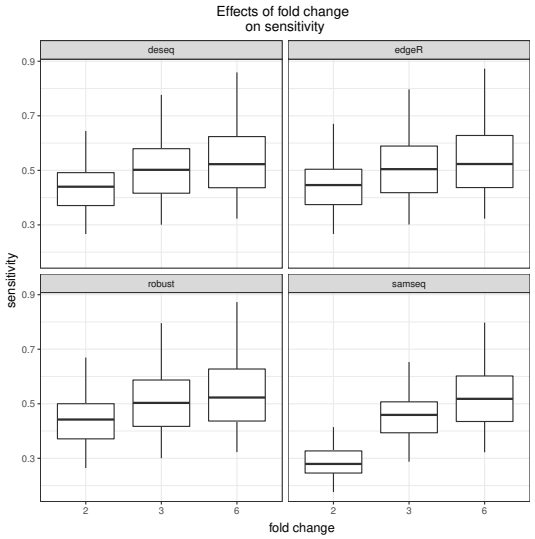

Figure 8.19: Box plots of fold change on sensitivity results for four R packages. Fold change for DE transcripts was either two, three and six. Sensitivity increased for all 4 packages.

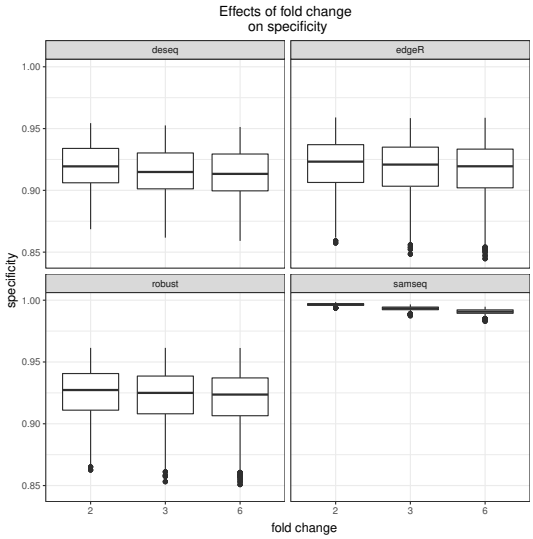

Figure 8.20: Box plots of fold change on specificity results for four R packages. Fold change for DE transcripts was either two, three and six. Specificity remains stable (constant) for 3 of the 4 packages.

Effect of outliers on performance measures Outliers were added to some of the datasets to simulate real RNA-seq data. The percentage of outliers for each dataset was either 0 or 10%. The effect of the presence of outliers on sensitivity and specificity can be seen in Figures 8.21 and 8.22. The percentage of outliers is located along the x-axis, and the calculated performance measure along the y-axis.

From Figures 8.21 and 8.22 the percentage of outliers in the dataset appears to have no effect on either sensitivity or specificity for any of the four packages.

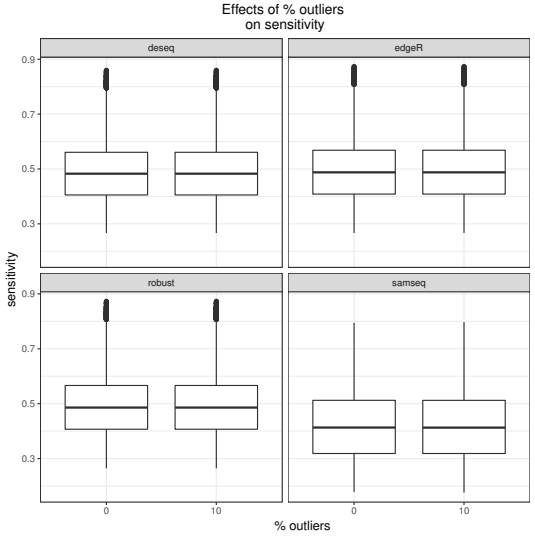

Figure 8.21: Box plots of % outliers on sensitivity results for four R packages. Datasets with 0 and 10% outliers were analyzed for performance. Sensitivity remains stable (constant) for all 4 packages.

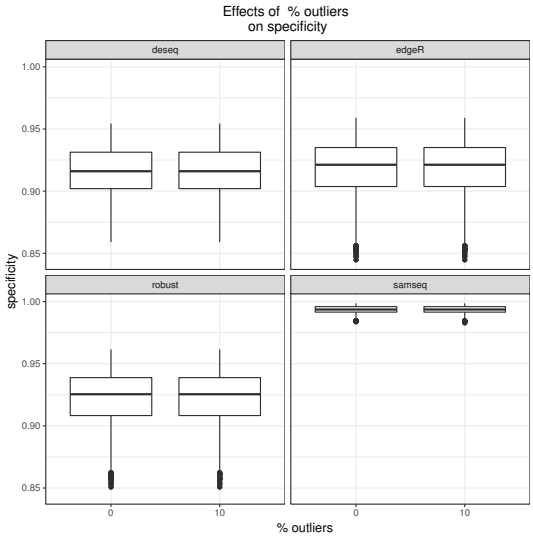

Figure 8.22: Box plots of % outliers on specificity results for four R packages. Datasets with 0 and 10% outliers were analyzed for performance. Specificity remains stable (constant) for all 4 packages.

Interaction effects on performance measures A 2-way interaction effect exists when there is a change in main effect of one variable over the levels of another variable. When there is a significant interaction effects between two explanatory variables, the means of each variable cannot be modeled by considering the main effects alone. Instead, another parameter is required to explain the differences in means. Interaction effects can be visualized in an **interaction plot**, where the geometric relationship between lines of a factor level can alert us to the presence of an interaction effect. If the lines describing the main effects are not parallel, then the presence of an interaction effect is possible.

Figures 8.23 and 8.24 show the effect of fold change on sensitivity and specificity, respectively, over the levels of CPM cut-off. The lines in Figure 8.23 are non-parallel for all packages. This suggests there is an interaction effect between fold change and the cut-off value on sensitivity.

The lines in Figure 8.24 are close to parallel for all packages. This suggests there is minimal or no interaction effect between fold change and the cut-off value on sensitivity.

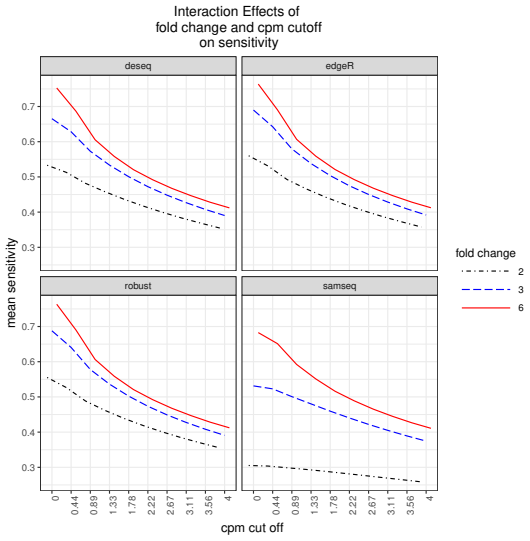

Figure 8.23: Interaction of effects of fold change over the levels of cut-off for four R packages. Non-parallel lines suggest there is an interaction effect between these variables.

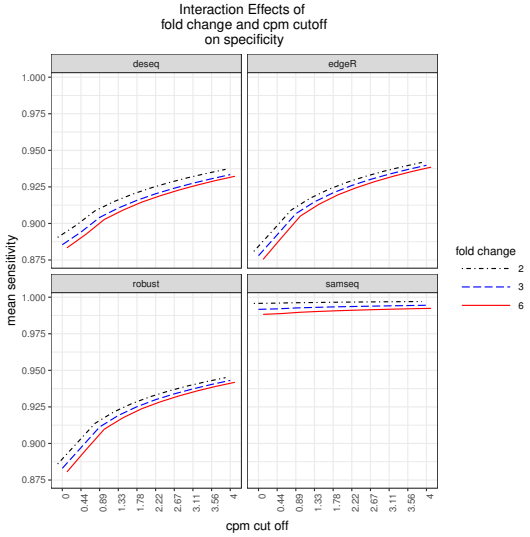

Figure 8.24: Interaction of effects of fold change over the levels of cut-off for four R packages. Parallel lines suggest only minimal interaction effect exists.

Figures 8.25 and 8.26 show the interaction effect of the number of transcripts analyzed (50000, 100000, 150000) on sensitivity and specificity, respectively, over the levels of fold change (2, 3, 6). The lines in both plots are fairly parallel for all packages. This suggests there is minimal interaction effect between fold change and the dataset size for both performance measures.

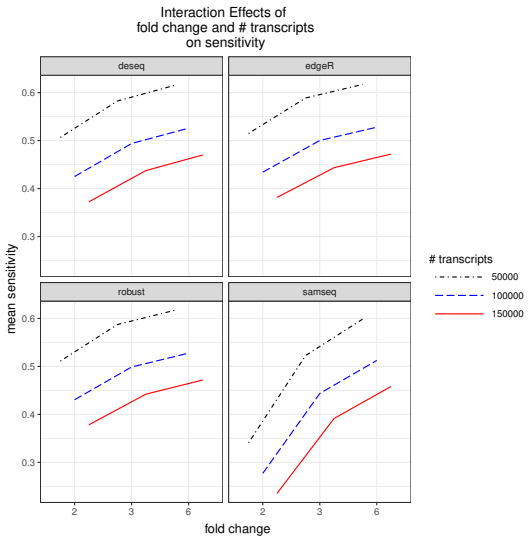

Figure 8.25: Interaction of effects of dataset size over the levels of fold change for four R packages. Lines are close to parallel suggesting only minimal interaction effect exists.

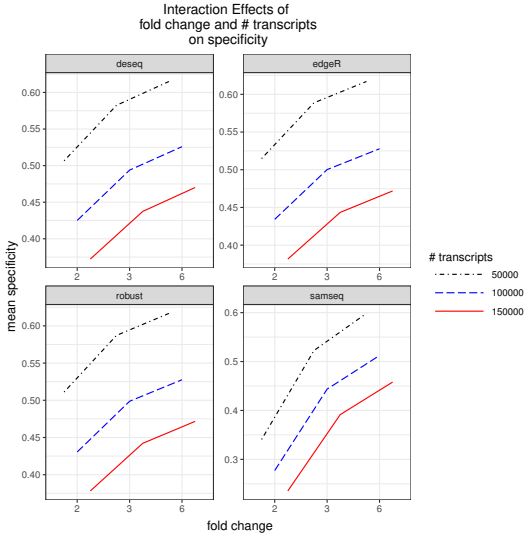

Figure 8.26: Interaction of effects of dataset size over the levels of fold change for four R packages. Lines are close to parallel suggesting only minimal interaction effect exists.

Figures 8.27 and 8.28 show the changes in the effect of dataset size on sensitivity and specificity, respectively, over the levels of CPM cut-off. The lines in both plots are non-parallel for all packages. This suggests there is an interaction effect between the number of transcripts analyzed and the cut-off value on both sensitivity and specificity.

The interaction effect of the number of transcripts analyzed and the CPM cut-off on specificity may be smaller for SAMseq, as the lines are close to overlapping.

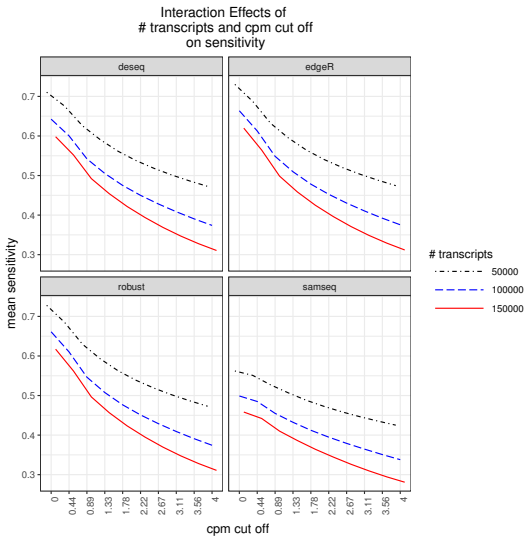

Figure 8.27: Interaction of effects of dataset size over the levels of cut-off for four R packages. Non-parallel lines suggest there is an interaction effect between these variables.

Data Models of Performance Measures

We wish to determine if the choice of filter cut-off value has a significant effect on the performance of statistical packages used to analyze RNA-seq data. To achieve this, the analysis results from DeSeq2, SAMSeq, EdgeR and Robust EdgeR on 18000 simulated datasets were compiled. Sensitivity and specificity results were calculated for each package under varying simulation conditions. See section 7 for more details. Filtering was done both prior to and after analysis for comparison.

Plots of perfomance measures against the various simulation settings and filter cut-off values (Figures 8.1 to 8.28 above), suggest that the size of dataset, choice of filter cut-off,

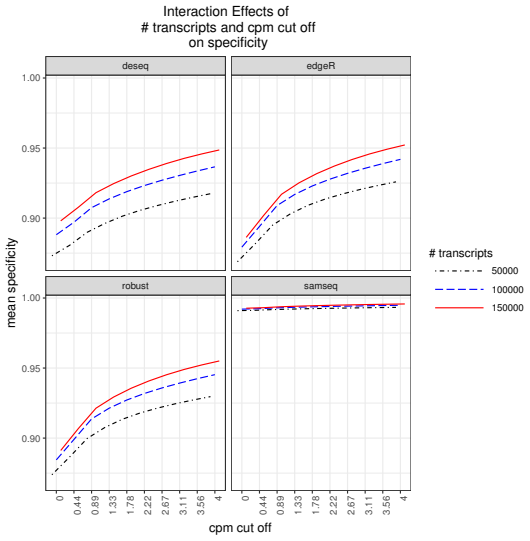

Figure 8.28: Interaction of effects of dataset size over the levels of cut-off for four R packages. Non-parallel lines suggest there is an interaction effect between these variables.

and fold change of truly DE transcripts may have an effect on sensitivity and specificity. The percentage of outliers in the dataset do not appear to have any effect on the performance variables (Figures 8.7, 8.8, 8.21 and 8.22). The plots suggest that both size of dataset and fold change may interact with the effect of filter cut-off on sensitivity and specificity.

To test these effects, we model both perfomance measures using a **generalized linear model** of the form

$$g(\pi_i) = \mathbf{x}_i^T \beta \tag{8.1}$$

Where $\mathbf{x}_i$ is the vector of explanatory variables, β is a vector of parameters and $g(\pi_i)$ is a link function. The explanatory variables are:

- Number of transcripts analyzed. Either 50000, 100000, or 150000.
- Percentage of outliers in the dataset. Either 0 or 10.
- Fold change for the truly DE transcripts. Either 2, 3 or 6.
- Ten evenly distributed filter cut-off values. $x|x \in (0, \cdots, 4)$

As these variables are categorical, each is a represented as factor. Dummy variables are used to model the different levels within each factor. Therefore, the full model (considering main effects only) will have $p = 18 - 1$ parameters, one for each factor level.

The response variables, sensitivity and specificity, represent proportions of the total genes analyzed. These are modelled as $P_i = Y_i/n_i$, $i = (1, \cdots, 18000)$, where Y_i is the number of successes in n_i trials for covariate grouping i. The random variable Y_i is distributed as $\text{Bin}(n_i, \pi_i)$:

$$Pr(Y_i = y_i) = \binom{n_i}{y_i} \pi_i^{y_i} (1 - \pi_i)^{n_i - y_i} \tag{8.2}$$

Since π is bounded by the interval, $[0, 1]$, we map it to $(-\infty, \infty)$ using a **tolerance function**, $f(s)$.

$$\pi = \int_{-\infty}^{x} f(s)\,\mathrm{d}s = \frac{exp(\beta_1 + \beta_2 x)}{1 + exp(\beta_1 + \beta_2 x)} \tag{8.3}$$

This yields the **logit** link function

$$log(\frac{\pi}{1 - \pi}) = \beta_1 + \beta_2 x \tag{8.4}$$

where

$$log(1 - \pi) = -log[1 + exp(\beta_1 + \beta_2 x)] \tag{8.5}$$

The log likelihood function for the binomial distribution is

$$l(\pi_1 \cdots \pi_N; y_1 \cdots, y_N) = \sum_{i=1}^{N} \left[y_i \log\left(\frac{\pi_i}{1 - \pi_i}\right) + n_i \log(1 - \pi_i) + log\binom{n_i}{y_i} \right] \tag{8.6}$$

The log likelihood is used in inference to assess how well a model fits the data. Equation 8.6 will be refered back to later in the analysis.

GLM Results

Sensitivity and specificity were modelled using all combinations of explanatory variables for each package seperately. This was repeated for each filtering strategy. The output for all 16 models can be seen in Appendix D. To illustrate the process, we derive the sensitivity model for the EdgeR pre-filtered data analysis results. The workflow is shown below.

The combined sensitivity and specificity calculation results for the pre-filtered data for all packages is read into a dataframe. Number datatypes are declared for sensitivity and specificity, the others are read into the dataframe as factors.

```
dt<-read.table("./../output/csv/sens_spec_all.csv", header=FALSE, sep=",",
               colClasses=c(rep("factor",5),"numeric","numeric","factor"))
colnames(dt)<-c("ngenes","pout","fold","trial","package","sens",
"spec","cutoff")

str(dt)
'data.frame': 172320 obs. of  8 variables:
 $ ngenes : Factor w/ 3 levels "100000","150000",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ pout   : Factor w/ 2 levels "0","0.1": 1 1 1 1 1 1 1 1 1 1 ...
 $ fold   : Factor w/ 3 levels "2","3","6": 1 1 1 1 1 1 1 1 1 1 ...
 $ trial  : Factor w/ 100 levels "1","10","100",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ package: Factor w/ 4 levels "deseq","edgeR",..: 4 1 2 3 4 1 2 3 4 1 ...
 $ sens   : num  0.033 0.5162 0.5529 0.5506 0.0346 ...
 $ spec   : num  0.966 0.892 0.885 0.889 0.966 ...
 $ cutoff : Factor w/ 10 levels "0","0.44","0.89",..: 1 1 1 1 2 2 2 2 3 3 ...
```

When modelling proportion data, we must give the number of "successes" as well as "failures" in a two-vector response variable. In our case, the sensitivity of the EdgeR test results is defined as TP/P, where TP is the number of true DE transcripts found by EdgeR. P is the number of true DE transcripts in the dataset, which was simulated as 10% of the total number of transcripts. The response vector was back-calculated from the sensitivity results.

```
dt$P=as.numeric(as.character(dt$ngenes))*0.1
```

```
dt$TP=round(dt$sens*dt$P,0)
```

The function `glm` from the R Bioconductor [34] `base` package is used to model the data. The `base` package contains the basic functions which let R function as a language and are available through inheritance from any environment. We fit the intial EdgeR sensitivity model with pre-filtered data, `fit_edgeR`, by issuing following command:

```
fit_edgeR<-glm(cbind(TP,P-TP)~ ngenes+pout+cutoff+fold+
               cutoff:fold+cutoff:ngenes+fold:ngenes,
               family=binomial, subset(dt,package=='edgeR'))
```

The response was the two-column matrix of successes and failures. Explanatory variables included one for each of the main effects and 2-way interactions. As % outliers was suspected not to have an effect on sensitivity, interaction effects with this variable were not included in the model. The main effect of % outliers was included in the initial fit and subsequently removed. Only sensitivity results for EdgeR was considered.

Referring to back to Equation 8.3, we do not model $log(\frac{TP}{P}) = \beta_1 + \beta_2 x$ directly with simple linear regression as the logits for p become infinite as p approaches the extremes of zero and one. As well, we lose information on the number of transcripts from which the proportion was estimated.

Instead, we assume the errors are binomial distributed and pass `family = binomial` to `glm`. This setting uses the `logit` link by default thereby ensuring the response variable is not bounded on [0,1]. R uses the number of transcripts as weights and the logit function to maintain linearity.

The model tests the hypothesis that fold change, number of transcripts in the dataset, CPM cut-off value or % outliers have an effect on the sensitivity. As well, it tests for 2-way interaction effects between cut-off, fold change and number of transcripts.

Model Fit In the previous section, we defined the log likelihood for the binomial distribution (Equation 8.6) as:

$$l(\pi_1 \cdots \pi_N; y_1 \cdots, y_N) = \sum_{i=1}^{N} \left[y_i \log\left(\frac{\pi_i}{1-\pi_i}\right) + n_i \log(1-\pi_i) + log\binom{n_i}{y_i} \right]$$

This equation can be used to derive the AIC and deviance statistics, which provide a way to measure the goodness of fit for the model.

The deviance, or **log-likelihood (ratio) statistic**, is defined as $D = 2log\lambda$, where $\lambda = \frac{L(\mathbf{b_{max}};\mathbf{y})}{L(\mathbf{b};\mathbf{y})}$, $L(\mathbf{b_{max}};\mathbf{y})$ is the likelihood function for the saturated model (all parameters included) and $L(\mathbf{b};\mathbf{y})$ is the likelihood of model *fit_edgeR*. The asymptotic distribution of D, under the hypothesis that the model is correct, is $D \sim \chi^2(N-p)$ [11].

AIC is defined as $-2l(\hat{\boldsymbol{\pi}};\mathbf{y}) + 2p$, where p is the number of parameters in the model and $\hat{\pi}$ is the estimated probablity for Y_i under our model *fit_edgeR*. The fitted values for *fit_edgeR* are $\hat{y}_i = n_i\hat{\pi}_i$ [11].

After running the model, we call `drop1`, a function in `base` R which sequentially determines which single terms can be dropped from the model, fits the models and computes a table of the changes in fit (Table 8.1). The AIC and deviance statistics are reported for each updated model that could be refit from the current one by dropping a single parameter.

```
print(drop1(fit_edgeR))
```

	Df	Deviance	AIC
<none>		218536.36	624956.88
pout	1	218536.37	624954.89
cutoff:fold	18	943053.78	1349438.30
ngenes:cutoff	18	261950.28	668334.80
ngenes:fold	4	231360.32	637772.84

Table 8.1: Summary of the changes in fit that would result from dropping terms from the EdgeR sensitivity model. Only removable terms are shown. Modelled using pre-filtered data.

Table 8.1 lists terms that are removable from the model which include `pout`, and the 2-way interaction effects. As $D \sim \chi^2(N-p)$, we are searching for a model with small deviance and the lowest AIC. This would suggest the p-values are low, and the model fits well. From Table 8.1 it appears that removing any of the interaction effects would result in

a new model with higher AIC and deviance statistics. Removing the main effect `pout` will decrease both AIC and D, which is not suprising as the percentage of outliers was suspected of not having an effect on performance. Removing any one of the interaction effects will not yield a significantly better fitting model.

The model was refit without the main effect of % outliers (`pout`). As interaction terms were left in the model, the updated model could not be reduced further. The final model is $fit1\_edgeR$.

```
fit1_edgeR<-update(fit_edgeR,.~.-pout)
```

Table 8.2 shows the results of the final model fit. The estimated coefficient values, standard errors, and levels of significance are given along with their probability of significance to the model. All terms are significant at the $\alpha = 0.0001$ level. Although the model was reduced, the deviance still appears extremely high for this model ($D = 218536$, Table 8.1). As the number of data points in the dataset is 43080 and the number of parmeters in the model is $53+1$, $D \sim \chi^2(43026)$. However $D = 218536$ is much greater than the upper 99th% point on $\chi^2(43026)$. This suggests the model is overdispersed.

Figure 8.29 shows the final predicted values plotted against the observed sensitivity for the EdgeR pre-filtered data. There is a fair amount of spread in the estimates, likely due to the overdispersion. Similar results were obtained for the sensitivity and specificity models for all packages and both filtering strategies. The final model fits can be found in Appendix D. Although the coefficient estimates appear slightly different for each model, each is best fit with the same explanatory variables.

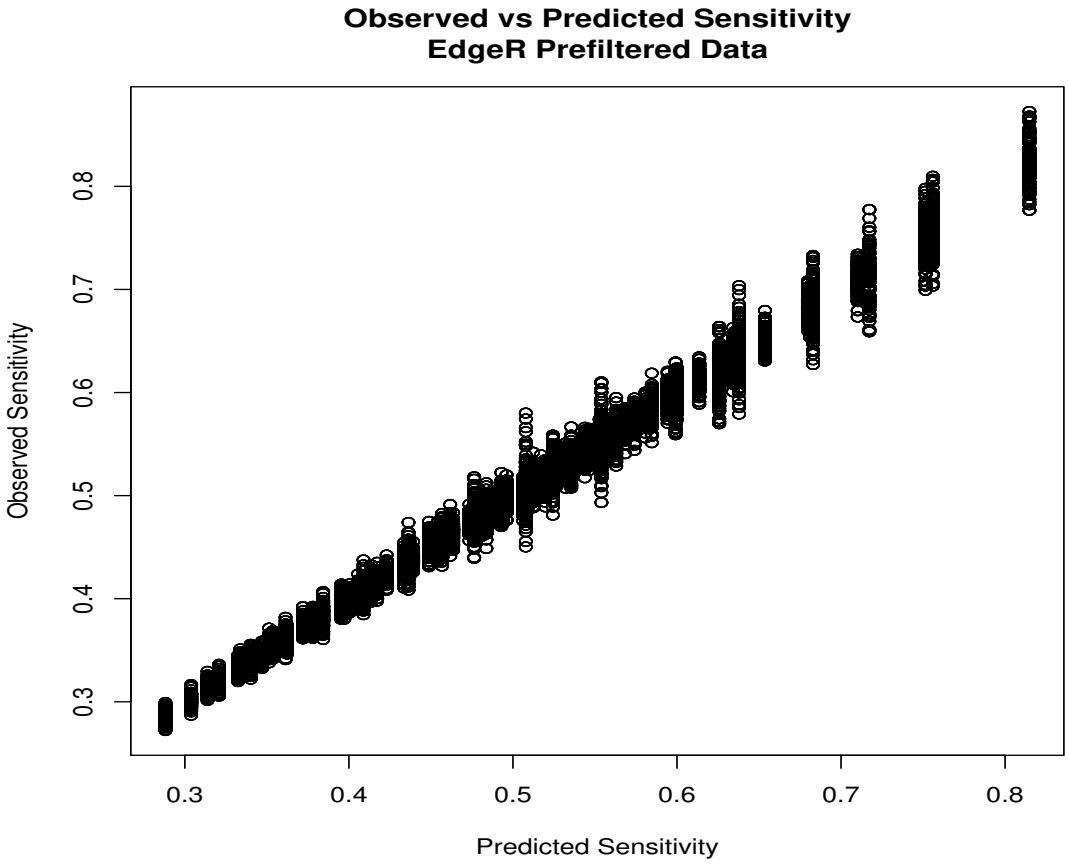

Figure 8.29: Sensitivity vs fitted values for the EdgeR pre-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	0.2164	0.0007	296.81	0.00000
ngenes150000	-0.1850	0.0008	-225.43	0.00000
ngenes50000	0.2975	0.0010	290.39	0.00000
cutoff0.44	-0.1190	0.0010	-119.74	0.00000
cutoff0.89	-0.2814	0.0010	-284.46	0.00000
cutoff1.33	-0.3900	0.0010	-394.29	0.00000
cutoff1.78	-0.4829	0.0010	-487.38	0.00000
cutoff2.22	-0.5660	0.0010	-569.62	0.00000
cutoff2.67	-0.6418	0.0010	-643.72	0.00000
cutoff3.11	-0.7140	0.0010	-713.34	0.00000
cutoff3.56	-0.7819	0.0010	-777.74	0.00000
cutoff4	-0.8484	0.0010	-839.81	0.00000
fold3	0.5509	0.0009	635.15	0.00000
fold6	0.9143	0.0009	996.66	0.00000
cutoff0.44:fold3	-0.0982	0.0011	-87.96	0.00000
cutoff0.89:fold3	-0.2106	0.0011	-189.84	0.00000
cutoff1.33:fold3	-0.2726	0.0011	-245.95	0.00000
cutoff1.78:fold3	-0.3159	0.0011	-284.58	0.00000
cutoff2.22:fold3	-0.3476	0.0011	-312.27	0.00000
cutoff2.67:fold3	-0.3703	0.0011	-331.48	0.00000
cutoff3.11:fold3	-0.3875	0.0011	-345.35	0.00000
cutoff3.56:fold3	-0.4028	0.0011	-357.39	0.00000
cutoff4:fold3	-0.4128	0.0011	-364.41	0.00000
cutoff0.44:fold6	-0.2532	0.0012	-216.87	0.00000
cutoff0.89:fold6	-0.4676	0.0012	-405.46	0.00000
cutoff1.33:fold6	-0.5549	0.0012	-482.23	0.00000
cutoff1.78:fold6	-0.6116	0.0012	-531.21	0.00000
cutoff2.22:fold6	-0.6406	0.0012	-555.21	0.00000
cutoff2.67:fold6	-0.6613	0.0012	-571.37	0.00000
cutoff3.11:fold6	-0.6732	0.0012	-579.52	0.00000
cutoff3.56:fold6	-0.6836	0.0012	-586.14	0.00000
cutoff4:fold6	-0.6863	0.0012	-585.92	0.00000
ngenes150000:cutoff0.44	-0.0071	0.0011	-6.63	0.00000
ngenes50000:cutoff0.44	0.0069	0.0013	5.18	0.00000
ngenes150000:cutoff0.89	-0.0062	0.0011	-5.86	0.00000
ngenes50000:cutoff0.89	0.0204	0.0013	15.54	0.00000
ngenes150000:cutoff1.33	-0.0099	0.0011	-9.42	0.00000
ngenes50000:cutoff1.33	0.0184	0.0013	14.04	0.00000
ngenes150000:cutoff1.78	-0.0198	0.0011	-18.76	0.00000
ngenes50000:cutoff1.78	0.0182	0.0013	13.97	0.00000
ngenes150000:cutoff2.22	-0.0349	0.0011	-32.91	0.00000
ngenes50000:cutoff2.22	0.0237	0.0013	18.19	0.00000
ngenes150000:cutoff2.67	-0.0524	0.0011	-49.27	0.00000
ngenes50000:cutoff2.67	0.0338	0.0013	25.92	0.00000
ngenes150000:cutoff3.11	-0.0674	0.0011	-63.07	0.00000
ngenes50000:cutoff3.11	0.0467	0.0013	35.85	0.00000
ngenes150000:cutoff3.56	-0.0786	0.0011	-73.19	0.00000
ngenes50000:cutoff3.56	0.0625	0.0013	47.90	0.00000
ngenes150000:cutoff4	-0.0874	0.0011	-80.96	0.00000
ngenes50000:cutoff4	0.0779	0.0013	59.56	0.00000
ngenes150000:fold3	-0.0163	0.0006	-28.76	0.00000
ngenes50000:fold3	0.0417	0.0007	61.80	0.00000
ngenes150000:fold6	-0.0148	0.0006	-25.67	0.00000
ngenes50000:fold6	0.0532	0.0007	76.65	0.00000

Table 8.2: GLM output from final EdgeR sensitivity model. Modelled using pre-filtered data.

8.1.4 Conclusion

The main focus of the study was to determine what effect increasing the filter cut-off had on the performance of differential expression analysis methods. We performed an analysis of 18000 simulated datasets, using four methods of differential expression analysis. We quantified the performance of each method in terms of sensitivity and specificity. Sensitivity and specificity were plotted against the various simulation settings (fold change, number of transcripts and percentage of outliers) and CPM cut-off value. Most settings appeared to have some effect on performance for at least one method. The percentage of outliers in the data did not appear to have any effect on performance for any method. Appropriate interaction effects were plotted, omitting the interaction effects of outliers with other settings.

We then modelled the performance measures in terms of these variables, determing the cut-off appears to have some effect on both both sensitivity and specificity of the statistical test for differential expression. Our model was overdispersed, and therefore further refinement to the model is required.

These preliminary results suggest that CPM filter cutoff value may have some effect on the sensitivity and specificity of a test for differential expression. More specifically, an increase in filter cutoff results in an increase in specificity, and decrease in sensitivity of a test for DE. For both pre and post-filtered data, boxplots show sensitivity decreased by approximately 50% over the range of filter cut-off. With the exception of SAMSeq, specificity increased by approximately 6% over the same range. The output from the GLM suggests main effects alone have a much larger influence on performance, however the model fit was overdispersed.

The size of change in performance measure appears to depend somewhat on the fold change of truly DE transcripts in the data, and the size of the dataset. The percentage of outliers in the dataset does not appear to have any effect on performance measures for any package. For both pre and post-filtered data, increasing dataset size resulted in decreased sensitivity and increased specificity. Increasing fold change resulted in an increase in sensitivity for all packages. Only SAMSeq showed a decrease in specificity due to increase in fold change.

Boxplots suggest small interaction effects between log fold change, size of dataset and CPM cut-off may exist. For example, for all packages except SAMSeq, log fold change

appears to be directly proportional to sensitivity, but sensitivity decreases more sharply as CPM cut-off increases. This effect is more significant at lower CPM thresholds. Re-fitting the GLM is recommended to better quantify these effects.

Chapter 9

Final Conclusion and Discussion

Many packages used to test for differential expression of RNA-sequencing data have been developed over recent years. The statistical methods used are varied, and include Bayesian, non-parametric and parametric based methods. RNA-seq data are often modelled using a negative binomial, or over-dispersed Poisson distribution. Two important steps that are crucial to obtaining good differential expression results are normalization and filtering of the data. Several methods exist to address these steps.

Some software programs used to test for differential expression implement filtering and normalization procedures, while others do not. Before analyzing data, it is important to know some of the technical details regarding how the chosen software handles low count reads, and different library sizes. It is important to have a workflow to guide the process. Included in this document are technical details of several R Bioconductor packages used to test for differential expression. All five methods produce slightly different results, however BaySeq was computationally demanding. SAMSeq reported results for only up-regulated transcripts, a result noted by the package author to be obtained from some datasets [22].

A pipeline was developed to analyze RNA-seq data for differential expression using these packages. Analysis results from a sample dataset was used in this document to demonstrate use of the pipeline. One package, DESeq2, was chosen over the others to further examine results post-analysis in more depth.

In practice, the choice of filter cut-off used on a dataset is rather arbitrary and is not

readily determined from available literature. An experiment was designed to test whether increasing the filter cut-off would affect the specificity or sensitivity of a test for differential expression. The results were fairly consistent for all four packages. An increase in filter cut-off appeared to result in an increase in specificity and decrease in sensitivity of a test for differential expression. The effect of increasing the cut-off appeared to taper off somewhat as the value approached 4 CPM. This suggests that increasing the cut-off value past 4 CPM may not be beneficial to discovery of DE transcripts.

The presence of many transcripts with low to zero read counts is typical in an RNA-seq dataset. Removing these will result in the most gains in performance, however the results of this study suggest the effect of filtering can possibly be further optimized.

Appendix A

Technical review for DESeq2

The following additional information contains theroetical information for the Bioconductor R statistical software package Deseq2 [26]. The following sections describing this package were prepared by UVic Master's degree student Xin Yu for his graduation project under supervisor Dr. Mary Lesperance.

A.1 Introduction to DESeq2 package

DESeq2 [27] is another Bioconductor software package which uses RNA-seq read count data for differential expression analysis and it is a successor to the previous DESeq method [1]. The DESeq2 package fits Negative Binomial generalized linear models (GLMs) to obtain dispersion and log fold-change (LFC) estimates based on the empirical Bayes shrinkage method. Wald or Likelihood ratio tests are used to test whether LFCs are significantly different from zero or not, which is equivalent to the test for DE.

A.1.1 Model

The DESeq2 package applies GLMs on a count matrix Y with a row corresponding to transcripts indexed by g and columns representing samples indexed by i. The element y_{gi} in the count matrix is the number of reads mapped to gth transcript in ith sample. A GLM is

fitted for each transcript.

A Negative Binomial distribution is used to model the read counts Y_{gi} with mean μ_{gi} and dispersion ϕ_g,

$$Y_{gi} \sim NB(\mu_{gi}, \phi_g). \tag{A.1}$$

Let q_{gi} represent the proportion of cDNA fragments from transcript g in sample i and s_{gi} denote the size factor. The mean μ_{gi} is the product of q_{gi} and s_{gi},

$$\mu_{gi} = s_{gi} q_{gi}. \tag{A.2}$$

Usually, a common sample size factor s_i for all transcripts in a sample is used which accounts for differences in sequencing depth [27]. In DESeq [1], Anders and Huber introduced the median-of-ratios method to estimate size factors. The reason that Anders and Huber use the median is that if there exists highly and differentially expressed transcripts, the total read count could be affected strongly, causing the ratio of total read counts to be a poor approximation of the ratio of expected counts. Therefore, the median of the ratios of observed counts are used here. The formula is given as follows:

$$\hat{s_i} = \text{median} \frac{y_{gi}}{(\prod_{\nu=1}^{m} y_{g\nu})^{1/m}}, \quad g = 1, \ldots, G,$$

where the denominator is the geometric mean of all samples and we can regard it as a pseudo-reference sample.

A log-linear model is used to represent q_{gi},

$$log_2(q_{gi}) = \sum_r x_{ir} \beta_{gr} \tag{A.3}$$

with design matrix elements x_{ir} for the i^{th} sample and coefficients β_{gr} [27]. For a set with control or treatment groups, the design matrix elements are covariates that indicate the group condition (control or treatment), and the coefficients represent the log-expression strength of the transcript and the log_2-fold change in expression between control group and treated group. As for the edgeR package, DESeq2 detects for DE by testing whether the log-fold changes differ significantly from zero or not.

A.1.2 Empirical Bayes shrinkage for dispersion estimation

Within group variability is modeled by the dispersion parameter ϕ_g such that

$$Var(Y_{gi}) = \mu_{gi} + \phi_g \mu_{gi}^2. \tag{A.4}$$

It is vital to obtain good estimation of the dispersion parameter for detecting DE. Since RNA-seq data usually has a small number of replicates, it is not appropriate to apply GLM directly to each transcript. To solve this problem, DESeq2 assumes that transcripts of similar average expression strength have similar dispersion to share information between transcripts [27].

DESeq2 assumes the dispersion parameter ϕ_g follows a log-Normal prior distribution which is centered around a trend that depends on the transcript's mean normalized count,

$$log(\phi_g) \sim N(log[\phi_{tr}(\hat{\mu}_g)], \sigma_d^2), \tag{A.5}$$

where ϕ_{tr} is a function of the transcript's mean normalized count, $\hat{\mu}_g = \frac{1}{m}\sum_i (Y_{gi}/s_i)$, and σ_d^2 is the width of the prior [27]. The trend function is as follows:

$$\phi_{tr}(\hat{\mu}) = \frac{a_1}{\hat{\mu}} + \phi_0. \tag{A.6}$$

Gene-wise dispersion estimates DESeq2 first estimates the transcript-wise dispersion based on each individual transcript using maximum likelihood estimation (MLE). The initial GLMs are fitted to obtain initial fitted values of $\hat{\mu}_{gi}^0$. Then transcript-wise estimate ϕ_g^{gw} is obtained by maximizing the Cox-Reid ajusted likelihood of the dispersion given $\hat{\mu}_{gi}^0$,

$$\phi_g^{gw} = \arg\max_{\phi} APL_g(\phi_g) \tag{A.7}$$

$$APL_g(\phi_g) = \ell(\phi_g; y_g, \hat{\mu_g^0}) - \frac{1}{2} logdet(I_g), \tag{A.8}$$

where ℓ is the log-likelihood function and I_g is the Fisher information of μ_g evaluated at $\hat{\mu}_g^0$.

Dispersion trend A smooth curve of the form (A.6) is fitted by regressing the transcript-wise dispersion estimates ϕ_g^{gw} onto the means of the normalized counts, $\hat{\mu}_g$, to allow for

dependence on average expression. The hyperparameters a_1 and ϕ_0 are estimated by fitting a Gamma-family GLM, since the transcript-wise dispersions can be highly skewed [27].

Dispersion prior The transcript-wise dispersion estimates are shrunken toward the values predicted by the curve to get the final dispersion estimates. Since the dispersion parameter ϕ_g follows a log-Normal prior distribution in the form of (A.5), the width of the prior σ_d needs to be estimated. In Love's paper [27], it is demonstrated that the prior variance σ_d^2 can be obtained by calculating as follows:

$$\sigma_d^2 = max\{s_{lr}^2 - \psi_1((m-p)/2), 0.25\}, \tag{A.9}$$

where ψ_1 is the trigamma function and s_{lr}^2 is the variance of the logarithmic residual. A robust estimator for the standard deviation of the logarithmic residuals is used to avoid the influence of dispersion outliers,

$$s_{lr} = mad(log(\phi_g^{gw}) - log[\phi_{tr}(\hat{\mu}_g)]), \tag{A.10}$$

where mad represents the median absolute deviation.

Final dispersion estimates DESeq2 provides the maximum a *posteriori* value (MAP), as the final estimate [27]. Genes with transcript-wise dispersion estimates below the curve are raised to reach the curve, this helps avoid underestimating dispersion. If the transcript-wise dispersion estimates for some transcripts are far above the curve, the shrinkage will reduce the estimates a lot. The final dispersion estimates can be computationally obtained as,

$$\phi_g^{MAP} = \arg\max_{\phi} \left(\ell(\phi; y_g, \hat{\mu}_g^0) - \Lambda_g(\phi)\right) \tag{A.11}$$

where

$$\Lambda_g(\phi) = \frac{-(log(\phi) - log[\phi_{tr}(\hat{\mu}_g)])^2}{2\sigma_d^2}$$

Dispersion outliers If the transcript-wise dispersions are more than 2 residual standard deviations above the curve, $log(\phi_g^{gw}) > log[\phi_{tr}(\hat{\mu}_g)] + 2s_{lr}$, DESeq2 will regard these dispersions as outliers. For dispersion outliers, DESeq2 uses the transcript-wise dispersions not the shrunken estimates to consider those transcripts not following model assumptions or they

show much higher variability for biological or technical reasons [27].

A.1.3 Empirical Bayes shrinkage for fold-change estimation

The coefficients β_{gr} that represent log fold-changes in model (A.3) are assumed to follow a zero-centered Normal prior distribution,

$$\beta_{gr} \sim N(0, \sigma_r^2). \tag{A.12}$$

The problem that transcripts with low counts have large variances for LFCs complicates further analysis [27], DESeq2 solved this problem by shrinking LFC estimates toward zero in a way that if the information for a transcript is low then the shrinkage will be stronger. Then an empirical Bayes approach is applied: the MLEs of LFC are obtained by using the traditional GLM first and then fit a zero-centered Normal distribution to the observed distribution of MLEs over all transcripts which is used as the prior of LFCs in a second round of GLM fits, and last the maximum a posteriori (MAP) estimates are the final estimates of LFC.

Empirical prior estimate First, the standard iteratively reweighted least squares (IRLS) algorithm [30] for each transcript's model (A.1) and (A.3) is used to obtain the MLEs of the coefficients β_{gr}^{MLE}. Then a zero-centered Normal is fit for each column r of the design matrix (except for the intercept) to the empirical distribution of $\boldsymbol{\beta_r^{MLE}}$.

The quantile matching approach is used to make the fit robust agains outliers with high absolute LFC values. The prior width is computed as:

$$\sigma_r = \frac{Q_{|\beta_r|}(1-p)}{Q_N(1-p/2)}, \tag{A.13}$$

where $Q_{|\beta_r|}(1-p)$ is the $(1-p)$ empirical quantile of the absolute value of the observed LFCs, $Q_N(1-p/2)$ is the $(1-p/2)$ theoretical quantile of the prior and p is set to 0.05 by default. Note that extreme LFC values ($|\beta_{gr}^{MLE}| > log_2 10$) are excluded [27].

Final estimates of logarithmic fold changes The final log fold-change estimate can be obtained by maximizing a logarithmic posterior (MAP) for $\boldsymbol{\beta_g}$ for transcript g as follows:

$$\boldsymbol{\beta_g} = \arg_{\boldsymbol{\beta}} \max(\sum_i log\ell(Y_{gi}; \mu_i(\boldsymbol{\beta}), \phi_g) + \Lambda(\boldsymbol{\beta}))$$

where

$$\mu_i(\boldsymbol{\beta}) = s_i e^{\sum_r x_{ir}\beta_{gr}}, \quad \Lambda(\boldsymbol{\beta}) = \sum_r \frac{-\beta_r^2}{2\sigma_r^2},$$

ϕ_g is the final dispersion estimate already obtained.

A.1.4 Normalization and Filtering

In DESeq2, we fit the negative binomial GLM with log link $log(\mu_{gi}) = \sum_r x_{ir}\beta_{gr} + log(s_i)$ which can be transformed as

$$log(\mu_{gi}/s_i) = \sum_r x_{ir}\beta_{gr},$$

where μ_{gi}/s_i can be regarded as the normalized counts.

In DESeq2, the *results* function conducts the filtering procedure using the mean of normalized counts as a filter statistic by default. The threshold value of the filter statistic is obtained optimizing the number of adjusted p values lower than a significance level alpha which is set to 0.1. The details of how the filtering works can be found in the package document of DESeq2.

A.1.5 Hypothesis testing for DE

DESeq2 uses the Wald test to compare the final estimate of LFC divided by its standard error which leads to a $z - statistic$ with a standard Normal. Since the tests are multiple testing, the $p - values$ from Wald tests are adjusted by the procedure of Benjamini and Hochberg [4]. The likelihood ratio test that fits a reduced model to test if the LFC is significantly different from zero is also provided in the DESeq2 package.

More mathematical details for models, parameter estimation and regression algorithms are demonstrated in the Methods section of the DESeq2 document [27].

Appendix B

Technical review for EdgeR and Robust EdgeR

The following additional information contains theroetical information for the Bioconductor R statistical software packages, EdgeR[39], and it's relative Robust EdgeR[39]. The following sections describing this packages were prepared by UVic Master's degree student Xin Yu for his graduation project under supervisor Dr. Mary Lesperance.

B.1 Introduction to edgeR package

edgeR is a Bioconductor software package that is used for differential expression analysis for RNA read count data. Negative Binomial models (overdispersed Poisson models) are used in the edgeR package to incorporate variabilities including both biological and technical sources [25]. The quadratic mean-variance relationship of the observed data can be accommodated with the Negative Binomial models using generalized linear models (GLM).

Obtaining good dispersion estimates is very important in GLM fitting, however, the problem of the small number of replicates in RNA-seq datasets makes it difficult to get accurate estimates of the dispersion parameters by directly applying univariate estimators of dispersion. Therefore, the edgeR package uses Empirical Bayes method to estimate transcript-specific dispersion parameters, which makes it possible to get good estimates of dispersion

when the number of replicates is not large enough. The use of Empirical Bayes methods shares information between transcripts and hence transcript-specific variation estimates can be obtained [25] [33]. Weighted likelihood methods are used in edgeR to approximate the Empirical Bayes methods, as it is shown that an Empirical Bayes estimator can be equivalently obtained by maximizing a weighted likelihood function of observations [42][2]. The quasi-likelihood method is used to get the prior weight for the empirical Bayes prior.

Lastly, after obtaining transcript-specific dispersion parameters, edgeR fits GLMs and then obtains the log-fold change and other estimates to detect differentially expressed transcripts by constructing likelihood ratio tests (LRT). The edgeR package also displays some useful plots such as the multiple dimensional scaling (MDS) plot and the biological coefficient of variation (BCV) plot that are helpful for data exploration and displaying results.

B.1.1 Generalized linear model

The edgeR package uses generalized linear models (GLMs) to deal with RNA-seq count data, as GLMs allow for non-normally distributed data and the distribution of the data can be modelled using its mean-variance relationship.

RNA-seq data can be summarized as the number of reads that are mapped to a certain transcript of interest. Read counts can be used for differential analysis, that is to detect transcripts which are differentially expressed under different experimental groups or conditions. Each group may have several samples, and we regard samples in the same group as replicates. The table of read counts is a $G \times n$ matrix, where G is the total number transcripts and n is the total number of samples over all groups. Obviously, the count matrix is of non-negative integers.

Let a *libray* denote the reads counts in a sample and the total number of reads in the library is represented as *library size*. For a particular transcript g, let y_{gi} represent the read count in ith sample. Given the biological groups and sequencing depth (which represents the number of nucleotides contributing to a portion of an assembly), we denote N_i as the library size and λ_{gi} as the expected proportion of reads in sample i mapped to transcript g, then

$$E(y_{gi}) = \mu_{gi} = \lambda_{gi} \cdot N_i \tag{B.1}$$

For example, if we have two experimental groups consisting of two replicates in each group, let i = 1, 2 refer to group 1 and i = 3, 4 refer to group 2, we have $\lambda_{g1} = \lambda_{g2} = \lambda_g^1$ and $\lambda_{g3} = \lambda_{g4} = \lambda_g^2$ are the expected proportions of reads mapped to transcript g in samples from group 1 and group 2 respectively. For each transcript $g = 1, 2, \ldots, G$, the task of detecting differentially expressed transcripts is to test

$$H_0 : \lambda_g^1 = \lambda_g^2 \quad versus \quad H_1 : \lambda_g^1 \neq \lambda_g^2 \tag{B.2}$$

B.1.2 Technical variation and biological variation

RNA-seq data have two sources of variation, one is technical variation and the other is biological variation. Technical variation is caused by imperfect measurement of transcript expression, as the experimental technology cannot ensure that every test has 100 percent accuracy. Biological variation results from different levels of transcript expression between different biological samples. The variation in expression levels from different biological samples will always exist even if the experiments are conducted under the same group. We need to detect these two levels of variation, which helps us understand the differential expression analysis better.

Let the true unobserved expression level of transcript g from sample i be denoted by π_{gi} which is the true concentration of fragments in the ith sample from transcript g. Given the library size N_i and π_{gi}, the expected count from transcript g in sample i is $E(y_{gi}|\pi_{gi}) = \pi_{gi}N_i$. If the counts are from the same RNA sample for any transcript, they should follow a Poisson distribution and the variance of counts is $var(y_{gi}|\pi_{gi}) = \pi_{gi}N_i$ [29]. This is the variation from the source of sequencing technology (technical variation).

However, counts for biological replicates usually have larger variance than their means. Hence, for replicated RNA-seq count data, the Poisson distribution predicts smaller variation than what it should be. We say there is a problem of overdispersion, therefore, we need to further build a variance-mean relationship to model the variance. The variance is defined so that the coefficient of variation (CV) should remain constant for any given transcript, which gives that $E(\pi_{gi}) = \lambda_{gi}$ and $var(\pi_{gi}) = \phi_g\lambda_{gi}^2$, where λ_{gi} is the population expected proportion of reads mapped to transcript g in the ith sample given the biological groups, and ϕ_g is the square coefficient of variation [6]. According to the law of total variance, the

variance of read counts y_{gi} can be derived as

$$var(y_{gi}) = E_\pi[var(y_{gi}|\pi_{gi})] + var_\pi[E(y_{gi}|\pi_{gi})] = \mu_{gi} + \phi_g \mu_{gi}^2 \tag{B.3}$$

where μ_{gi} is defined in (B.1). Dividing by μ_{gi}^2, the two sides become

$$CV^2(y_{gi}) = 1/\mu_{gi} + \phi_g. \tag{B.4}$$

The first term on the right side is the squared CV of y_{gi} given π_{gi} and the second term is the squared CV of π_{gi}. The squared CV of y_{gi} given π_{gi} is the squared technical CV and the squared CV of π_{gi} is the squared biological CV. Hence we can represent the equation above as

$$TotalCV^2 = TechnicalCV^2 + BiologicalCV^2. \tag{B.5}$$

This decomposition of total CV was first introduced by [25], and $\phi_g^{1/2}$ is called the biological coefficient of variation (BCV). BCV represents the coefficient of variation that demonstrates how the expression level of transcripts would vary between different biological samples. When expression level increases, the technical CV typically decreases while the biological CV (BCV) does not and BCV becomes the dominant source of variation.

B.1.3 Negative binomial GLM

As mentioned in Section B.1.2, the Poisson distribution does not model overdispersion for replicated RNA-seq count data. Therefore, we need to find another distribution family to model replicated read counts. Given the variance-mean relationship derived in (B.3), a Negative Binomial distribution family is used to model read counts. Here, a Negative Binomial GLM,

$$y_{gi} \sim NB(\mu_{gi}, \phi_g), \tag{B.6}$$

is used in the edgeR package to conduct differential expression analysis. The probability mass function of a Negative Binomial distribution is given as

$$f(k_{gi}; r_{gi}, p_{gi}) = Pr(X = k_{gi}) = \binom{k_{gi} + r_{gi} - 1}{k_{gi}} p_{gi}^{k_{gi}} (1 - p_{gi})^{r_{gi}},$$

$$\mu_{gi} = \frac{p_{gi} r_{gi}}{1 - p_{gi}}; \quad \sigma^2_{gi} = \mu_{gi} + \phi_g \mu^2_{gi} = \frac{p_{gi} r_{gi}}{(1 - p_{gi})^2},$$

where μ_{gi} is the mean read count from transcript g in ith sample, σ^2_{gi} is the variance of y_{gi} and ϕ_g is the negative binomial dispersion parameter. It is supposed that we can use a log-linear model to represent μ_{gi},

$$log(\mu_{gi}) = x_i^T \beta_g + log N_i, \tag{B.7}$$

where $\mu_{gi} = \lambda_{gi} N_i$, β_g is the regression coefficient for transcript g and x_i is the covariate vector referring to the experimental groups matched to sample i.

The vector of λ_{gi} can be represented by the product of a design matrix X and β_g, where the design matrix X is a matrix that contains the covariate vectors x_i. For example, if we have four transcripts and two replicates for two groups. The design matrix can be given as

$$X = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}, \tag{B.8}$$

where the first two rows indicate the experimental groups for samples in group 1 and the second two rows indicate those of group 2. The coefficient vector β_g is

$$\beta_g = \begin{pmatrix} \beta_{g1} \\ \beta_{g2} \end{pmatrix}, \tag{B.9}$$

where $\beta_{g1} = log\lambda^1_g$ denotes the log-expression in the first group and $\beta_{g2} = log(\lambda_{g2}/\lambda_{g1})$ represents the log-fold change of the expression in the second group compared to the first group. Therefore, the original test for differential expression

$$H_0 : \lambda^1_g = \lambda^2_g \quad versus \quad H_0 : \lambda^1_g \neq \lambda^2_g \tag{B.10}$$

is equivalent to the test

$$H_0 : \beta_{g2} = 0 \quad versus \quad H_1 : \beta_{g2} \neq 0. \tag{B.11}$$

A modified Fisher-scoring algorithm is used in the edgeR package to estimate the parameter vector β_g. Here, A Levenberg damping modification is added to the usual Fisher-scoring algorithm, which is used to ensure the sequence of iterations can converge for all transcripts and all datasets [25]. edgeR provides tests like the likelihood ratio test using a chi-square approximation or some other approximations such as an F-test approximations to test whether the coefficients are zero or not. edgeR also provides an exact test for DE.

B.1.4 Dispersion estimation based on Empirical Bayes

As mentioned in the previous section, RNA-seq data usually has small samples, therefore, methods such as Maximum Likelihood estimation (MLE) do not perform well for RNA-seq data analysis and we may not obtain good estimations using those approaches. Dispersion parameter estimation is very important for fitting GLMs and detecting DE. In order to get accurate estimation of the dispersion parameter in the negative binomial model, edgeR uses the Empirical Bayes approach introduced by McCarthy [25] to estimate the dispersion parameter and the method is based on the idea of an adjusted profile likelihood proposed by Cox and Reid [4]

B.1.5 Cox-Reid adjusted profile likelihood

The Cox-Reid Adjusted Profile Likelihood (CR) method is based on the idea of approximate conditional likelihood which reduces to residual maximum likelihood (REML). REML removes the nuisance parameter effect, and this gives unbiased estimation of the dispersion parameter [8].

The dispersion parameter ϕ_g is the parameter of interest in this section, and other parameters such as the regression coefficients β_g and the means μ_{gi} are nuisance parameters. CR methods assume that the parameters of interest are orthogonal to nuisance parameters, which results in the Fisher information matrix being block diagonal [4]. The Cox-Reid adjusted profile likelihood (APL) for ϕ_g is an adjusted log-likelihood given as

$$APL_g(\phi_g) = \ell(\phi_g; y_g, \hat{\beta}_g) - \frac{1}{2} logdet(I), \tag{B.12}$$

where the first term on the right side is the log-likelihood and the second term is the adjusted term, y_g is the count vector for transcript g, $\hat{\beta}_g$ is the regression coefficient parameter and I is the Fisher information of β_g evaluated at $\hat{\beta}_g$. The regression coefficient parameter $\hat{\beta}_g$ is estimated by MLE method given ϕ_g, hence $\hat{\beta}_g$ is a function of ϕ_g. Therefore, both terms on the right side are functions of ϕ_g, which makes $APL_g(\phi_g)$ contain only one parameter ϕ_g and we can get the estimate for ϕ_g by maximizing $APL_g(\phi_g)$.

B.1.6 Weighted likelihood Empirical Bayes

The empirical Bayes approach is used in the edgeR package to estimate dispersion parameters. Compared to classical maximum likelihood estimation, Empirical Bayes estimation has been shown to perform much better for high dimensional problems [5][6][40]. The idea of Empirical Bayes estimation is that we first estimate the prior distribution from the data and then we get the posterior estimates based on the standard Bayesian approach. As mentioned before, the number of replicates is usually very small in RNA-seq data and in order to get reliable dispersion estimates, an empirical Bayes approach with information shared across transcripts is applied in edgeR. However, no conjugate prior is available for the negative binomial dispersion making it impossible to apply the empirical Bayes method directly to RNA-seq count data.

A weighted likelihood is used to approximate the empirical Bayes method as it can be shown that maximizing a weighted likelihood function on a dataset is equivalent to an empirical Bayes estimator [2][42]. Therefore, the empirical Bayes method is approximated by applying a weighted likelihood method.

Common Dispersion When we assume that all transcripts share a same dispersion parameter, this shared dispersion parameter is called the common dispersion, which is the simplest way to share the information between transcripts. We can obtain the common dispersion by maximizing the common APL, which is given as

$$APL_c = \frac{1}{G}\sum_{g=1}^{G} APL_g(\phi) \tag{B.13}$$

where G is the total number of transcripts. We can regard the common APL as a weighted likelihood where each transcript has equal weight.

Trended Dispersion The reality is that the transcript-specific dispersion parameter may vary from each other, therefore, the common dispersion parameter is too naive to be the estimated dispersion parameter. Analysis of RNA-seq datasets have found that transcripts that have higher levels of expression have smaller dispersions and transcripts that have lower level of expression have larger dispersions [6]. In edgeR, another version of dispersion is introduced which is called the trended dispersion. Trended dispersions are obtained by modelling a mean-dispersion on the transcript-wise expression level [**?**].

We can estimate the trended dispersion using the weighted likelihood method. In edgeR, a log two average count-per-million (log_2CPM) is calculated for each transcript first using the aveLogCPM function, where CPM is computed as the raw count divided by the library size and multiplied by one million. The log two average CPM is calculated as log2(AveCPM), where AveCPM is the weighted average of CPM such that larger library sizes are given more weight. After that, transcripts are sorted by their average logCPM. In order to obtain the trended dispersion, we first need to introduce a locally shared APL denoted as $APL_{s_g}(\phi_g)$. Suppose there are a set of transcripts, represented as C_g, which are closest to transcript g in average logCPM. The set C_g should contain at least 25% of all transcripts and we need to increase the proportion if the total number of transcripts is small. This ensures that the locally shared APL is generated by a large enough number of transcripts to obtain good estimation of trended dispersions [6].

In order to consider the relevance of expression levels between transcript g and transcripts in the set C_g, a graduated weighting approach is used. The graduated weighting approach gives a weight for the APL of transcript a in C_g, denoted as w_a, using a tricube function

$$w_a = (1 - |x_a|^3)^3 \tag{B.14}$$

where $-1 < x_a < 1$ denotes the scaled difference in average logCPMs between transcript a and transcript g. This means the closer the expression level of transcript a is to that of transcript g, the smaller $|x_a|$ will be, and the larger w_a will be [6]. Last, the locally shared

APL for transcript g is defined as

$$APL_{s_g}(\phi_g) = \frac{\sum_{a \in C_g} w_a \cdot APL_a(\phi_g)}{\sum_{a \in c_g} w_a} \tag{B.15}$$

We can obtain the trended dispersion estimate for transcript g by maximizing $APL_{s_g}(\phi_g)$.

Tagwise Dispersion If transcripts with equal expression levels have the same dispersion and true dispersion follows the mean-dispersion trend, the trended dispersion would be sufficient to estimate the dispersion [6]. But the truth is that we need to estimate individual transcript-specific dispersion for the real RNA-seq data. Due to lack of replicates in RNA-seq data, it is difficult to obtain good estimation of dispersion based on a single transcript. Here, the Empirical Bayes method that combines individual and shared information is used to get reliable transcript-specific dispersion estimates. As mentioned in the previous section, a weighted likelihood approach is used to approximate the empirical Bayes method as there is no conjugate prior distribution for the negative binomial dispersion. The weighted APL for transcript g is defined as

$$APL_{w_g}(\phi_g) = APL_g(\phi_g) + G_0 \cdot APL_{s_g}(\phi_g) \tag{B.16}$$

where $APL_g(\phi_g)$ is the APL for transcript g, $APL_{s_g}(\phi_g)$ is the locally shared APL for transcript g and G_0 is the weight of $APL_{s_g}(\phi_g)$. Then we can obtain the tagwise dispersion estimate by maximizing the weighted APL $APL_{w_g}(\phi_g)$.

We can also get an idea from (B.16) for why we can approximate the empirical Bayes strategy using the weighted likelihood approach. The first term on the right side can be regarded as the observation term and the second can be referred to as the prior distribution of ϕ_g where G_0 is the prior weight, which makes $APL_{w_g}(\phi_g)$ contain prior and observed information and hence $APL_{w_g}(\phi_g)$ can be regarded as the posterior distribution of ϕ_g. This is exactly the idea of empirical Bayes approach. Now, in order to get the tagwise dispersion estimate and shrink the individual dispersions to the trended dispersions, we need to choose the prior weight G_0 . Basically, when the dispersions are constant or they strictly follow the mean-dispersion trend, large prior weight needs to be assigned to the locally shared APL. If dispersions vary among different transcripts, we need to set the prior weight to a small value. If the prior weight is zero, it means that no shared information from other transcripts

is included to get the dispersion and the transcript-wise dispersion is the final dispersion. If the prior weight is set to positive infinity, this represents the trended dispersion is the final dispersion and information from individual transcripts will be ignored. In the next section, an approach to estimate the prior weight is introduced.

B.1.7 Estimating prior weight

The prior weight is defined as

$$G_0 = \frac{d_0}{d_g}, \tag{B.17}$$

where d_0 is the *prior degrees of freedom* and d_g is the *residual degrees of freedom* for transcript g. The prior degrees of freedom d_0 is estimated using a quasi-likelihood approach. The quasi-likelihood variance function can be represented in this way

$$var(y_{gi}) = \sigma_g^2 \cdot V(\mu_{gi}) \tag{B.18}$$

where σ_g^2 is the $quasi - dispersion\ parameter$. According to [15], the prior distribution for σ_g is assumed to be a scaled inverse χ^2 -distribution with degrees of freedom d_0 and scaling factor $s_0^2 d_0$,

$$\sigma_g^2 \quad \sim \quad s_0^2 \cdot \frac{d_0}{\chi^2_{d_0}}. \tag{B.19}$$

Let D_g denote the residual deviance of the GLM fitted on read counts for transcript g. Then the mean residual deviance

$$s_g^2 = \frac{1}{d_g} D_g \tag{B.20}$$

is an estimator of σ_g^2. In [2], it has been shown that applying the saddlepoint approximation [8] that the mean deviance s_g^2 approximately follows a χ^2 -distribution with degrees of freedom d_g and scaling factor s_g^2/d_g. The marginal distribution of s_g^2 is a scaled F-distribution,

$$s_g^2 \quad \sim \quad s_0^2 \cdot F_{d_g, d_0}, \tag{B.21}$$

where F_{d_g,d_0} represents the F-distribution with degrees of freedom d_g and d_0 [15][40]. We can estimate s_g^2 and d_0 using the method of moments.

After we obtain prior degrees of freedom d_0, the prior weight G_0 is found and then we

can get the tagwise dispersion estimates using (B.16). The transcript-specific dispersion is used for fitting transcript-wise negative binomial GLMs, through which we can get the the estimates of coefficient vector β_g. Then the likelihood ratio test is conducted for detecting differentially expressed transcripts.

B.2 Robust edgeR

Since outliers may have impact on the estimation of parameters, therefore, it is vital to check how the outliers in RNA-seq data could affect the differential analysis and build a robust approach to control the influence of outliers. It has been shown that edgeR can be sensitive to outliers when there is sufficient dispersion smoothing towards the trend, which would lead to underestimation of dispersion in the process of shrinking [2]. Zhou, Lindsay and Robinson introduced an approach of *observation weights* that downweights outliers to weaken their influence and implemented this method in edgeR [47], which can be regarded as a robust version of edgeR.

B.2.1 Regular negative binomial GLM

As in the previous section, the log mean vector μ_g can be represented as the sum of the product of the design matrix X and regression coefficient vector β_g, and log library size vector $logN$.

$$log(\mu_g) = X\beta_g + logN. \tag{B.22}$$

Maximum likelihood estimation is used to estimate the regression coefficients. The derivative of the log-likelihood function with respective to β_g is $X^T z_g$, where $z_{gi} = (y_{gi} - \mu_{gi})/(1 + \phi_g \mu_{gi})$. The iteratively re-weighed least squares (IRLS) algorithm derived by Agostinelli and Alqallaf [**?**] is used to obtain the estimates of β_g in the following form:

$$\beta_g^{new} = \beta_g^{old} + (X^T \Omega_g X)^{-1} X^T z_g, \tag{B.23}$$

where Ω_g is the diagonal matrix of working weights, which are $\mu_g/(1+\phi_g\mu_g)$ for the negative binomial model, and $(X^T\Omega_g X)^{-1}$ is the Fisher information matrix [47]. The form of the β_g

estimates in (B.23) are the same as those obtained using the Fisher scoring method.

B.2.2 A robust negative binomial GLM

The way that the robust negative binomial GLM works is that a weight is assigned to each observation. For observations that deviate far from the fit, lower weights are attached to those observations. Pearson residuals from the current fit are used to generate the weight function and the weight is used in the next iteration of estimation. The Cox-Reid adjusted profile likelihood (APL) introduced in regular edgeR is assigned the same observation weight, therefore, the influence of extreme observations is weakened for both regression coefficients and dispersions estimates. The Pearson residual of an observation y_{gi} is calculated as:

$$r_{gi} = \frac{y_{gi} - \hat{\mu}_{gi}}{\sqrt{\hat{\mu}_{gi}(1 + \hat{\phi}_g \hat{\mu}_{gi})}} \tag{B.24}$$

where $\hat{\mu}_{gi}$ is the fitted value and $\hat{\phi}_g$ is the moderated dispersion estimate. The weight is geneated using the Pearson residuals as follows:

$$\omega_{gi} = \omega(r_{gi}) = \begin{cases} \dfrac{k}{abs(r_{gi})}, & \text{for abs}(r_{gi}) > \text{k} \quad \text{(B.25)} \\ 1, & \text{for abs}(r_{gi}) \leq \text{k} \quad \text{(B.25}') \end{cases}$$

where k is a tuning constant which is usually set to 1.345 [15]. The weight ω_{gi} is assigned to its observation and it is used in the next iteration of estimation, to obtain a new IRLS equation:

$$\beta_g^{W-new} = \beta_g^{W-old} + (X^T[W_g \Omega_g]X)^{-1} X^T[W_g] z_g, \tag{B.26}$$

where W_g is a diagonal matrix of weights of observation for transcript g and $X^T[W_g \Omega_g]X$ is the new Fisher information matrix with the observation weights. Attaching the weights to the observations, we get the new APL with respective dispersion ϕ_g:

$$APL_g^W(\phi_g) = \ell^W(\phi_g; y_g, \hat{\beta}_g) - \frac{1}{2} logdet(I_g^W), \tag{B.27}$$

where $\ell^W(\cdot) \equiv \sum_i \omega_{gi} l(\cdot)$ is the weighted log-likelihood function and $I_g^W = X^T[W_g \Omega_g]X$ is the weighted Fisher information matrix. Applying the new APL to the regular edgeR procedure we can obtain robust dispersion estimates, then regression coefficients estimates are obtained. Lastly, we can test DE by conducting hypothesis testing based on regression coefficient parameters. Since the main difference between robust edgeR and regular edgeR is that weights are attached to observation in robust edgeR, there is only a small change that needs to be added to the regular edgeR pipeline for building robust edgeR pipeline.

Appendix C

Additional Information

C.1 NB-Poisson Relationship

The section to show the relationship between the Poisson distribution and the Negative Binomial distribution. RNA-seq data is often modelled using the overdispersed Poisson, or Negative Binomial, distibution to account for variation in transcript expression that naturally occurs in replicates that come from different biological sources.

C.1.1 Negative Binomial Distribution

The negative binomial distribution is a discrete distribution which gives the probability of r successes over x independent and identically distributed Bernouille trials, in which there is sucess on the X'th trial. If we denote the probability of success as p and the probability of failure as $1-p$ then the probability density function can be given by

$$\mathbb{P}(X = x|r, p) = \binom{x-1}{r-1} p^r (1-p)^{x-r}, x = r, r+1, r+2, \cdots \tag{C.1}$$

There are several other variations of the negative binomial distribution.

The moment generating function is $\sum_{x=0}^{\infty} e^{tx}\binom{x+r-1}{r-1}p^r(1-p)^x$. This leads to mean, variance, skewness and kurtosis

$$\mu = \frac{r(1-p)}{p}$$
$$\sigma^2 = \frac{r(1-p)}{p^2}$$
$$\gamma_1 = \frac{2-p}{\sqrt{r(1-p)}}$$
$$\gamma_2 = \frac{p^2-6p+6}{r(1-p)}$$

C.1.2 Poisson Distribution

The Poisson Distribution models the probability that exactly Y events occur in an experiment or over a set period of time. It is a limiting form of the binomial distribution, in which the number of trials, N, extends to infinity.

The probability function of the distribution is

$$\mathbb{P}(Y = y|\lambda) = \frac{\lambda^y e^{-\lambda}}{y!}, y = 0, 1, 2, ... \tag{C.2}$$

Where λ is a rate parameter, $\lambda = Np$ from the binomial pdf. The moment-generating function of the Poisson distribution is given by $\sum_{y=0}^{\infty} e^{ty}\frac{\lambda^y e^{-\lambda}}{y!}$, leading to mean and variance

$$\mu = \lambda$$
$$\sigma^2 = \lambda$$
$$\gamma_1 = \lambda^{\frac{-1}{2}}$$
$$\gamma_2 = \frac{\lambda(1+3\lambda)}{\lambda^2} - 3$$

C.1.3 NB-Poisson Relationship

Consider a random variable, Y, taken from a poisson distribution with rate parameter λ. Then Y has pdf

$$\mathbb{P}(Y = y|\lambda) = \frac{\lambda^y e^{-\lambda}}{y!}, y = 0, 1, 2, ... \tag{C.3}$$

Now suppose λ is not constant, but is itself a random variable drawn from a gamma distribution with parameters β and α. Then λ has pdf

$$\mathbb{P}(\lambda|\beta, \alpha) = \frac{\lambda^{\alpha-1} e^{\frac{-\lambda}{\beta}}}{\Gamma(\alpha)\beta^\alpha}, \lambda > 0 \tag{C.4}$$

Then using the total law of probability to find the proability of Y we have

$$\mathbb{P}(Y = y) = \int_0^\infty \mathbb{P}(Y = y|\lambda) f(\lambda)\, \mathrm{d}x \tag{C.5}$$

$$= \int_0^\infty \frac{\lambda^y e^{-\lambda}}{y!} \cdot \frac{\lambda^{\alpha-1} e^{\frac{-\lambda}{\beta}}}{\Gamma(\alpha)\beta^\alpha}\, \mathrm{d}\lambda \tag{C.6}$$

$$= \frac{1}{\Gamma(\alpha)\beta^\alpha} \int_0^\infty \lambda^{y+\alpha-1} e^{-\lambda \frac{\beta}{\beta+1}} \mathrm{d}\lambda \tag{C.7}$$

Inside the integral is the numerator for another Gamma distribution with paramters $y + \alpha - 1$ and $\frac{\beta}{\beta+1}$. Multiplying the whole result by $\Gamma(y + \alpha)\frac{\beta}{\beta+1}^{y+\alpha}$ gives

$$\mathbb{P}(Y=y) = \frac{\Gamma(y+\alpha)\frac{\beta}{\beta+1}^{y+\alpha}}{\Gamma(\alpha)\beta^{\alpha}} \int_0^{\infty} \frac{\lambda^{y+\alpha-1}e^{-\lambda\frac{\beta}{\beta+1}}}{\Gamma(y+\alpha)\frac{\beta}{\beta+1}^{y+\alpha}}\mathrm{d}x \tag{C.8}$$

$$= \frac{\Gamma(y+\alpha)\frac{\beta}{\beta+1}^{y+\alpha}}{\Gamma(\alpha)\beta^{\alpha}} \cdots \tag{C.9}$$

$$= \frac{(y+\alpha-1)!}{y!\cdot(\alpha-1)!} \cdot \frac{\beta^y}{(\beta+1)^y} \cdot \frac{1}{(\beta+1)^{\alpha}} \tag{C.10}$$

$$= \binom{y+\alpha-1}{\alpha-1} \cdot \left(\frac{\beta}{\beta+1}\right)^y \cdot \left(\frac{1}{\beta+1}\right)^{\alpha} \tag{C.11}$$

Which is negative binomial with $p = \frac{\beta}{(\beta+1)}$ and $r = \alpha$

C.1.4 Mean and Variance

The mean of the negative binomial distribution can be derived as follows.

$$\begin{aligned}
\mathrm{E}[X] &= \sum_{x=i} f(x_i)\cdot x_i \\
&= \sum_{x=0}^{\infty} \binom{x+r-1}{r-1} p^x(1-p)^r \cdot x \\
&= \binom{0+r-1}{r-1} p^0(1-p)^r + \sum_{1}^{\infty} \binom{x+r-1}{r-1} p^x(1-p)^r \cdot x \\
&= 0 + \sum_{x=1}^{\infty} \frac{(x+r-1)!}{(r-1)!x!} p^x(1-p)^r \cdot x \\
&= \frac{rp}{(1-p)} \sum_{x=1}^{\infty} \frac{(x+r-1)!}{(r)!(x-1)!} p^{x-1}(1-p)^{r+1}
\end{aligned}$$

Let $s = r+1$ and $w = x-1$ inside the summation.

$$\begin{aligned}\mathrm{E}[X] &= \frac{rp}{(1-p)} \sum_{w=0}^{\infty} \frac{(w+s-1)!}{(s-1)!w!} p^w (1-p)^s \\ &= \frac{rp}{(1-p)} \sum_{w=0}^{\infty} \binom{w+s-1}{s-1} p^w (1-p)^s\end{aligned}$$

The summation is the sum over a pmf $NB(s,p)$, which evaluates to 1. Then

$$\mathrm{E}[X] = \frac{rp}{(1-p)} \tag{C.12}$$

The variance can be derived similarly using the following formula:

$$\mathrm{Var}[X] = \mathrm{E}[X^2] - (\mathrm{E}[X])^2 \tag{C.13}$$

$$\begin{aligned}\mathrm{E}[X^2] &= \sum_i f(x_i) \cdot x_i^2 \\ &= \sum_{x=0}^{\infty} \binom{x+r-1}{r-1} p^x (1-p)^r \cdot x^2 \\ &= 0 + \sum_{x=1}^{\infty} \binom{x+r-1}{r-1} p^x (1-p)^r \cdot x^2 \\ &= \sum_{x=1}^{\infty} \frac{(x+r-1)!}{(r-1)!x!} p^x (1-p)^r \cdot x^2 \\ &= \frac{rp}{(1-p)} \sum_{x=1}^{\infty} \frac{(x+r-1)!}{(r)!(x-1)!} p^{x-1} (1-p)^{r+1} \cdot x\end{aligned}$$

Let $s = r+1$ and $w = x-1$ inside the summation.

$$\begin{aligned}
\mathrm{E}[X^2] &= \frac{rp}{(1-p)} \sum_{w=0}^{\infty} \frac{(w+s-1)!}{(s-1)!w!} p^w (1-p)^s (w+1) \\
&= \frac{rp}{(1-p)} \sum_{w=0}^{\infty} \binom{w+s-1}{s-1} p^w (1-p)^s (w+1) \\
&= \frac{rp}{(1-p)} \left[\sum_{w=0}^{\infty} \binom{w+s-1}{s-1} p^w (1-p)^s \cdot w + \sum_{w=0}^{\infty} \binom{w+s-1}{s-1} p^w (1-p)^s \right]
\end{aligned}$$

The first summation inside the large brackets is the expected value of a negative binomial random variable $\mathrm{E}[X], X \sim NB(s,p)$ The second summation is the sum over a pmf $NB(s,p)$, which evaluates to 1.

$$\begin{aligned}
\mathrm{E}[X^2] &= \frac{rp}{(1-p)} \left[\frac{sp}{(1-p)} + 1 \right] \\
&= \frac{rp(1+rp)}{(1-p)^2}
\end{aligned}$$

Inserting into formula C.13 and substituting in C.12 gives:

$$\begin{aligned}
\mathrm{Var}[X] &= \frac{rp(1+rp)}{(1-p)^2} - \left(\frac{rp}{1-p} \right)^2 \\
&= \frac{rp}{(1-p)^2}
\end{aligned}$$

Appendix D

GLM output

The following additional information contains the R output used to assess model fit for the experimental data (see Chapter 8). The general linear model statistical package used to model the data, `glm`, is available through Bioconductor R's `base` statistical software package.

D.1 Pre-filtered Data

D.1.1 Sensitivity

	Df	Deviance	AIC
<none>		219635.70	626062.77
pout	1	219635.71	626060.77
cutoff:fold	18	964290.56	1370681.63
ngenes:cutoff	18	263053.26	669444.32
ngenes:fold	4	232372.16	638791.22

Table D.1: Summary of the changes in fit that would result from dropping terms from the Robust EdgeR sensitivity model. Only removable terms are shown. Modelled using pre-filtered data.

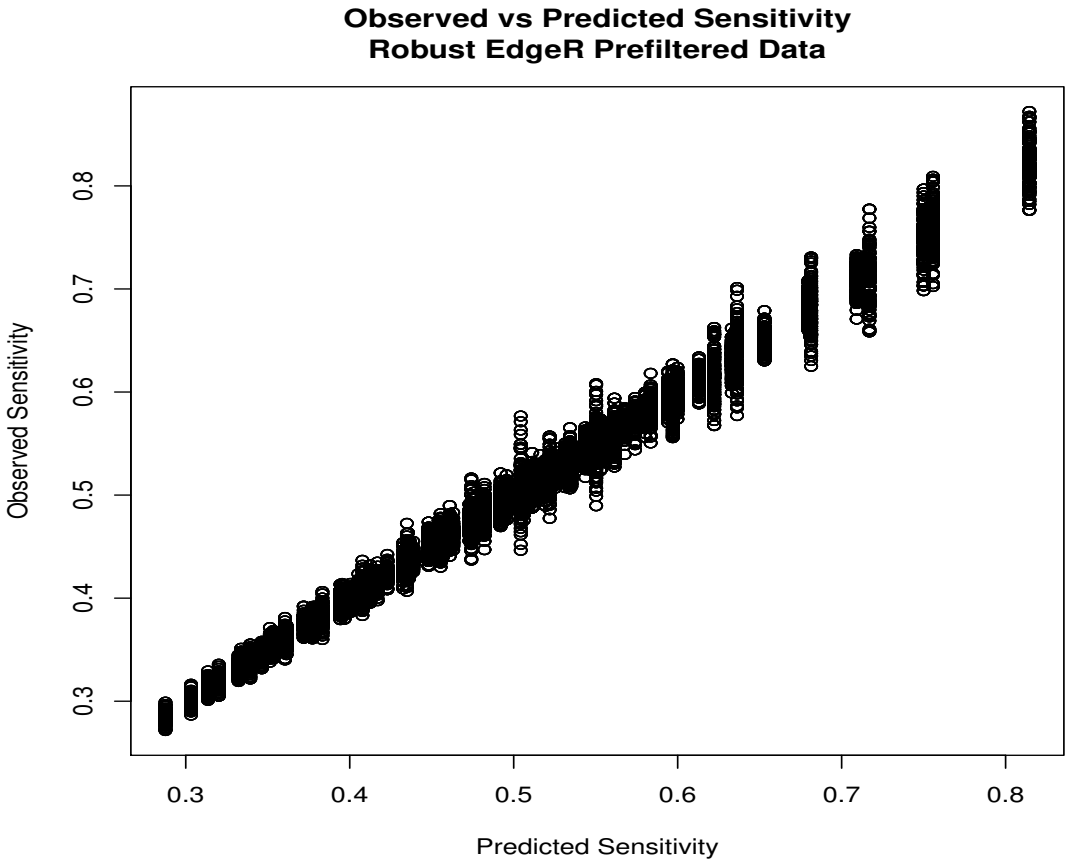

Figure D.1: Observed sensitivity vs predicted sensitivity values for the Robust EdgeR prefiltered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	0.2019	0.0007	277.06	0.00000
ngenes150000	-0.1847	0.0008	-225.16	0.00000
ngenes50000	0.2973	0.0010	290.57	0.00000
cutoff0.44	-0.1141	0.0010	-114.90	0.00000
cutoff0.89	-0.2733	0.0010	-276.38	0.00000
cutoff1.33	-0.3807	0.0010	-385.02	0.00000
cutoff1.78	-0.4730	0.0010	-477.47	0.00000
cutoff2.22	-0.5556	0.0010	-559.30	0.00000
cutoff2.67	-0.6311	0.0010	-633.20	0.00000
cutoff3.11	-0.7031	0.0010	-702.54	0.00000
cutoff3.56	-0.7708	0.0010	-766.84	0.00000
cutoff4	-0.8371	0.0010	-828.77	0.00000
fold3	0.5584	0.0009	644.26	0.00000
fold6	0.9275	0.0009	1011.45	0.00000
cutoff0.44:fold3	-0.1011	0.0011	-90.58	0.00000
cutoff0.89:fold3	-0.2146	0.0011	-193.47	0.00000
cutoff1.33:fold3	-0.2771	0.0011	-250.08	0.00000
cutoff1.78:fold3	-0.3207	0.0011	-289.02	0.00000
cutoff2.22:fold3	-0.3526	0.0011	-316.88	0.00000
cutoff2.67:fold3	-0.3754	0.0011	-336.20	0.00000
cutoff3.11:fold3	-0.3928	0.0011	-350.21	0.00000
cutoff3.56:fold3	-0.4082	0.0011	-362.31	0.00000
cutoff4:fold3	-0.4183	0.0011	-369.38	0.00000
cutoff0.44:fold6	-0.2578	0.0012	-220.85	0.00000
cutoff0.89:fold6	-0.4748	0.0012	-411.79	0.00000
cutoff1.33:fold6	-0.5631	0.0012	-489.47	0.00000
cutoff1.78:fold6	-0.6204	0.0012	-538.97	0.00000
cutoff2.22:fold6	-0.6498	0.0012	-563.29	0.00000
cutoff2.67:fold6	-0.6707	0.0012	-579.62	0.00000
cutoff3.11:fold6	-0.6829	0.0012	-587.97	0.00000
cutoff3.56:fold6	-0.6935	0.0012	-594.73	0.00000
cutoff4:fold6	-0.6963	0.0012	-594.59	0.00000
ngenes150000:cutoff0.44	-0.0065	0.0011	-6.13	0.00000
ngenes50000:cutoff0.44	0.0079	0.0013	5.89	0.00000
ngenes150000:cutoff0.89	-0.0060	0.0011	-5.66	0.00000
ngenes50000:cutoff0.89	0.0211	0.0013	16.03	0.00000
ngenes150000:cutoff1.33	-0.0098	0.0011	-9.32	0.00000
ngenes50000:cutoff1.33	0.0190	0.0013	14.58	0.00000
ngenes150000:cutoff1.78	-0.0196	0.0011	-18.59	0.00000
ngenes50000:cutoff1.78	0.0189	0.0013	14.49	0.00000
ngenes150000:cutoff2.22	-0.0347	0.0011	-32.77	0.00000
ngenes50000:cutoff2.22	0.0244	0.0013	18.75	0.00000
ngenes150000:cutoff2.67	-0.0522	0.0011	-49.05	0.00000
ngenes50000:cutoff2.67	0.0344	0.0013	26.45	0.00000
ngenes150000:cutoff3.11	-0.0672	0.0011	-62.91	0.00000
ngenes50000:cutoff3.11	0.0474	0.0013	36.40	0.00000
ngenes150000:cutoff3.56	-0.0783	0.0011	-72.93	0.00000
ngenes50000:cutoff3.56	0.0632	0.0013	48.43	0.00000
ngenes150000:cutoff4	-0.0871	0.0011	-80.71	0.00000
ngenes50000:cutoff4	0.0785	0.0013	60.05	0.00000
ngenes150000:fold3	-0.0165	0.0006	-29.21	0.00000
ngenes50000:fold3	0.0411	0.0007	60.91	0.00000
ngenes150000:fold6	-0.0154	0.0006	-26.64	0.00000
ngenes50000:fold6	0.0526	0.0007	75.83	0.00000

Table D.2: GLM output from final Robust EdgeR sensitivity model. Modelled using pre-filtered data.

	Df	Deviance	AIC
<none>		224684.38	631245.99
pout	1	224684.38	631244.00
cutoff:fold	18	1040551.67	1447077.29
ngenes:cutoff	18	270168.52	676694.14
ngenes:fold	4	235389.25	641942.86

Table D.3: Summary of the changes in fit that would result from dropping terms from the DESeq2 sensitivity model. Only removable terms are shown. Modelled using pre-filtered data.

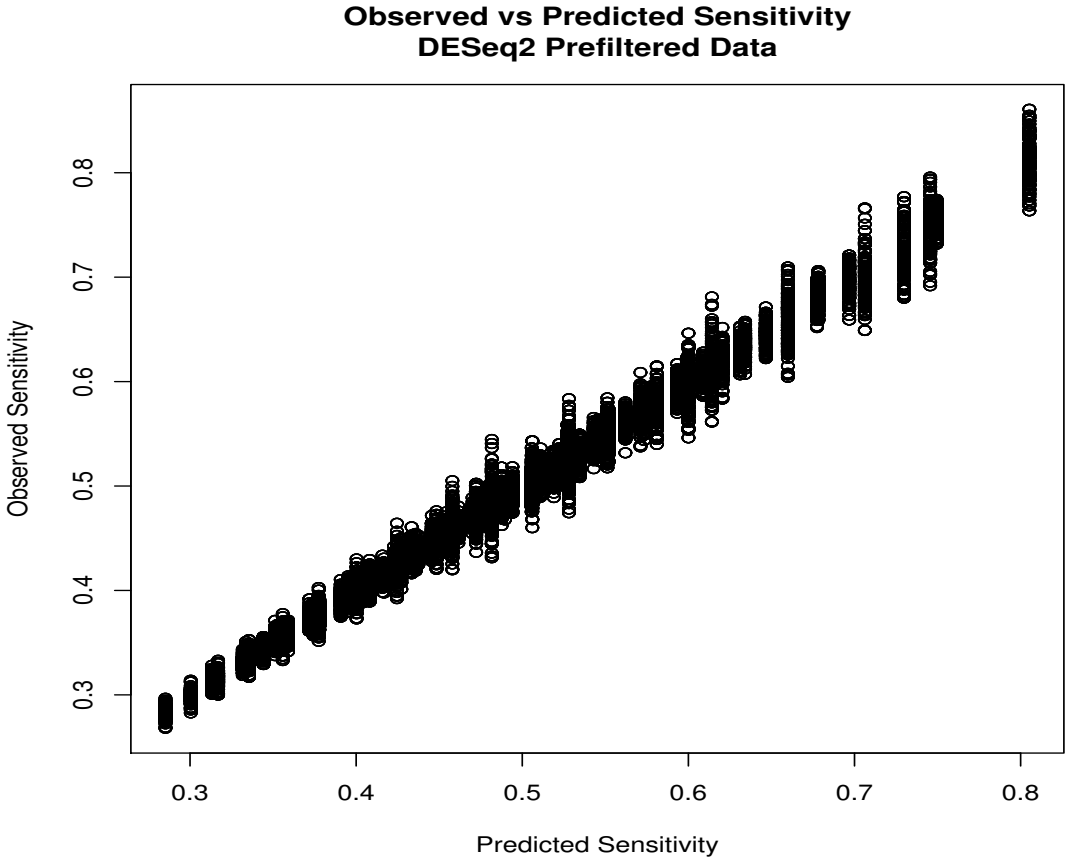

Figure D.2: Observed sensitivity vs predicted sensitivity values for the DESeq2 pre-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	0.1125	0.0007	155.24	0.00000
ngenes150000	-0.1859	0.0008	-228.12	0.00000
ngenes50000	0.2931	0.0010	289.96	0.00000
cutoff0.44	-0.0884	0.0010	-89.32	0.00000
cutoff0.89	-0.2236	0.0010	-226.84	0.00000
cutoff1.33	-0.3198	0.0010	-324.29	0.00000
cutoff1.78	-0.4058	0.0010	-410.80	0.00000
cutoff2.22	-0.4842	0.0010	-488.70	0.00000
cutoff2.67	-0.5567	0.0010	-559.95	0.00000
cutoff3.11	-0.6262	0.0010	-627.29	0.00000
cutoff3.56	-0.6921	0.0010	-690.30	0.00000
cutoff4	-0.7571	0.0010	-751.47	0.00000
fold3	0.5503	0.0009	640.51	0.00000
fold6	0.9625	0.0009	1057.92	0.00000
cutoff0.44:fold3	-0.0828	0.0011	-74.75	0.00000
cutoff0.89:fold3	-0.1896	0.0011	-171.89	0.00000
cutoff1.33:fold3	-0.2537	0.0011	-230.12	0.00000
cutoff1.78:fold3	-0.2989	0.0011	-270.69	0.00000
cutoff2.22:fold3	-0.3329	0.0011	-300.59	0.00000
cutoff2.67:fold3	-0.3577	0.0011	-321.78	0.00000
cutoff3.11:fold3	-0.3765	0.0011	-337.30	0.00000
cutoff3.56:fold3	-0.3932	0.0011	-350.52	0.00000
cutoff4:fold3	-0.4042	0.0011	-358.57	0.00000
cutoff0.44:fold6	-0.2448	0.0012	-210.90	0.00000
cutoff0.89:fold6	-0.4739	0.0011	-412.91	0.00000
cutoff1.33:fold6	-0.5719	0.0011	-499.33	0.00000
cutoff1.78:fold6	-0.6349	0.0011	-554.02	0.00000
cutoff2.22:fold6	-0.6685	0.0011	-582.07	0.00000
cutoff2.67:fold6	-0.6925	0.0012	-601.01	0.00000
cutoff3.11:fold6	-0.7069	0.0012	-611.28	0.00000
cutoff3.56:fold6	-0.7191	0.0012	-619.34	0.00000
cutoff4:fold6	-0.7232	0.0012	-620.23	0.00000
ngenes150000:cutoff0.44	-0.0070	0.0011	-6.59	0.00000
ngenes50000:cutoff0.44	0.0095	0.0013	7.22	0.00000
ngenes150000:cutoff0.89	-0.0071	0.0011	-6.77	0.00000
ngenes50000:cutoff0.89	0.0237	0.0013	18.23	0.00000
ngenes150000:cutoff1.33	-0.0111	0.0010	-10.61	0.00000
ngenes50000:cutoff1.33	0.0223	0.0013	17.25	0.00000
ngenes150000:cutoff1.78	-0.0211	0.0011	-20.10	0.00000
ngenes50000:cutoff1.78	0.0231	0.0013	17.85	0.00000
ngenes150000:cutoff2.22	-0.0359	0.0011	-34.04	0.00000
ngenes50000:cutoff2.22	0.0289	0.0013	22.43	0.00000
ngenes150000:cutoff2.67	-0.0532	0.0011	-50.20	0.00000
ngenes50000:cutoff2.67	0.0392	0.0013	30.37	0.00000
ngenes150000:cutoff3.11	-0.0684	0.0011	-64.25	0.00000
ngenes50000:cutoff3.11	0.0521	0.0013	40.35	0.00000
ngenes150000:cutoff3.56	-0.0796	0.0011	-74.46	0.00000
ngenes50000:cutoff3.56	0.0679	0.0013	52.45	0.00000
ngenes150000:cutoff4	-0.0883	0.0011	-82.11	0.00000
ngenes50000:cutoff4	0.0835	0.0013	64.36	0.00000
ngenes150000:fold3	-0.0121	0.0006	-21.49	0.00000
ngenes50000:fold3	0.0378	0.0007	56.22	0.00000
ngenes150000:fold6	-0.0120	0.0006	-20.84	0.00000
ngenes50000:fold6	0.0513	0.0007	73.97	0.00000

Table D.4: GLM output from final DESeq2 sensitivity model. Modelled using pre-filtered data.

	Df	Deviance	AIC
<none>		71711.13	404298.07
pout	1	71711.13	404296.07
cutoff:fold	18	117885.01	450435.95
ngenes:cutoff	18	83159.53	415710.47
ngenes:fold	4	74288.21	406867.15

Table D.5: Summary of the changes in fit that would result from dropping terms from the SAMSeq sensitivity model. Only removable terms are shown. Modelled using pre-filtered data.

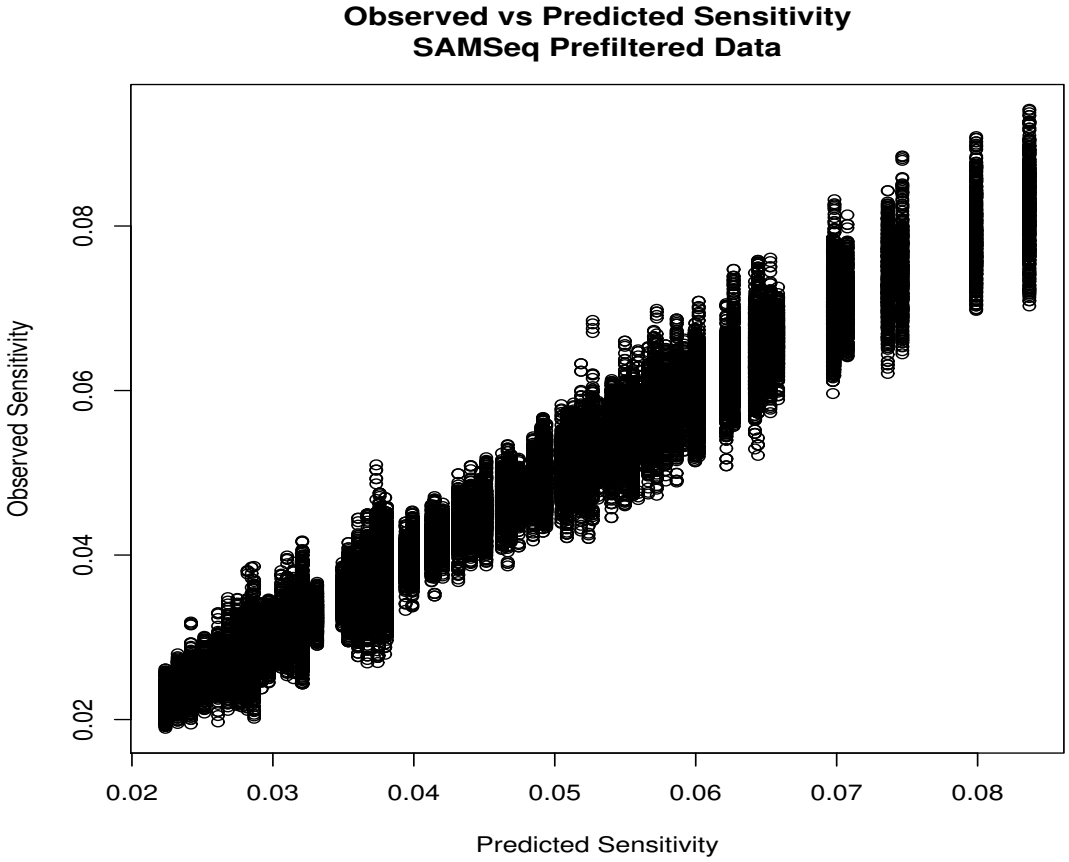

Figure D.3: Observed sensitivity vs predicted sensitivity values for the SAMSeq pre-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-3.4058	0.0019	-1798.76	0.00000
ngenes150000	-0.1170	0.0019	-60.42	0.00000
ngenes50000	0.1560	0.0022	71.38	0.00000
cutoff0.44	-0.0010	0.0025	-0.39	0.69671
cutoff0.89	-0.0021	0.0026	-0.81	0.41575
cutoff1.33	-0.0209	0.0026	-8.12	0.00000
cutoff1.78	-0.0361	0.0026	-13.95	0.00000
cutoff2.22	-0.0502	0.0026	-19.26	0.00000
cutoff2.67	-0.0807	0.0026	-30.71	0.00000
cutoff3.11	-0.0981	0.0026	-37.08	0.00000
cutoff3.56	-0.1245	0.0027	-46.68	0.00000
cutoff4	-0.1494	0.0027	-55.56	0.00000
fold3	0.6041	0.0021	285.58	0.00000
fold6	0.8886	0.0021	432.69	0.00000
cutoff0.44:fold3	-0.0102	0.0027	-3.76	0.00017
cutoff0.89:fold3	-0.0588	0.0027	-21.44	0.00000
cutoff1.33:fold3	-0.0939	0.0028	-34.01	0.00000
cutoff1.78:fold3	-0.1194	0.0028	-42.93	0.00000
cutoff2.22:fold3	-0.1603	0.0028	-57.21	0.00000
cutoff2.67:fold3	-0.1694	0.0028	-59.94	0.00000
cutoff3.11:fold3	-0.1991	0.0029	-69.85	0.00000
cutoff3.56:fold3	-0.2142	0.0029	-74.49	0.00000
cutoff4:fold3	-0.2298	0.0029	-79.18	0.00000
cutoff0.44:fold6	-0.0568	0.0026	-21.47	0.00000
cutoff0.89:fold6	-0.1597	0.0027	-59.92	0.00000
cutoff1.33:fold6	-0.2149	0.0027	-79.91	0.00000
cutoff1.78:fold6	-0.2729	0.0027	-100.53	0.00000
cutoff2.22:fold6	-0.3142	0.0027	-114.85	0.00000
cutoff2.67:fold6	-0.3366	0.0028	-121.82	0.00000
cutoff3.11:fold6	-0.3620	0.0028	-129.92	0.00000
cutoff3.56:fold6	-0.3754	0.0028	-133.52	0.00000
cutoff4:fold6	-0.3944	0.0028	-138.94	0.00000
ngenes150000:cutoff0.44	-0.0054	0.0023	-2.32	0.02027
ngenes50000:cutoff0.44	0.0076	0.0026	2.85	0.00436
ngenes150000:cutoff0.89	-0.0149	0.0024	-6.33	0.00000
ngenes50000:cutoff0.89	0.0230	0.0027	8.58	0.00000
ngenes150000:cutoff1.33	-0.0224	0.0024	-9.35	0.00000
ngenes50000:cutoff1.33	0.0385	0.0027	14.22	0.00000
ngenes150000:cutoff1.78	-0.0329	0.0024	-13.60	0.00000
ngenes50000:cutoff1.78	0.0413	0.0027	15.09	0.00000
ngenes150000:cutoff2.22	-0.0461	0.0025	-18.80	0.00000
ngenes50000:cutoff2.22	0.0530	0.0028	19.21	0.00000
ngenes150000:cutoff2.67	-0.0542	0.0025	-21.83	0.00000
ngenes50000:cutoff2.67	0.0630	0.0028	22.62	0.00000
ngenes150000:cutoff3.11	-0.0767	0.0025	-30.52	0.00000
ngenes50000:cutoff3.11	0.0778	0.0028	27.73	0.00000
ngenes150000:cutoff3.56	-0.0908	0.0025	-35.71	0.00000
ngenes50000:cutoff3.56	0.0879	0.0028	31.08	0.00000
ngenes150000:cutoff4	-0.1049	0.0026	-40.75	0.00000
ngenes50000:cutoff4	0.0929	0.0029	32.52	0.00000
ngenes150000:fold3	0.0295	0.0015	19.74	0.00000
ngenes50000:fold3	-0.0301	0.0016	-18.26	0.00000
ngenes150000:fold6	0.0451	0.0015	30.79	0.00000
ngenes50000:fold6	-0.0323	0.0016	-19.80	0.00000

Table D.6: GLM output from final SAMSeq sensitivity model. Modelled using pre-filtered data.

D.1.2 Specificity

	Df	Deviance	AIC
<none>		264093.70	716786.25
pout	1	264094.01	716784.56
cutoff:fold	18	264756.51	717413.05
ngenes:cutoff	18	649312.32	1101968.87
ngenes:fold	4	264292.76	716977.31

Table D.7: Summary of the changes in fit that would result from dropping terms from the EdgeR specificity model. Only removable terms are shown. Modelled using pre-filtered data.

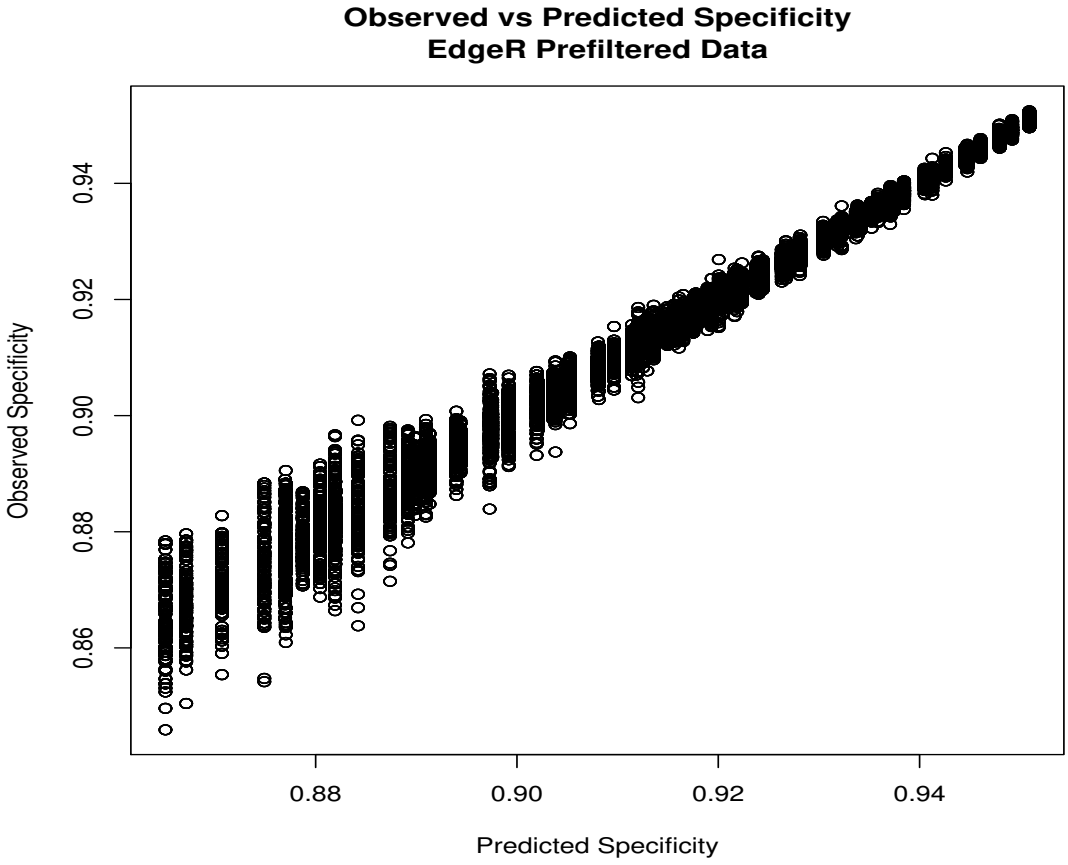

Figure D.4: Observed specificity vs predicted specificity values for the EdgeR pre-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	1.9966	0.0004	5414.03	0.00000
ngenes150000	0.0678	0.0004	161.82	0.00000
ngenes50000	-0.0895	0.0005	-184.82	0.00000
cutoff0.44	0.1354	0.0005	265.21	0.00000
cutoff0.89	0.2934	0.0005	555.25	0.00000
cutoff1.33	0.3875	0.0005	717.39	0.00000
cutoff1.78	0.4670	0.0006	847.68	0.00000
cutoff2.22	0.5347	0.0006	953.64	0.00000
cutoff2.67	0.5978	0.0006	1048.40	0.00000
cutoff3.11	0.6554	0.0006	1131.41	0.00000
cutoff3.56	0.7105	0.0006	1208.08	0.00000
cutoff4	0.7646	0.0006	1280.37	0.00000
fold3	-0.0320	0.0004	-72.52	0.00000
fold6	-0.0516	0.0004	-115.51	0.00000
cutoff0.44:fold3	0.0004	0.0006	0.78	0.43531
cutoff0.89:fold3	-0.0008	0.0006	-1.44	0.15063
cutoff1.33:fold3	-0.0020	0.0006	-3.24	0.00118
cutoff1.78:fold3	-0.0036	0.0006	-5.92	0.00000
cutoff2.22:fold3	-0.0045	0.0006	-7.12	0.00000
cutoff2.67:fold3	-0.0047	0.0006	-7.47	0.00000
cutoff3.11:fold3	-0.0048	0.0006	-7.37	0.00000
cutoff3.56:fold3	-0.0051	0.0007	-7.83	0.00000
cutoff4:fold3	-0.0046	0.0007	-6.89	0.00000
cutoff0.44:fold6	0.0019	0.0006	3.27	0.00108
cutoff0.89:fold6	0.0021	0.0006	3.46	0.00053
cutoff1.33:fold6	-0.0014	0.0006	-2.33	0.01968
cutoff1.78:fold6	-0.0030	0.0006	-4.78	0.00000
cutoff2.22:fold6	-0.0048	0.0006	-7.68	0.00000
cutoff2.67:fold6	-0.0059	0.0006	-9.12	0.00000
cutoff3.11:fold6	-0.0072	0.0007	-11.09	0.00000
cutoff3.56:fold6	-0.0084	0.0007	-12.66	0.00000
cutoff4:fold6	-0.0079	0.0007	-11.73	0.00000
ngenes150000:cutoff0.44	0.0193	0.0005	35.78	0.00000
ngenes50000:cutoff0.44	-0.0314	0.0006	-50.21	0.00000
ngenes150000:cutoff0.89	0.0326	0.0006	58.28	0.00000
ngenes50000:cutoff0.89	-0.0642	0.0006	-99.73	0.00000
ngenes150000:cutoff1.33	0.0461	0.0006	80.42	0.00000
ngenes50000:cutoff1.33	-0.0741	0.0007	-112.81	0.00000
ngenes150000:cutoff1.78	0.0625	0.0006	106.72	0.00000
ngenes50000:cutoff1.78	-0.0844	0.0007	-126.34	0.00000
ngenes150000:cutoff2.22	0.0812	0.0006	136.12	0.00000
ngenes50000:cutoff2.22	-0.0968	0.0007	-142.90	0.00000
ngenes150000:cutoff2.67	0.0981	0.0006	161.33	0.00000
ngenes50000:cutoff2.67	-0.1132	0.0007	-164.87	0.00000
ngenes150000:cutoff3.11	0.1152	0.0006	185.99	0.00000
ngenes50000:cutoff3.11	-0.1305	0.0007	-187.76	0.00000
ngenes150000:cutoff3.56	0.1259	0.0006	199.97	0.00000
ngenes50000:cutoff3.56	-0.1478	0.0007	-210.03	0.00000
ngenes150000:cutoff4	0.1347	0.0006	210.25	0.00000
ngenes50000:cutoff4	-0.1677	0.0007	-235.54	0.00000
ngenes150000:fold3	0.0006	0.0003	1.70	0.08827
ngenes50000:fold3	0.0008	0.0004	2.05	0.04071
ngenes150000:fold6	-0.0019	0.0003	-5.54	0.00000
ngenes50000:fold6	0.0027	0.0004	6.84	0.00000

Table D.8: GLM output from final EdgeR specificity model. Modelled using pre-filtered data.

	Df	Deviance	AIC
<none>		261321.99	713269.22
pout	1	261322.32	713267.56
cutoff:fold	18	261932.27	713843.50
ngenes:cutoff	18	633694.06	1085605.29
ngenes:fold	4	261530.36	713469.59

Table D.9: Summary of the changes in fit that would result from dropping terms from the Robust EdgeR specificity model. Only removable terms are shown. Modelled using pre-filtered data.

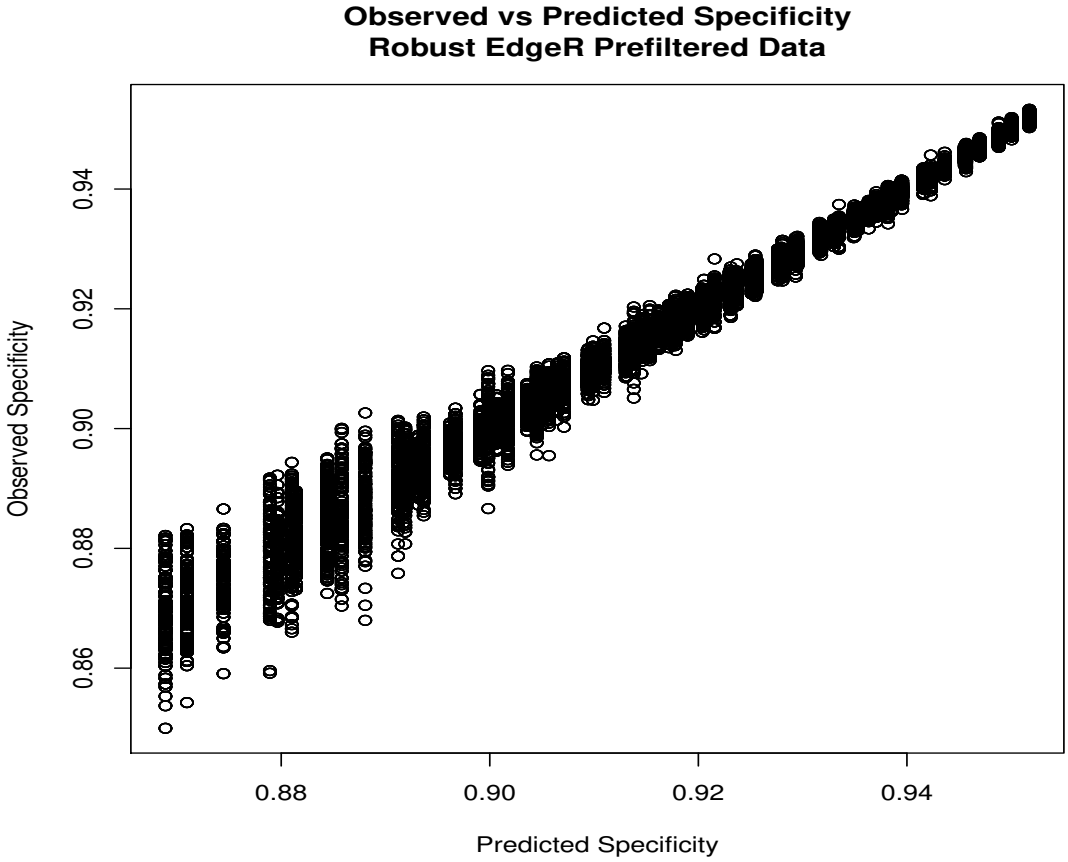

Figure D.5: Observed specificity vs predicted specificity values for the Robust EdgeR pre-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.0349	0.0004	5441.06	0.00000
ngenes150000	0.0682	0.0004	160.74	0.00000
ngenes50000	-0.0939	0.0005	-191.45	0.00000
cutoff0.44	0.1262	0.0005	244.04	0.00000
cutoff0.89	0.2775	0.0005	519.22	0.00000
cutoff1.33	0.3697	0.0005	676.95	0.00000
cutoff1.78	0.4486	0.0006	805.53	0.00000
cutoff2.22	0.5159	0.0006	910.38	0.00000
cutoff2.67	0.5788	0.0006	1004.50	0.00000
cutoff3.11	0.6363	0.0006	1087.10	0.00000
cutoff3.56	0.6912	0.0006	1163.18	0.00000
cutoff4	0.7447	0.0006	1234.49	0.00000
fold3	-0.0328	0.0004	-73.49	0.00000
fold6	-0.0527	0.0005	-116.44	0.00000
cutoff0.44:fold3	0.0004	0.0006	0.72	0.47411
cutoff0.89:fold3	-0.0006	0.0006	-1.04	0.29626
cutoff1.33:fold3	-0.0015	0.0006	-2.40	0.01654
cutoff1.78:fold3	-0.0033	0.0006	-5.29	0.00000
cutoff2.22:fold3	-0.0040	0.0006	-6.36	0.00000
cutoff2.67:fold3	-0.0045	0.0006	-6.94	0.00000
cutoff3.11:fold3	-0.0046	0.0007	-6.98	0.00000
cutoff3.56:fold3	-0.0049	0.0007	-7.41	0.00000
cutoff4:fold3	-0.0042	0.0007	-6.27	0.00000
cutoff0.44:fold6	0.0018	0.0006	3.16	0.00160
cutoff0.89:fold6	0.0023	0.0006	3.83	0.00013
cutoff1.33:fold6	-0.0007	0.0006	-1.15	0.24964
cutoff1.78:fold6	-0.0023	0.0006	-3.75	0.00018
cutoff2.22:fold6	-0.0044	0.0006	-6.98	0.00000
cutoff2.67:fold6	-0.0056	0.0006	-8.58	0.00000
cutoff3.11:fold6	-0.0068	0.0007	-10.35	0.00000
cutoff3.56:fold6	-0.0079	0.0007	-11.80	0.00000
cutoff4:fold6	-0.0076	0.0007	-11.16	0.00000
ngenes150000:cutoff0.44	0.0189	0.0005	34.61	0.00000
ngenes50000:cutoff0.44	-0.0303	0.0006	-47.90	0.00000
ngenes150000:cutoff0.89	0.0327	0.0006	57.76	0.00000
ngenes50000:cutoff0.89	-0.0628	0.0007	-96.57	0.00000
ngenes150000:cutoff1.33	0.0461	0.0006	79.63	0.00000
ngenes50000:cutoff1.33	-0.0729	0.0007	-109.90	0.00000
ngenes150000:cutoff1.78	0.0626	0.0006	105.76	0.00000
ngenes50000:cutoff1.78	-0.0835	0.0007	-123.79	0.00000
ngenes150000:cutoff2.22	0.0815	0.0006	135.09	0.00000
ngenes50000:cutoff2.22	-0.0958	0.0007	-140.01	0.00000
ngenes150000:cutoff2.67	0.0981	0.0006	159.59	0.00000
ngenes50000:cutoff2.67	-0.1119	0.0007	-161.36	0.00000
ngenes150000:cutoff3.11	0.1148	0.0006	183.42	0.00000
ngenes50000:cutoff3.11	-0.1290	0.0007	-183.81	0.00000
ngenes150000:cutoff3.56	0.1250	0.0006	196.50	0.00000
ngenes50000:cutoff3.56	-0.1461	0.0007	-205.63	0.00000
ngenes150000:cutoff4	0.1338	0.0006	206.67	0.00000
ngenes50000:cutoff4	-0.1653	0.0007	-230.07	0.00000
ngenes150000:fold3	0.0008	0.0003	2.19	0.02870
ngenes50000:fold3	0.0013	0.0004	3.29	0.00101
ngenes150000:fold6	-0.0019	0.0004	-5.30	0.00000
ngenes50000:fold6	0.0030	0.0004	7.55	0.00000

Table D.10: GLM output from final Robust EdgeR specificity model. Modelled using pre-filtered data.

	Df	Deviance	AIC
<none>		276302.68	729093.62
pout	1	276303.56	729092.50
cutoff:fold	18	281942.18	734697.12
ngenes:cutoff	18	555695.43	1008450.38
ngenes:fold	4	276879.45	729662.39

Table D.11: Summary of the changes in fit that would result from dropping terms from the DESeq2 specificity model. Only removable terms are shown. Modelled using pre-filtered data.

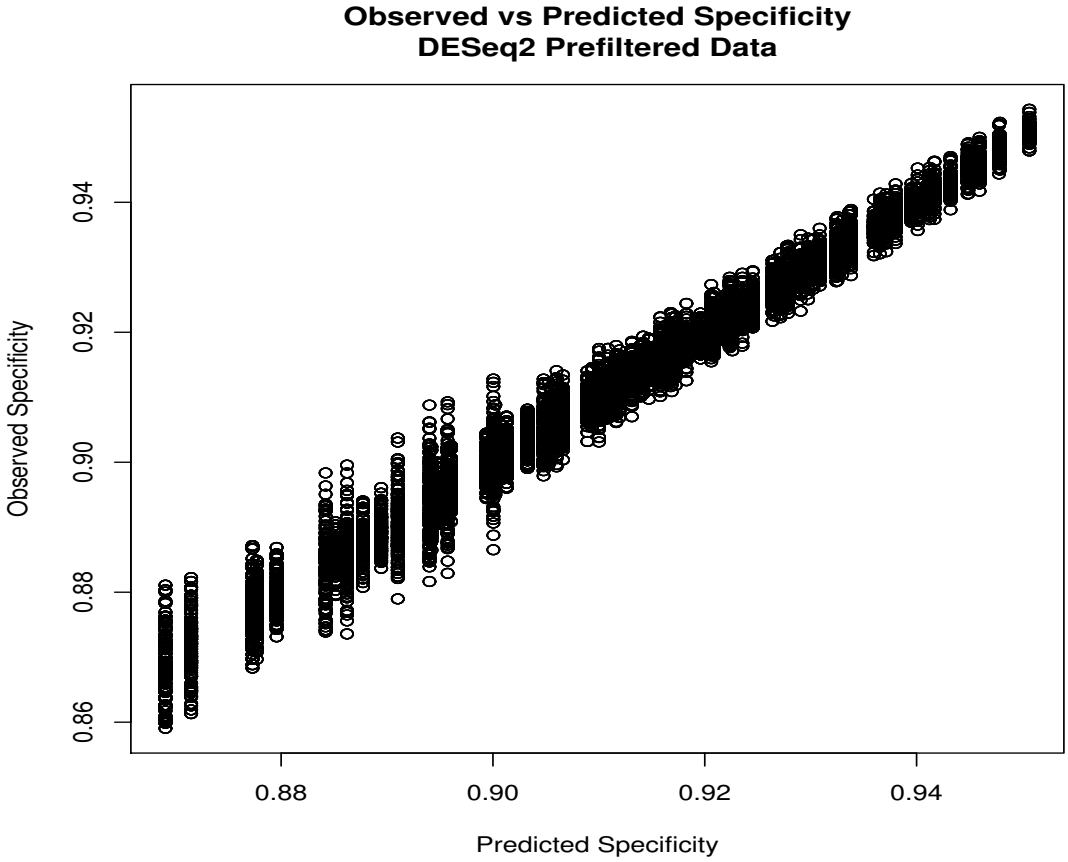

Figure D.6: Observed specificity vs predicted specificity values for the DESeq2 pre-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.1012	0.0004	5493.61	0.00000
ngenes150000	0.0965	0.0004	222.23	0.00000
ngenes50000	-0.1341	0.0005	-271.18	0.00000
cutoff0.44	0.0985	0.0005	186.83	0.00000
cutoff0.89	0.2231	0.0005	412.08	0.00000
cutoff1.33	0.3045	0.0006	551.96	0.00000
cutoff1.78	0.3729	0.0006	665.03	0.00000
cutoff2.22	0.4336	0.0006	761.61	0.00000
cutoff2.67	0.4894	0.0006	847.52	0.00000
cutoff3.11	0.5416	0.0006	925.25	0.00000
cutoff3.56	0.5922	0.0006	998.20	0.00000
cutoff4	0.6417	0.0006	1067.47	0.00000
fold3	-0.0484	0.0005	-106.48	0.00000
fold6	-0.0683	0.0005	-148.24	0.00000
cutoff0.44:fold3	-0.0010	0.0006	-1.69	0.09058
cutoff0.89:fold3	-0.0029	0.0006	-4.76	0.00000
cutoff1.33:fold3	-0.0049	0.0006	-7.94	0.00000
cutoff1.78:fold3	-0.0067	0.0006	-10.69	0.00000
cutoff2.22:fold3	-0.0080	0.0006	-12.65	0.00000
cutoff2.67:fold3	-0.0097	0.0006	-15.02	0.00000
cutoff3.11:fold3	-0.0114	0.0007	-17.53	0.00000
cutoff3.56:fold3	-0.0116	0.0007	-17.55	0.00000
cutoff4:fold3	-0.0123	0.0007	-18.45	0.00000
cutoff0.44:fold6	0.0030	0.0006	5.04	0.00000
cutoff0.89:fold6	0.0013	0.0006	2.06	0.03935
cutoff1.33:fold6	-0.0035	0.0006	-5.60	0.00000
cutoff1.78:fold6	-0.0064	0.0006	-10.11	0.00000
cutoff2.22:fold6	-0.0121	0.0006	-18.96	0.00000
cutoff2.67:fold6	-0.0166	0.0006	-25.60	0.00000
cutoff3.11:fold6	-0.0216	0.0007	-32.93	0.00000
cutoff3.56:fold6	-0.0252	0.0007	-37.87	0.00000
cutoff4:fold6	-0.0307	0.0007	-45.49	0.00000
ngenes150000:cutoff0.44	0.0177	0.0006	31.65	0.00000
ngenes50000:cutoff0.44	-0.0235	0.0006	-36.83	0.00000
ngenes150000:cutoff0.89	0.0299	0.0006	51.97	0.00000
ngenes50000:cutoff0.89	-0.0495	0.0007	-75.98	0.00000
ngenes150000:cutoff1.33	0.0414	0.0006	70.68	0.00000
ngenes50000:cutoff1.33	-0.0599	0.0007	-90.45	0.00000
ngenes150000:cutoff1.78	0.0555	0.0006	92.96	0.00000
ngenes50000:cutoff1.78	-0.0689	0.0007	-102.62	0.00000
ngenes150000:cutoff2.22	0.0706	0.0006	116.47	0.00000
ngenes50000:cutoff2.22	-0.0803	0.0007	-118.06	0.00000
ngenes150000:cutoff2.67	0.0866	0.0006	140.58	0.00000
ngenes50000:cutoff2.67	-0.0936	0.0007	-136.08	0.00000
ngenes150000:cutoff3.11	0.1011	0.0006	161.60	0.00000
ngenes50000:cutoff3.11	-0.1069	0.0007	-153.91	0.00000
ngenes150000:cutoff3.56	0.1106	0.0006	174.24	0.00000
ngenes50000:cutoff3.56	-0.1223	0.0007	-174.15	0.00000
ngenes150000:cutoff4	0.1188	0.0006	184.46	0.00000
ngenes50000:cutoff4	-0.1380	0.0007	-194.65	0.00000
ngenes150000:fold3	0.0013	0.0003	3.89	0.00010
ngenes50000:fold3	-0.0043	0.0004	-11.16	0.00000
ngenes150000:fold6	0.0030	0.0004	8.56	0.00000
ngenes50000:fold6	-0.0059	0.0004	-15.11	0.00000

Table D.12: GLM output from final DESeq2 specificity model. Modelled using pre-filtered data.

	Df	Deviance	AIC
<none>		282673.14	709958.75
pout	1	282673.39	709957.00
cutoff:fold	18	704647.02	1131896.63
ngenes:cutoff	18	384428.98	811678.59
ngenes:fold	4	308918.23	736195.84

Table D.13: Summary of the changes in fit that would result from dropping terms from the SAMSeq specificity model. Only removable terms are shown. Modelled using pre-filtered data.

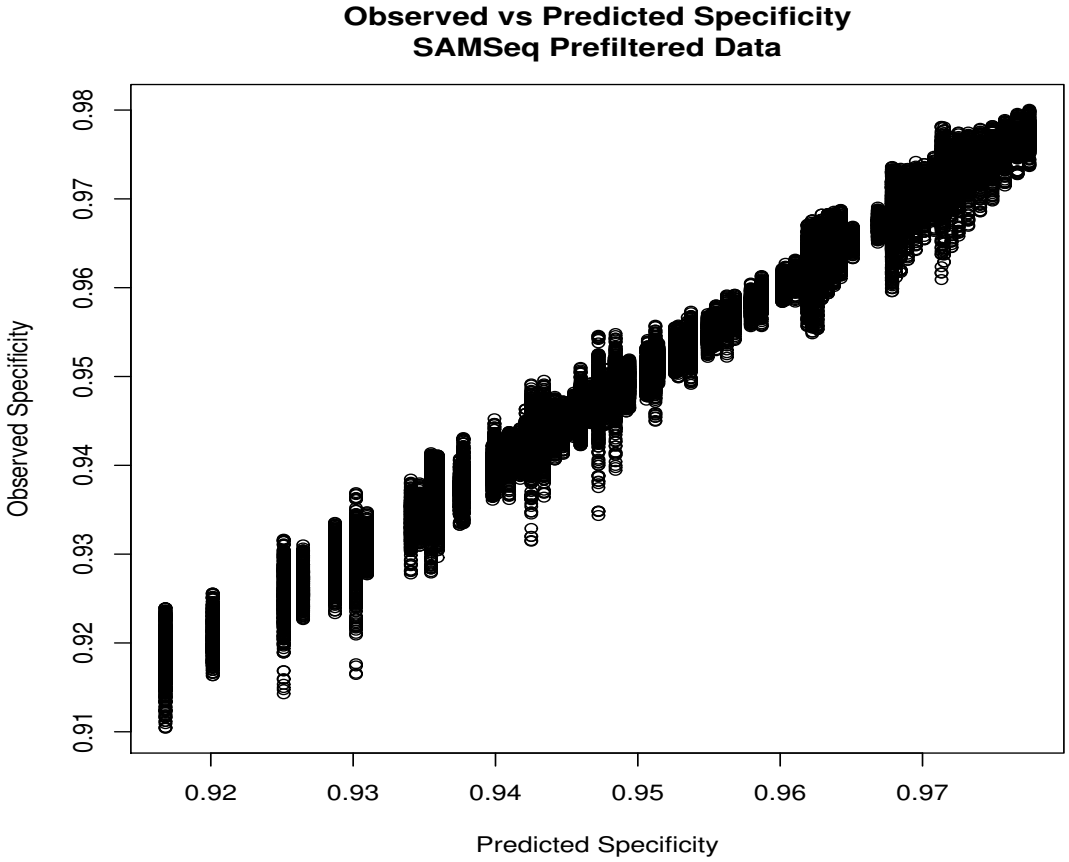

Figure D.7: Observed specificity vs predicted specificity values for the SAMSeq pre-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	3.4054	0.0006	5397.40	0.00000
ngenes150000	0.1174	0.0006	182.00	0.00000
ngenes50000	-0.1570	0.0007	-215.58	0.00000
cutoff0.44	-0.0010	0.0008	-1.21	0.22691
cutoff0.89	0.0083	0.0009	9.74	0.00000
cutoff1.33	0.0192	0.0009	22.35	0.00000
cutoff1.78	0.0354	0.0009	40.97	0.00000
cutoff2.22	0.0547	0.0009	62.87	0.00000
cutoff2.67	0.0748	0.0009	85.43	0.00000
cutoff3.11	0.0988	0.0009	112.07	0.00000
cutoff3.56	0.1257	0.0009	141.43	0.00000
cutoff4	0.1540	0.0009	171.86	0.00000
fold3	-0.6084	0.0007	-863.32	0.00000
fold6	-0.8914	0.0007	-1302.49	0.00000
cutoff0.44:fold3	0.0177	0.0009	19.47	0.00000
cutoff0.89:fold3	0.0575	0.0009	62.87	0.00000
cutoff1.33:fold3	0.0944	0.0009	102.65	0.00000
cutoff1.78:fold3	0.1270	0.0009	137.07	0.00000
cutoff2.22:fold3	0.1545	0.0009	165.39	0.00000
cutoff2.67:fold3	0.1813	0.0009	192.53	0.00000
cutoff3.11:fold3	0.2021	0.0009	212.80	0.00000
cutoff3.56:fold3	0.2207	0.0010	230.33	0.00000
cutoff4:fold3	0.2368	0.0010	244.98	0.00000
cutoff0.44:fold6	0.0543	0.0009	61.61	0.00000
cutoff0.89:fold6	0.1533	0.0009	172.42	0.00000
cutoff1.33:fold6	0.2201	0.0009	245.55	0.00000
cutoff1.78:fold6	0.2720	0.0009	300.69	0.00000
cutoff2.22:fold6	0.3085	0.0009	338.03	0.00000
cutoff2.67:fold6	0.3399	0.0009	369.19	0.00000
cutoff3.11:fold6	0.3623	0.0009	390.07	0.00000
cutoff3.56:fold6	0.3809	0.0009	406.53	0.00000
cutoff4:fold6	0.3928	0.0009	415.60	0.00000
ngenes150000:cutoff0.44	0.0076	0.0008	9.87	0.00000
ngenes50000:cutoff0.44	-0.0089	0.0009	-10.04	0.00000
ngenes150000:cutoff0.89	0.0173	0.0008	21.97	0.00000
ngenes50000:cutoff0.89	-0.0275	0.0009	-30.82	0.00000
ngenes150000:cutoff1.33	0.0247	0.0008	31.07	0.00000
ngenes50000:cutoff1.33	-0.0373	0.0009	-41.38	0.00000
ngenes150000:cutoff1.78	0.0347	0.0008	42.98	0.00000
ngenes50000:cutoff1.78	-0.0465	0.0009	-51.05	0.00000
ngenes150000:cutoff2.22	0.0479	0.0008	58.63	0.00000
ngenes50000:cutoff2.22	-0.0550	0.0009	-59.84	0.00000
ngenes150000:cutoff2.67	0.0623	0.0008	75.35	0.00000
ngenes50000:cutoff2.67	-0.0664	0.0009	-71.61	0.00000
ngenes150000:cutoff3.11	0.0747	0.0008	89.28	0.00000
ngenes50000:cutoff3.11	-0.0781	0.0009	-83.46	0.00000
ngenes150000:cutoff3.56	0.0857	0.0008	101.13	0.00000
ngenes50000:cutoff3.56	-0.0928	0.0009	-98.35	0.00000
ngenes150000:cutoff4	0.0949	0.0009	110.61	0.00000
ngenes50000:cutoff4	-0.1076	0.0010	-113.10	0.00000
ngenes150000:fold3	-0.0268	0.0005	-53.78	0.00000
ngenes50000:fold3	0.0340	0.0005	61.81	0.00000
ngenes150000:fold6	-0.0417	0.0005	-85.36	0.00000
ngenes50000:fold6	0.0428	0.0005	78.83	0.00000

Table D.14: GLM output from final SAMSeq specificity model. Modelled using pre-filtered data.

D.2 Post-filtered Data

D.2.1 Sensitivity

	Df	Deviance	AIC
<none>		220779.21	563111.02
pout	1	220779.21	563109.02
cutoff:fold	18	858376.91	1200672.72
ngenes:cutoff	18	255773.84	598069.66
ngenes:fold	4	229283.31	571607.13

Table D.15: Summary of the changes in fit that would result from dropping terms from the EdgeR sensitivity model. Only removable terms are shown. Modelled using post-filtered data.

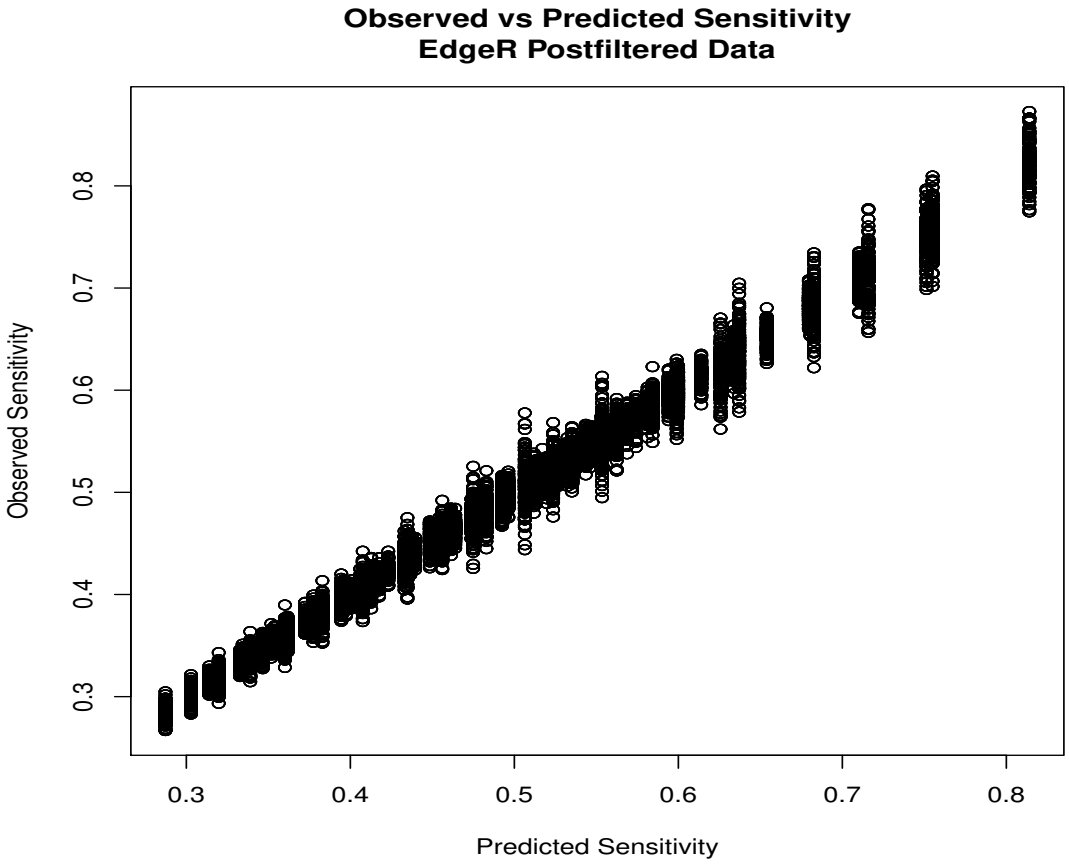

Figure D.8: Observed sensitivity vs predicted sensitivity values for the EdgeR post-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	0.2148	0.0008	269.71	0.00000
ngenes150000	-0.1892	0.0009	-221.11	0.00000
ngenes50000	0.2992	0.0012	250.34	0.00000
cutoff0.44	-0.1201	0.0011	-111.16	0.00000
cutoff0.89	-0.2830	0.0011	-263.24	0.00000
cutoff1.33	-0.3911	0.0011	-363.75	0.00000
cutoff1.78	-0.4842	0.0011	-449.54	0.00000
cutoff2.22	-0.5675	0.0011	-525.33	0.00000
cutoff2.67	-0.6433	0.0011	-593.31	0.00000
cutoff3.11	-0.7153	0.0011	-656.98	0.00000
cutoff3.56	-0.7829	0.0011	-715.79	0.00000
cutoff4	-0.8489	0.0011	-772.40	0.00000
fold3	0.5512	0.0009	594.61	0.00000
fold6	0.9109	0.0010	948.04	0.00000
cutoff0.44:fold3	-0.0965	0.0012	-81.14	0.00000
cutoff0.89:fold3	-0.2065	0.0012	-174.67	0.00000
cutoff1.33:fold3	-0.2684	0.0012	-227.14	0.00000
cutoff1.78:fold3	-0.3112	0.0012	-262.85	0.00000
cutoff2.22:fold3	-0.3429	0.0012	-288.73	0.00000
cutoff2.67:fold3	-0.3661	0.0012	-306.92	0.00000
cutoff3.11:fold3	-0.3838	0.0012	-320.24	0.00000
cutoff3.56:fold3	-0.3995	0.0012	-331.69	0.00000
cutoff4:fold3	-0.4100	0.0012	-338.56	0.00000
cutoff0.44:fold6	-0.2493	0.0012	-203.74	0.00000
cutoff0.89:fold6	-0.4617	0.0012	-381.45	0.00000
cutoff1.33:fold6	-0.5489	0.0012	-454.25	0.00000
cutoff1.78:fold6	-0.6049	0.0012	-500.09	0.00000
cutoff2.22:fold6	-0.6337	0.0012	-522.48	0.00000
cutoff2.67:fold6	-0.6540	0.0012	-537.38	0.00000
cutoff3.11:fold6	-0.6657	0.0012	-544.74	0.00000
cutoff3.56:fold6	-0.6760	0.0012	-550.77	0.00000
cutoff4:fold6	-0.6786	0.0012	-550.32	0.00000
ngenes150000:cutoff0.44	-0.0060	0.0011	-5.46	0.00000
ngenes50000:cutoff0.44	0.0073	0.0016	4.70	0.00000
ngenes150000:cutoff0.89	-0.0049	0.0011	-4.47	0.00001
ngenes50000:cutoff0.89	0.0201	0.0015	13.18	0.00000
ngenes150000:cutoff1.33	-0.0085	0.0011	-7.80	0.00000
ngenes50000:cutoff1.33	0.0178	0.0015	11.76	0.00000
ngenes150000:cutoff1.78	-0.0184	0.0011	-16.86	0.00000
ngenes50000:cutoff1.78	0.0177	0.0015	11.75	0.00000
ngenes150000:cutoff2.22	-0.0330	0.0011	-30.09	0.00000
ngenes50000:cutoff2.22	0.0235	0.0015	15.56	0.00000
ngenes150000:cutoff2.67	-0.0502	0.0011	-45.68	0.00000
ngenes50000:cutoff2.67	0.0335	0.0015	22.24	0.00000
ngenes150000:cutoff3.11	-0.0652	0.0011	-59.04	0.00000
ngenes50000:cutoff3.11	0.0464	0.0015	30.78	0.00000
ngenes150000:cutoff3.56	-0.0765	0.0011	-68.95	0.00000
ngenes50000:cutoff3.56	0.0620	0.0015	41.06	0.00000
ngenes150000:cutoff4	-0.0854	0.0011	-76.50	0.00000
ngenes50000:cutoff4	0.0770	0.0015	50.91	0.00000
ngenes150000:fold3	-0.0140	0.0006	-23.65	0.00000
ngenes50000:fold3	0.0404	0.0008	51.16	0.00000
ngenes150000:fold6	-0.0123	0.0006	-20.65	0.00000
ngenes50000:fold6	0.0515	0.0008	64.63	0.00000

Table D.16: GLM output from final EdgeR sensitivity model. Modelled using post-filtered data.

	Df	Deviance	AIC
<none>		223351.66	565676.77
pout	1	223351.66	565674.77
cutoff:fold	18	872323.08	1214612.19
ngenes:cutoff	18	258262.21	600551.32
ngenes:fold	4	231852.52	574169.63

Table D.17: Summary of the changes in fit that would result from dropping terms from the Robust EdgeR sensitivity model. Only removable terms are shown. Modelled using post-filtered data.

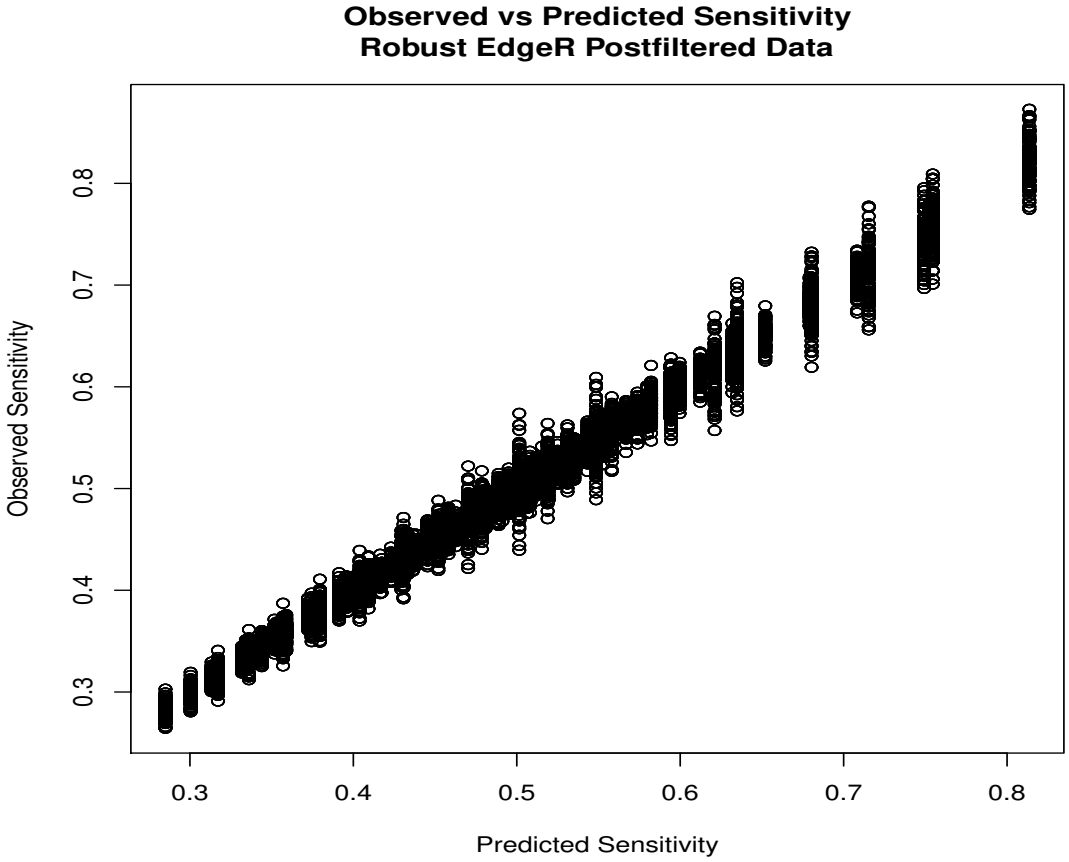

Figure D.9: Observed sensitivity vs predicted sensitivity values for the Robust EdgeR post-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	0.1947	0.0008	244.75	0.00000
ngenes150000	-0.1886	0.0009	-220.58	0.00000
ngenes50000	0.2993	0.0012	250.84	0.00000
cutoff0.44	-0.1191	0.0011	-110.35	0.00000
cutoff0.89	-0.2803	0.0011	-260.85	0.00000
cutoff1.33	-0.3873	0.0011	-360.27	0.00000
cutoff1.78	-0.4792	0.0011	-444.94	0.00000
cutoff2.22	-0.5616	0.0011	-519.85	0.00000
cutoff2.67	-0.6366	0.0011	-587.10	0.00000
cutoff3.11	-0.7078	0.0011	-650.06	0.00000
cutoff3.56	-0.7748	0.0011	-708.28	0.00000
cutoff4	-0.8403	0.0011	-764.40	0.00000
fold3	0.5608	0.0009	605.53	0.00000
fold6	0.9283	0.0010	966.69	0.00000
cutoff0.44:fold3	-0.0960	0.0012	-80.81	0.00000
cutoff0.89:fold3	-0.2061	0.0012	-174.45	0.00000
cutoff1.33:fold3	-0.2683	0.0012	-227.10	0.00000
cutoff1.78:fold3	-0.3115	0.0012	-263.19	0.00000
cutoff2.22:fold3	-0.3437	0.0012	-289.41	0.00000
cutoff2.67:fold3	-0.3672	0.0012	-307.90	0.00000
cutoff3.11:fold3	-0.3853	0.0012	-321.53	0.00000
cutoff3.56:fold3	-0.4013	0.0012	-333.23	0.00000
cutoff4:fold3	-0.4121	0.0012	-340.28	0.00000
cutoff0.44:fold6	-0.2494	0.0012	-203.90	0.00000
cutoff0.89:fold6	-0.4630	0.0012	-382.65	0.00000
cutoff1.33:fold6	-0.5511	0.0012	-456.19	0.00000
cutoff1.78:fold6	-0.6082	0.0012	-502.79	0.00000
cutoff2.22:fold6	-0.6377	0.0012	-525.81	0.00000
cutoff2.67:fold6	-0.6587	0.0012	-541.24	0.00000
cutoff3.11:fold6	-0.6711	0.0012	-549.12	0.00000
cutoff3.56:fold6	-0.6819	0.0012	-555.54	0.00000
cutoff4:fold6	-0.6850	0.0012	-555.41	0.00000
ngenes150000:cutoff0.44	-0.0061	0.0011	-5.50	0.00000
ngenes50000:cutoff0.44	0.0072	0.0015	4.66	0.00000
ngenes150000:cutoff0.89	-0.0049	0.0011	-4.48	0.00001
ngenes50000:cutoff0.89	0.0198	0.0015	13.01	0.00000
ngenes150000:cutoff1.33	-0.0084	0.0011	-7.75	0.00000
ngenes50000:cutoff1.33	0.0175	0.0015	11.58	0.00000
ngenes150000:cutoff1.78	-0.0184	0.0011	-16.85	0.00000
ngenes50000:cutoff1.78	0.0174	0.0015	11.56	0.00000
ngenes150000:cutoff2.22	-0.0329	0.0011	-30.06	0.00000
ngenes50000:cutoff2.22	0.0231	0.0015	15.32	0.00000
ngenes150000:cutoff2.67	-0.0502	0.0011	-45.62	0.00000
ngenes50000:cutoff2.67	0.0332	0.0015	22.02	0.00000
ngenes150000:cutoff3.11	-0.0651	0.0011	-58.98	0.00000
ngenes50000:cutoff3.11	0.0461	0.0015	30.59	0.00000
ngenes150000:cutoff3.56	-0.0766	0.0011	-68.97	0.00000
ngenes50000:cutoff3.56	0.0616	0.0015	40.82	0.00000
ngenes150000:cutoff4	-0.0854	0.0011	-76.55	0.00000
ngenes50000:cutoff4	0.0767	0.0015	50.71	0.00000
ngenes150000:fold3	-0.0139	0.0006	-23.50	0.00000
ngenes50000:fold3	0.0399	0.0008	50.49	0.00000
ngenes150000:fold6	-0.0128	0.0006	-21.46	0.00000
ngenes50000:fold6	0.0514	0.0008	64.56	0.00000

Table D.18: GLM output from final Robust EdgeR sensitivity model. Modelled using post-filtered data.

	Df	Deviance	AIC
<none>		203282.41	545723.59
pout	1	203282.41	545721.59
cutoff:fold	18	932211.75	1274616.93
ngenes:cutoff	18	238960.77	581365.95
ngenes:fold	4	209589.40	552022.59

Table D.19: Summary of the changes in fit that would result from dropping terms from the DESeq2 sensitivity model. Only removable terms are shown. Modelled using post-filtered data.

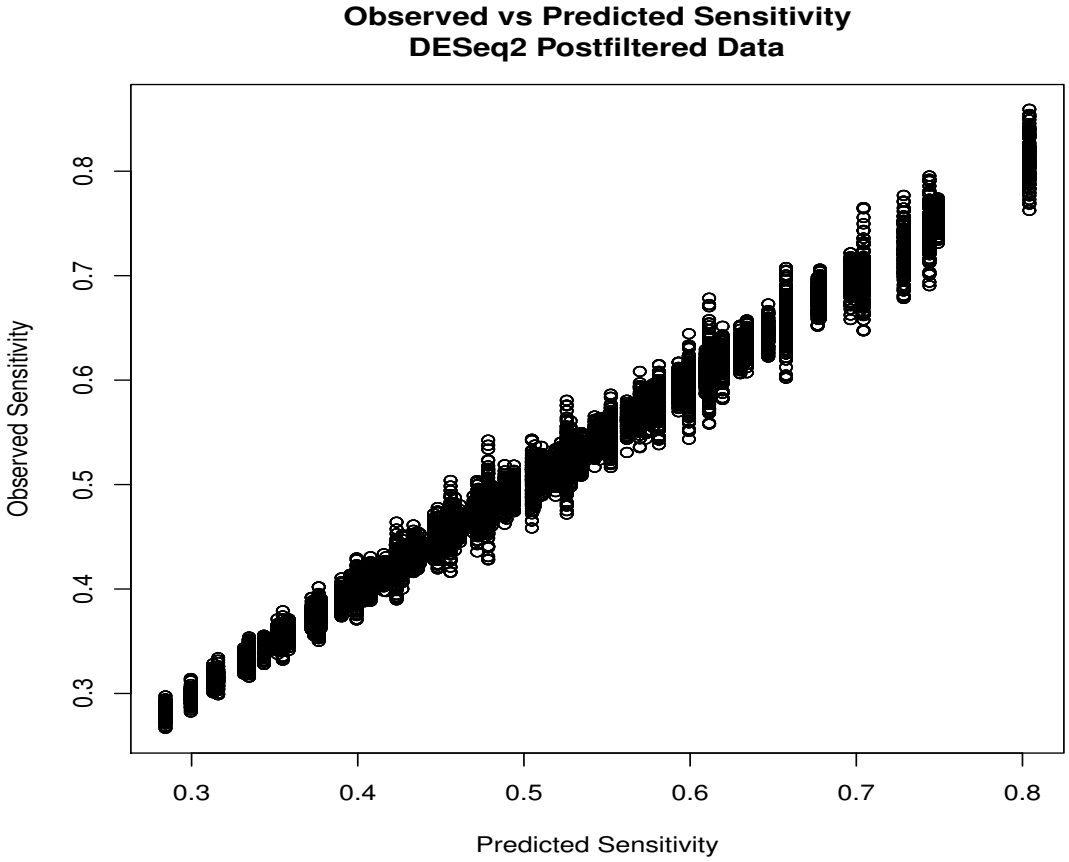

Figure D.10: Observed sensitivity vs predicted sensitivity values for the DESeq2 post-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	0.1031	0.0008	130.25	0.00000
ngenes150000	-0.1891	0.0008	-222.67	0.00000
ngenes50000	0.2994	0.0012	254.12	0.00000
cutoff0.44	-0.0840	0.0011	-78.13	0.00000
cutoff0.89	-0.2158	0.0011	-201.40	0.00000
cutoff1.33	-0.3114	0.0011	-290.59	0.00000
cutoff1.78	-0.3977	0.0011	-370.35	0.00000
cutoff2.22	-0.4766	0.0011	-442.47	0.00000
cutoff2.67	-0.5495	0.0011	-508.25	0.00000
cutoff3.11	-0.6194	0.0011	-570.51	0.00000
cutoff3.56	-0.6855	0.0011	-628.52	0.00000
cutoff4	-0.7505	0.0011	-684.76	0.00000
fold3	0.5501	0.0009	598.93	0.00000
fold6	0.9637	0.0010	1011.01	0.00000
cutoff0.44:fold3	-0.0817	0.0012	-69.19	0.00000
cutoff0.89:fold3	-0.1884	0.0012	-160.20	0.00000
cutoff1.33:fold3	-0.2525	0.0012	-214.75	0.00000
cutoff1.78:fold3	-0.2974	0.0012	-252.42	0.00000
cutoff2.22:fold3	-0.3311	0.0012	-280.06	0.00000
cutoff2.67:fold3	-0.3556	0.0012	-299.59	0.00000
cutoff3.11:fold3	-0.3744	0.0012	-313.89	0.00000
cutoff3.56:fold3	-0.3908	0.0012	-325.97	0.00000
cutoff4:fold3	-0.4018	0.0012	-333.25	0.00000
cutoff0.44:fold6	-0.2440	0.0012	-200.47	0.00000
cutoff0.89:fold6	-0.4745	0.0012	-393.83	0.00000
cutoff1.33:fold6	-0.5724	0.0012	-475.80	0.00000
cutoff1.78:fold6	-0.6348	0.0012	-527.05	0.00000
cutoff2.22:fold6	-0.6677	0.0012	-552.89	0.00000
cutoff2.67:fold6	-0.6908	0.0012	-570.05	0.00000
cutoff3.11:fold6	-0.7044	0.0012	-578.87	0.00000
cutoff3.56:fold6	-0.7161	0.0012	-585.83	0.00000
cutoff4:fold6	-0.7196	0.0012	-585.99	0.00000
ngenes150000:cutoff0.44	-0.0071	0.0011	-6.48	0.00000
ngenes50000:cutoff0.44	0.0094	0.0015	6.13	0.00000
ngenes150000:cutoff0.89	-0.0073	0.0011	-6.68	0.00000
ngenes50000:cutoff0.89	0.0231	0.0015	15.27	0.00000
ngenes150000:cutoff1.33	-0.0109	0.0011	-10.06	0.00000
ngenes50000:cutoff1.33	0.0216	0.0015	14.38	0.00000
ngenes150000:cutoff1.78	-0.0206	0.0011	-18.97	0.00000
ngenes50000:cutoff1.78	0.0220	0.0015	14.72	0.00000
ngenes150000:cutoff2.22	-0.0349	0.0011	-32.06	0.00000
ngenes50000:cutoff2.22	0.0276	0.0015	18.50	0.00000
ngenes150000:cutoff2.67	-0.0519	0.0011	-47.43	0.00000
ngenes50000:cutoff2.67	0.0376	0.0015	25.15	0.00000
ngenes150000:cutoff3.11	-0.0668	0.0011	-60.77	0.00000
ngenes50000:cutoff3.11	0.0502	0.0015	33.58	0.00000
ngenes150000:cutoff3.56	-0.0781	0.0011	-70.68	0.00000
ngenes50000:cutoff3.56	0.0655	0.0015	43.75	0.00000
ngenes150000:cutoff4	-0.0868	0.0011	-78.15	0.00000
ngenes50000:cutoff4	0.0805	0.0015	53.74	0.00000
ngenes150000:fold3	-0.0106	0.0006	-17.90	0.00000
ngenes50000:fold3	0.0345	0.0008	43.76	0.00000
ngenes150000:fold6	-0.0096	0.0006	-16.15	0.00000
ngenes50000:fold6	0.0461	0.0008	57.94	0.00000

Table D.20: GLM output from final DESeq2 sensitivity model. Modelled using post-filtered data.

	Df	Deviance	AIC
<none>		350190.03	690971.74
pout	1	350190.10	690969.82
cutoff:fold	18	1521654.32	1862400.04
ngenes:cutoff	18	390338.04	731083.76
ngenes:fold	4	357963.60	698737.32

Table D.21: Summary of the changes in fit that would result from dropping terms from the SAMSeq sensitivity model. Only removable terms are shown. Modelled using post-filtered data.

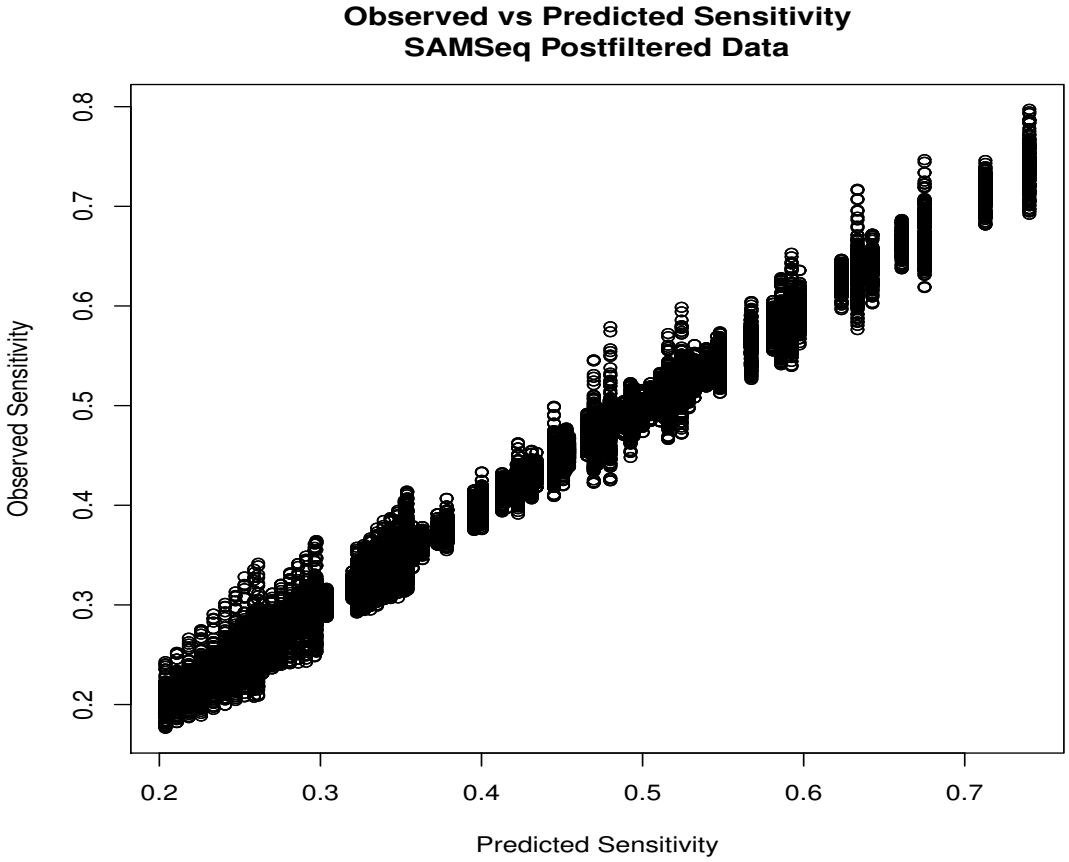

Figure D.11: Observed sensitivity vs predicted sensitivity values for the SAMSeq post-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-0.8594	0.0008	-1023.00	0.00000
ngenes150000	-0.1794	0.0009	-205.01	0.00000
ngenes50000	0.2565	0.0012	220.78	0.00000
cutoff0.44	-0.0052	0.0011	-4.58	0.00000
cutoff0.89	-0.0307	0.0011	-26.93	0.00000
cutoff1.33	-0.0544	0.0011	-47.68	0.00000
cutoff1.78	-0.0796	0.0011	-69.63	0.00000
cutoff2.22	-0.1073	0.0011	-93.50	0.00000
cutoff2.67	-0.1350	0.0012	-117.32	0.00000
cutoff3.11	-0.1655	0.0012	-143.25	0.00000
cutoff3.56	-0.1979	0.0012	-170.58	0.00000
cutoff4	-0.2326	0.0012	-199.60	0.00000
fold3	0.9562	0.0009	1010.41	0.00000
fold6	1.5904	0.0010	1641.02	0.00000
cutoff0.44:fold3	-0.0282	0.0012	-23.09	0.00000
cutoff0.89:fold3	-0.0962	0.0012	-78.65	0.00000
cutoff1.33:fold3	-0.1572	0.0012	-128.28	0.00000
cutoff1.78:fold3	-0.2152	0.0012	-175.05	0.00000
cutoff2.22:fold3	-0.2670	0.0012	-216.43	0.00000
cutoff2.67:fold3	-0.3142	0.0012	-253.71	0.00000
cutoff3.11:fold3	-0.3559	0.0012	-286.01	0.00000
cutoff3.56:fold3	-0.3932	0.0013	-314.48	0.00000
cutoff4:fold3	-0.4252	0.0013	-338.40	0.00000
cutoff0.44:fold6	-0.1389	0.0012	-111.24	0.00000
cutoff0.89:fold6	-0.3731	0.0012	-300.15	0.00000
cutoff1.33:fold6	-0.5181	0.0012	-416.98	0.00000
cutoff1.78:fold6	-0.6320	0.0012	-507.80	0.00000
cutoff2.22:fold6	-0.7126	0.0012	-571.07	0.00000
cutoff2.67:fold6	-0.7789	0.0013	-622.10	0.00000
cutoff3.11:fold6	-0.8305	0.0013	-660.70	0.00000
cutoff3.56:fold6	-0.8744	0.0013	-692.71	0.00000
cutoff4:fold6	-0.9074	0.0013	-715.70	0.00000
ngenes150000:cutoff0.44	-0.0081	0.0011	-7.30	0.00000
ngenes50000:cutoff0.44	0.0067	0.0015	4.46	0.00001
ngenes150000:cutoff0.89	-0.0141	0.0011	-12.74	0.00000
ngenes50000:cutoff0.89	0.0245	0.0015	16.51	0.00000
ngenes150000:cutoff1.33	-0.0200	0.0011	-18.11	0.00000
ngenes50000:cutoff1.33	0.0303	0.0015	20.49	0.00000
ngenes150000:cutoff1.78	-0.0299	0.0011	-27.04	0.00000
ngenes50000:cutoff1.78	0.0353	0.0015	23.88	0.00000
ngenes150000:cutoff2.22	-0.0423	0.0011	-38.13	0.00000
ngenes50000:cutoff2.22	0.0435	0.0015	29.44	0.00000
ngenes150000:cutoff2.67	-0.0577	0.0011	-51.86	0.00000
ngenes50000:cutoff2.67	0.0529	0.0015	35.81	0.00000
ngenes150000:cutoff3.11	-0.0716	0.0011	-64.04	0.00000
ngenes50000:cutoff3.11	0.0648	0.0015	43.82	0.00000
ngenes150000:cutoff3.56	-0.0831	0.0011	-73.98	0.00000
ngenes50000:cutoff3.56	0.0795	0.0015	53.67	0.00000
ngenes150000:cutoff4	-0.0915	0.0011	-81.00	0.00000
ngenes50000:cutoff4	0.0939	0.0015	63.23	0.00000
ngenes150000:fold3	0.0024	0.0006	3.91	0.00009
ngenes50000:fold3	0.0208	0.0008	25.76	0.00000
ngenes150000:fold6	-0.0047	0.0006	-7.57	0.00000
ngenes50000:fold6	0.0590	0.0008	72.33	0.00000

Table D.22: GLM output from final SAMSeq sensitivity model. Modelled using post-filtered data.

D.2.2 Specificity

	Df	Deviance	AIC
<none>		1265496.16	1644484.49
pout	1	1265496.16	1644482.49
cutoff:fold	18	1266073.89	1645026.21
ngenes:cutoff	18	1572030.08	1950982.40
ngenes:fold	4	1266263.47	1645243.79

Table D.23: Summary of the changes in fit that would result from dropping terms from the EdgeR specificity model. Only removable terms are shown. Modelled using post-filtered data.

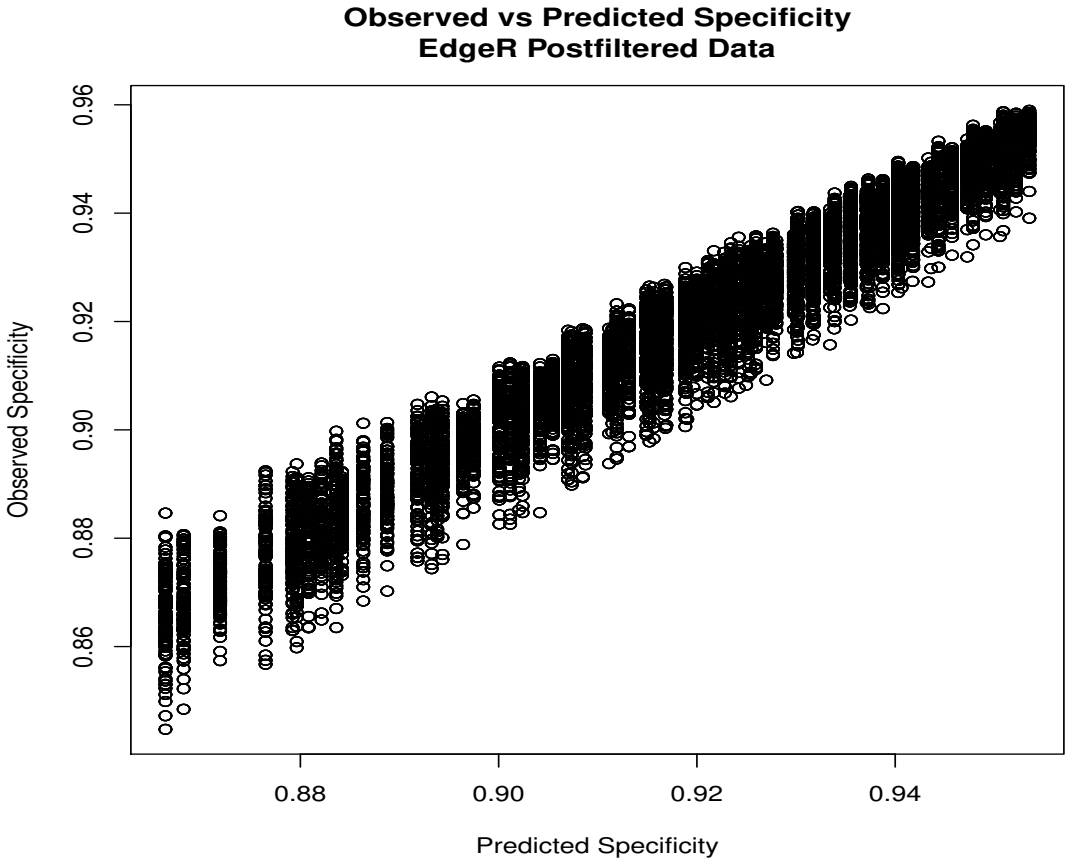

Figure D.12: Specificty vs fitted values for the EdgeR post-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.0128	0.0004	4949.45	0.00000
ngenes150000	0.0653	0.0004	148.30	0.00000
ngenes50000	-0.0949	0.0006	-167.56	0.00000
cutoff0.44	0.1453	0.0006	259.22	0.00000
cutoff0.89	0.3146	0.0006	540.92	0.00000
cutoff1.33	0.4163	0.0006	698.50	0.00000
cutoff1.78	0.4990	0.0006	819.71	0.00000
cutoff2.22	0.5704	0.0006	919.35	0.00000
cutoff2.67	0.6346	0.0006	1004.71	0.00000
cutoff3.11	0.6941	0.0006	1080.51	0.00000
cutoff3.56	0.7502	0.0007	1149.09	0.00000
cutoff4	0.8048	0.0007	1213.18	0.00000
fold3	-0.0278	0.0005	-58.06	0.00000
fold6	-0.0532	0.0005	-111.61	0.00000
cutoff0.44:fold3	-0.0000	0.0006	-0.03	0.97729
cutoff0.89:fold3	-0.0011	0.0006	-1.68	0.09316
cutoff1.33:fold3	-0.0025	0.0007	-3.86	0.00011
cutoff1.78:fold3	-0.0037	0.0007	-5.57	0.00000
cutoff2.22:fold3	-0.0040	0.0007	-5.87	0.00000
cutoff2.67:fold3	-0.0049	0.0007	-7.08	0.00000
cutoff3.11:fold3	-0.0060	0.0007	-8.43	0.00000
cutoff3.56:fold3	-0.0057	0.0007	-7.87	0.00000
cutoff4:fold3	-0.0056	0.0007	-7.66	0.00000
cutoff0.44:fold6	0.0045	0.0006	7.41	0.00000
cutoff0.89:fold6	0.0034	0.0006	5.39	0.00000
cutoff1.33:fold6	-0.0008	0.0007	-1.18	0.23614
cutoff1.78:fold6	-0.0012	0.0007	-1.85	0.06481
cutoff2.22:fold6	-0.0038	0.0007	-5.51	0.00000
cutoff2.67:fold6	-0.0045	0.0007	-6.47	0.00000
cutoff3.11:fold6	-0.0056	0.0007	-7.93	0.00000
cutoff3.56:fold6	-0.0053	0.0007	-7.41	0.00000
cutoff4:fold6	-0.0069	0.0007	-9.48	0.00000
ngenes150000:cutoff0.44	0.0210	0.0006	37.34	0.00000
ngenes50000:cutoff0.44	-0.0303	0.0007	-41.79	0.00000
ngenes150000:cutoff0.89	0.0338	0.0006	57.77	0.00000
ngenes50000:cutoff0.89	-0.0631	0.0007	-84.45	0.00000
ngenes150000:cutoff1.33	0.0470	0.0006	78.50	0.00000
ngenes50000:cutoff1.33	-0.0747	0.0008	-97.89	0.00000
ngenes150000:cutoff1.78	0.0634	0.0006	103.36	0.00000
ngenes50000:cutoff1.78	-0.0852	0.0008	-109.66	0.00000
ngenes150000:cutoff2.22	0.0813	0.0006	129.87	0.00000
ngenes50000:cutoff2.22	-0.0990	0.0008	-125.46	0.00000
ngenes150000:cutoff2.67	0.0994	0.0006	155.82	0.00000
ngenes50000:cutoff2.67	-0.1150	0.0008	-143.76	0.00000
ngenes150000:cutoff3.11	0.1160	0.0007	178.50	0.00000
ngenes50000:cutoff3.11	-0.1312	0.0008	-161.95	0.00000
ngenes150000:cutoff3.56	0.1280	0.0007	193.44	0.00000
ngenes50000:cutoff3.56	-0.1485	0.0008	-181.09	0.00000
ngenes150000:cutoff4	0.1384	0.0007	205.59	0.00000
ngenes50000:cutoff4	-0.1662	0.0008	-200.29	0.00000
ngenes150000:fold3	0.0036	0.0004	9.84	0.00000
ngenes50000:fold3	-0.0048	0.0005	-10.41	0.00000
ngenes150000:fold6	0.0023	0.0004	6.34	0.00000
ngenes50000:fold6	0.0047	0.0005	10.36	0.00000

Table D.24: GLM output from final EdgeR specificity model. Modelled using post-filtered data.

	Df	Deviance	AIC
<none>		1249705.25	1626935.13
pout	1	1249705.25	1626933.13
cutoff:fold	18	1250387.80	1627581.69
ngenes:cutoff	18	1550089.68	1927283.57
ngenes:fold	4	1250394.01	1627615.90

Table D.25: Summary of the changes in fit that would result from dropping terms from the Robust EdgeR specificity model. Only removable terms are shown. Modelled using post-filtered data.

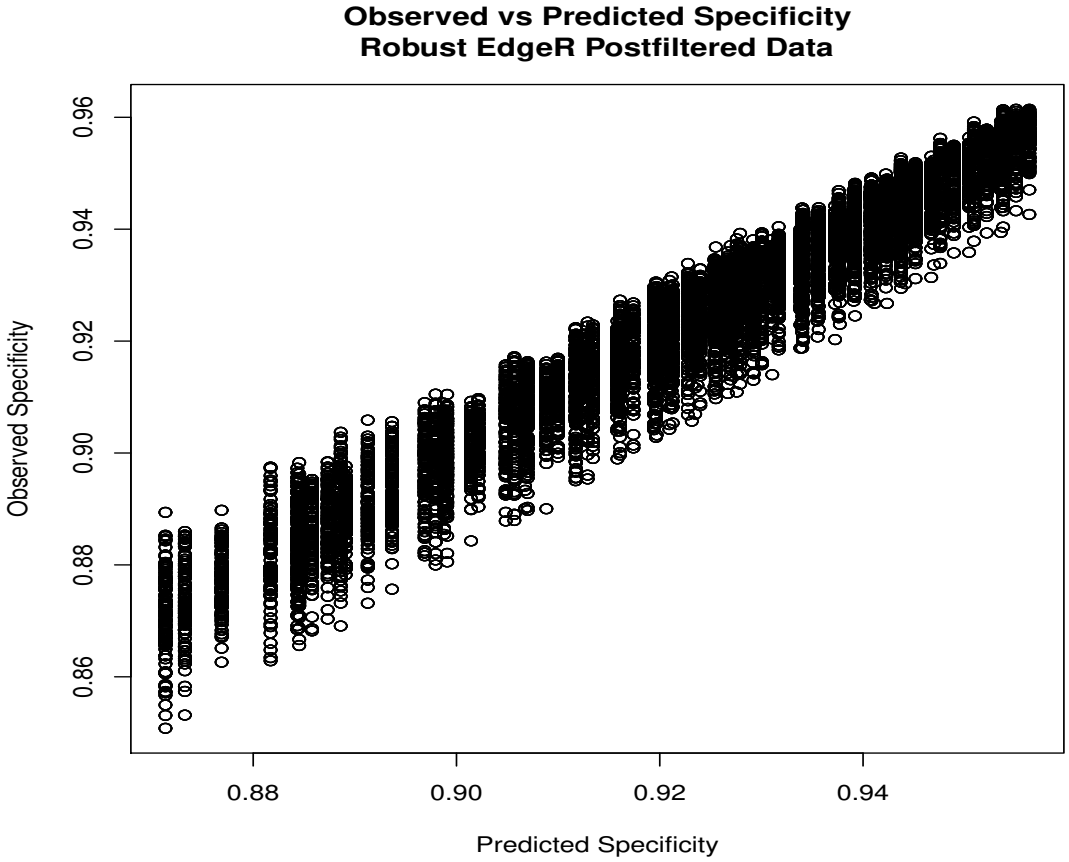

Figure D.13: Observed specificity vs predicted specificity values for the Robust EdgeR post-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.0637	0.0004	4974.55	0.00000
ngenes150000	0.0649	0.0004	144.37	0.00000
ngenes50000	-0.1004	0.0006	-174.00	0.00000
cutoff0.44	0.1495	0.0006	261.46	0.00000
cutoff0.89	0.3233	0.0006	543.72	0.00000
cutoff1.33	0.4269	0.0006	700.13	0.00000
cutoff1.78	0.5109	0.0006	819.73	0.00000
cutoff2.22	0.5832	0.0006	917.63	0.00000
cutoff2.67	0.6480	0.0006	1001.17	0.00000
cutoff3.11	0.7078	0.0007	1074.95	0.00000
cutoff3.56	0.7640	0.0007	1141.34	0.00000
cutoff4	0.8186	0.0007	1203.30	0.00000
fold3	-0.0295	0.0005	-60.36	0.00000
fold6	-0.0550	0.0005	-113.09	0.00000
cutoff0.44:fold3	-0.0001	0.0006	-0.13	0.89961
cutoff0.89:fold3	-0.0013	0.0007	-2.01	0.04428
cutoff1.33:fold3	-0.0028	0.0007	-4.18	0.00003
cutoff1.78:fold3	-0.0040	0.0007	-5.90	0.00000
cutoff2.22:fold3	-0.0044	0.0007	-6.28	0.00000
cutoff2.67:fold3	-0.0055	0.0007	-7.70	0.00000
cutoff3.11:fold3	-0.0067	0.0007	-9.18	0.00000
cutoff3.56:fold3	-0.0062	0.0007	-8.37	0.00000
cutoff4:fold3	-0.0060	0.0008	-8.05	0.00000
cutoff0.44:fold6	0.0047	0.0006	7.48	0.00000
cutoff0.89:fold6	0.0033	0.0007	5.11	0.00000
cutoff1.33:fold6	-0.0011	0.0007	-1.61	0.10645
cutoff1.78:fold6	-0.0017	0.0007	-2.48	0.01304
cutoff2.22:fold6	-0.0045	0.0007	-6.42	0.00000
cutoff2.67:fold6	-0.0054	0.0007	-7.65	0.00000
cutoff3.11:fold6	-0.0067	0.0007	-9.31	0.00000
cutoff3.56:fold6	-0.0064	0.0007	-8.72	0.00000
cutoff4:fold6	-0.0080	0.0007	-10.77	0.00000
ngenes150000:cutoff0.44	0.0214	0.0006	37.23	0.00000
ngenes50000:cutoff0.44	-0.0313	0.0007	-42.41	0.00000
ngenes150000:cutoff0.89	0.0341	0.0006	57.00	0.00000
ngenes50000:cutoff0.89	-0.0651	0.0008	-85.37	0.00000
ngenes150000:cutoff1.33	0.0476	0.0006	77.59	0.00000
ngenes50000:cutoff1.33	-0.0768	0.0008	-98.59	0.00000
ngenes150000:cutoff1.78	0.0642	0.0006	102.37	0.00000
ngenes50000:cutoff1.78	-0.0875	0.0008	-110.20	0.00000
ngenes150000:cutoff2.22	0.0823	0.0006	128.35	0.00000
ngenes50000:cutoff2.22	-0.1015	0.0008	-125.83	0.00000
ngenes150000:cutoff2.67	0.1005	0.0007	153.68	0.00000
ngenes50000:cutoff2.67	-0.1175	0.0008	-143.70	0.00000
ngenes150000:cutoff3.11	0.1172	0.0007	175.83	0.00000
ngenes50000:cutoff3.11	-0.1337	0.0008	-161.35	0.00000
ngenes150000:cutoff3.56	0.1293	0.0007	190.59	0.00000
ngenes50000:cutoff3.56	-0.1510	0.0008	-180.05	0.00000
ngenes150000:cutoff4	0.1398	0.0007	202.44	0.00000
ngenes50000:cutoff4	-0.1687	0.0008	-198.78	0.00000
ngenes150000:fold3	0.0046	0.0004	12.38	0.00000
ngenes50000:fold3	-0.0036	0.0005	-7.65	0.00000
ngenes150000:fold6	0.0030	0.0004	8.11	0.00000
ngenes50000:fold6	0.0048	0.0005	10.31	0.00000

Table D.26: GLM output from final Robust EdgeR specificity model. Modelled using post-filtered data.

	Df	Deviance	AIC
<none>		306276.06	686635.01
pout	1	306276.06	686633.01
cutoff:fold	18	306774.41	687097.36
ngenes:cutoff	18	540460.16	920783.10
ngenes:fold	4	307526.49	687877.44

Table D.27: Summary of the changes in fit that would result from dropping terms from the DESeq2 specificity model. Only removable terms are shown. Modelled using post-filtered data.

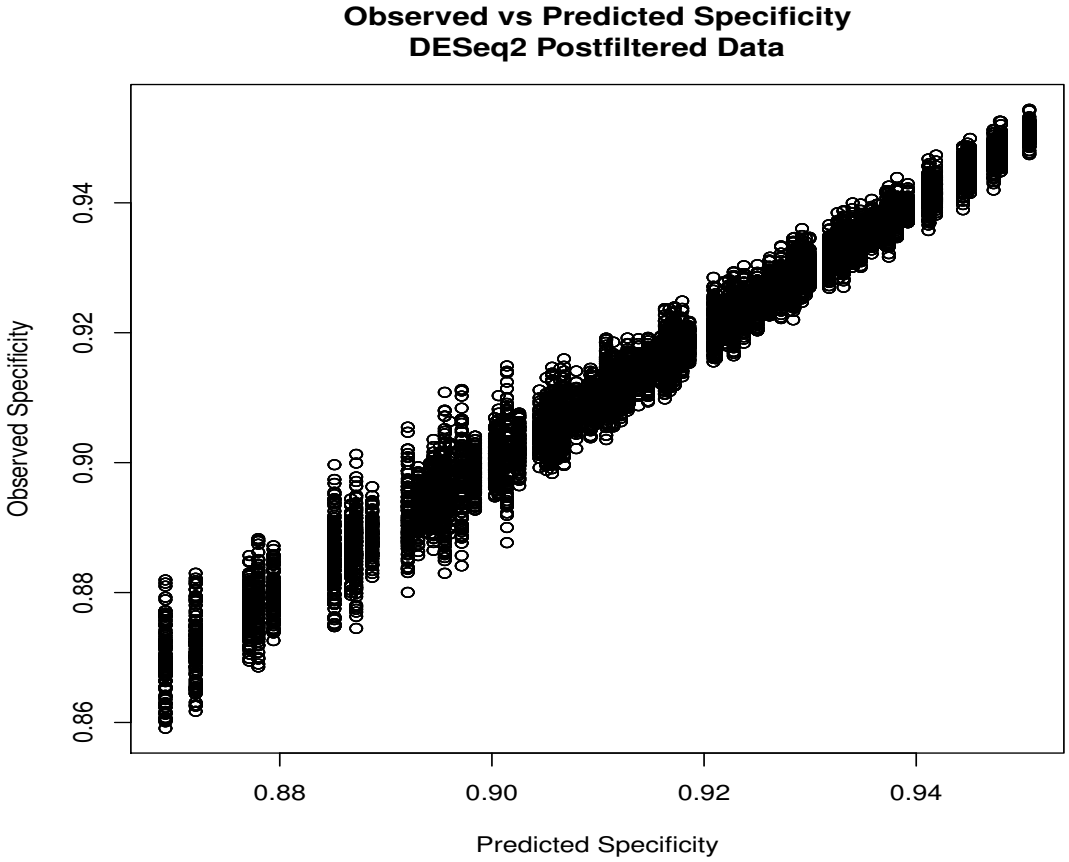

Figure D.14: Observed specificity vs predicted specificity values for the DESeq2 post-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.1120	0.0004	5023.35	0.00000
ngenes150000	0.1010	0.0005	222.48	0.00000
ngenes50000	-0.1386	0.0006	-241.50	0.00000
cutoff0.44	0.0922	0.0006	160.14	0.00000
cutoff0.89	0.2106	0.0006	356.57	0.00000
cutoff1.33	0.2894	0.0006	481.06	0.00000
cutoff1.78	0.3572	0.0006	584.11	0.00000
cutoff2.22	0.4179	0.0006	673.03	0.00000
cutoff2.67	0.4736	0.0006	751.71	0.00000
cutoff3.11	0.5261	0.0006	823.49	0.00000
cutoff3.56	0.5763	0.0006	889.83	0.00000
cutoff4	0.6256	0.0007	953.06	0.00000
fold3	-0.0495	0.0005	-100.63	0.00000
fold6	-0.0699	0.0005	-142.54	0.00000
cutoff0.44:fold3	-0.0009	0.0006	-1.44	0.14909
cutoff0.89:fold3	-0.0022	0.0006	-3.35	0.00082
cutoff1.33:fold3	-0.0038	0.0007	-5.75	0.00000
cutoff1.78:fold3	-0.0050	0.0007	-7.45	0.00000
cutoff2.22:fold3	-0.0057	0.0007	-8.35	0.00000
cutoff2.67:fold3	-0.0070	0.0007	-10.05	0.00000
cutoff3.11:fold3	-0.0082	0.0007	-11.70	0.00000
cutoff3.56:fold3	-0.0083	0.0007	-11.69	0.00000
cutoff4:fold3	-0.0086	0.0007	-11.95	0.00000
cutoff0.44:fold6	0.0031	0.0006	4.94	0.00000
cutoff0.89:fold6	0.0030	0.0006	4.59	0.00000
cutoff1.33:fold6	-0.0001	0.0007	-0.17	0.86634
cutoff1.78:fold6	-0.0003	0.0007	-0.49	0.62351
cutoff2.22:fold6	-0.0023	0.0007	-3.45	0.00057
cutoff2.67:fold6	-0.0031	0.0007	-4.43	0.00001
cutoff3.11:fold6	-0.0042	0.0007	-5.92	0.00000
cutoff3.56:fold6	-0.0039	0.0007	-5.46	0.00000
cutoff4:fold6	-0.0053	0.0007	-7.35	0.00000
ngenes150000:cutoff0.44	0.0179	0.0006	30.88	0.00000
ngenes50000:cutoff0.44	-0.0239	0.0007	-32.56	0.00000
ngenes150000:cutoff0.89	0.0308	0.0006	51.82	0.00000
ngenes50000:cutoff0.89	-0.0501	0.0007	-67.01	0.00000
ngenes150000:cutoff1.33	0.0429	0.0006	70.90	0.00000
ngenes50000:cutoff1.33	-0.0611	0.0008	-80.41	0.00000
ngenes150000:cutoff1.78	0.0569	0.0006	92.26	0.00000
ngenes50000:cutoff1.78	-0.0712	0.0008	-92.52	0.00000
ngenes150000:cutoff2.22	0.0718	0.0006	114.60	0.00000
ngenes50000:cutoff2.22	-0.0835	0.0008	-107.10	0.00000
ngenes150000:cutoff2.67	0.0873	0.0006	137.18	0.00000
ngenes50000:cutoff2.67	-0.0971	0.0008	-123.24	0.00000
ngenes150000:cutoff3.11	0.1015	0.0006	156.94	0.00000
ngenes50000:cutoff3.11	-0.1110	0.0008	-139.32	0.00000
ngenes150000:cutoff3.56	0.1115	0.0007	169.86	0.00000
ngenes50000:cutoff3.56	-0.1261	0.0008	-156.73	0.00000
ngenes150000:cutoff4	0.1199	0.0007	180.06	0.00000
ngenes50000:cutoff4	-0.1418	0.0008	-174.41	0.00000
ngenes150000:fold3	0.0024	0.0004	6.67	0.00000
ngenes50000:fold3	-0.0045	0.0004	-10.04	0.00000
ngenes150000:fold6	0.0056	0.0004	15.38	0.00000
ngenes50000:fold6	-0.0094	0.0004	-21.13	0.00000

Table D.28: GLM output from final DESeq2 specificity model. Modelled using post-filtered data.

	Df	Deviance	AIC
<none>		207924.15	496230.33
pout	1	207924.16	496228.34
cutoff:fold	18	209434.39	497704.56
ngenes:cutoff	18	225810.35	514080.53
ngenes:fold	4	218251.85	506550.03

Table D.29: Summary of the changes in fit that would result from dropping terms from the SAMSeq specificity model. Only removable terms are shown. Modelled using post-filtered data.

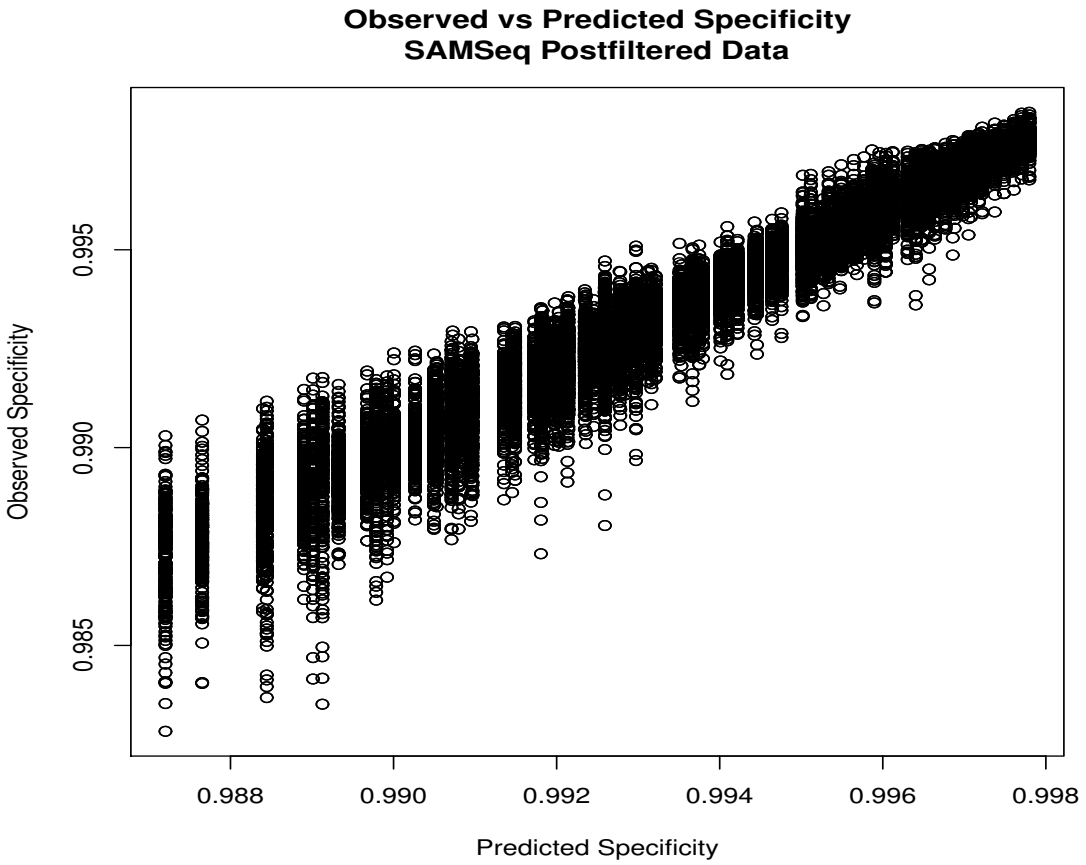

Figure D.15: Observed specificity vs predicted specificity values for the SAMSeq post-filtered data. Fit using R package `glm`, `family=binomial`, with a logistic link function.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	5.4918	0.0019	2913.53	0.00000
ngenes150000	0.1325	0.0018	73.20	0.00000
ngenes50000	-0.1948	0.0022	-87.26	0.00000
cutoff0.44	0.0347	0.0025	13.79	0.00000
cutoff0.89	0.1051	0.0026	41.01	0.00000
cutoff1.33	0.1534	0.0026	59.03	0.00000
cutoff1.78	0.2000	0.0026	75.95	0.00000
cutoff2.22	0.2399	0.0027	89.99	0.00000
cutoff2.67	0.2752	0.0027	102.18	0.00000
cutoff3.11	0.3080	0.0027	113.15	0.00000
cutoff3.56	0.3421	0.0028	124.41	0.00000
cutoff4	0.3761	0.0028	135.35	0.00000
fold3	-0.6955	0.0021	-334.40	0.00000
fold6	-1.0428	0.0020	-528.01	0.00000
cutoff0.44:fold3	0.0062	0.0027	2.31	0.02098
cutoff0.89:fold3	0.0224	0.0027	8.21	0.00000
cutoff1.33:fold3	0.0288	0.0028	10.42	0.00000
cutoff1.78:fold3	0.0342	0.0028	12.20	0.00000
cutoff2.22:fold3	0.0326	0.0028	11.48	0.00000
cutoff2.67:fold3	0.0401	0.0029	13.96	0.00000
cutoff3.11:fold3	0.0429	0.0029	14.77	0.00000
cutoff3.56:fold3	0.0458	0.0029	15.61	0.00000
cutoff4:fold3	0.0517	0.0030	17.44	0.00000
cutoff0.44:fold6	0.0162	0.0025	6.40	0.00000
cutoff0.89:fold6	0.0331	0.0026	12.79	0.00000
cutoff1.33:fold6	0.0440	0.0026	16.78	0.00000
cutoff1.78:fold6	0.0488	0.0027	18.35	0.00000
cutoff2.22:fold6	0.0531	0.0027	19.69	0.00000
cutoff2.67:fold6	0.0621	0.0027	22.80	0.00000
cutoff3.11:fold6	0.0665	0.0028	24.14	0.00000
cutoff3.56:fold6	0.0737	0.0028	26.45	0.00000
cutoff4:fold6	0.0741	0.0028	26.32	0.00000
ngenes150000:cutoff0.44	0.0124	0.0020	6.07	0.00000
ngenes50000:cutoff0.44	-0.0146	0.0026	-5.68	0.00000
ngenes150000:cutoff0.89	0.0303	0.0021	14.51	0.00000
ngenes50000:cutoff0.89	-0.0389	0.0026	-14.88	0.00000
ngenes150000:cutoff1.33	0.0458	0.0021	21.56	0.00000
ngenes50000:cutoff1.33	-0.0532	0.0026	-20.10	0.00000
ngenes150000:cutoff1.78	0.0578	0.0022	26.81	0.00000
ngenes50000:cutoff1.78	-0.0652	0.0027	-24.34	0.00000
ngenes150000:cutoff2.22	0.0755	0.0022	34.51	0.00000
ngenes50000:cutoff2.22	-0.0757	0.0027	-27.97	0.00000
ngenes150000:cutoff2.67	0.0839	0.0022	37.88	0.00000
ngenes50000:cutoff2.67	-0.0874	0.0027	-32.01	0.00000
ngenes150000:cutoff3.11	0.1011	0.0022	45.05	0.00000
ngenes50000:cutoff3.11	-0.0980	0.0028	-35.59	0.00000
ngenes150000:cutoff3.56	0.1078	0.0023	47.45	0.00000
ngenes50000:cutoff3.56	-0.1115	0.0028	-40.13	0.00000
ngenes150000:cutoff4	0.1155	0.0023	50.23	0.00000
ngenes50000:cutoff4	-0.1246	0.0028	-44.51	0.00000
ngenes150000:fold3	-0.0310	0.0015	-20.68	0.00000
ngenes50000:fold3	0.0681	0.0018	37.84	0.00000
ngenes150000:fold6	-0.0709	0.0014	-49.74	0.00000
ngenes50000:fold6	0.0914	0.0017	53.37	0.00000

Table D.30: GLM output from final SAMSeq specificity model. Modelled using post-filtered data.

Bibliography

[1] Simon Anders and Wolfgang Huber. Differential expression analysis for sequence count data. *Genome Biol*, 11(10):R106, 2010.

[2] Simon Anders, Davis J McCarthy, Yunshun Chen, Michal Okoniewski, Gordon K Smyth, Wolfgang Huber, and Mark D Robinson. Count-based differential expression analysis of rna sequencing data using r and bioconductor. *Nature Protocols*, 8(9):1765–1786, 2013.

[3] Paul L Auer and Rebecca W Doerge. A two-stage poisson model for testing rna-seq data. *Statistical Applications in Genetics and Molecular Biology*, 10(1):1–26, 2011.

[4] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 289–300, 1995.

[5] Richard Bourgon, Robert Gentleman, and Wolfgang Huber. Independent filtering increases detection power for high-throughput experiments. *Proceedings of the National Academy of Sciences*, 107(21):9546–9551, 2010.

[6] Yunshun Chen, Aaron TL Lun, and Gordon K Smyth. Differential expression analysis of complex rna-seq experiments using edger. In *Statistical Analysis of Next Generation Sequencing Data*, pages 51–74. Springer, 2014.

[7] William S Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836, 1979.

[8] David Roxbee Cox and Nancy Reid. Parameter orthogonality and approximate conditional inference. *Journal of the Royal Statistical Society. Series B (Methodological)*, 49(1):1–39, 1987.

[9] Yanming Di, Daniel W Schafer, Jason S Cumbie, and Jeff H Chang. The nbp negative binomial model for assessing differential gene expression from rna-seq. *Statistical Applications in Genetics and Molecular Biology*, 10(1), 2011.

[10] Marie-Agnès Dillies, Andrea Rau, Julie Aubert, Christelle Hennequet-Antier, Marine Jeanmougin, Nicolas Servant, Céline Keime, Guillemette Marot, David Castel, Jordi Estelle, et al. A comprehensive evaluation of normalization methods for illumina high-throughput rna sequencing data analysis. *Briefings in Bioinformatics*, 14(6):671–683, 2013.

[11] Annette J Dobson and Adrian Barnett. *An Introduction to Generalized Linear Models*. CRC press, 2008.

[12] Sorin Draghici. *Statistics and Data Analysis for Microarrays using R and Bioconductor*. Chapman and Hall/CRC, US, 2011.

[13] Parnell LD et al. Biostar: An online question and answer resource for the bioinformatics community. https://support.bioconductor.org/p/60205/, 2011.

[14] Parnell LD et al. Biostar: An online question and answer resource for the bioinformatics community. https://support.bioconductor.org/p/59299/, 2011.

[15] John Fox. *An R and S-Plus Companion to Applied Regression*. Sage, 2002.

[16] R. Gentleman, V. Carey, W. Huber, and F. Hahne. *genefilter: genefilter: methods for filtering genes from high-throughput experiments*, 2016. R package version 1.54.2.

[17] Thomas J. Hardcastle. *BaySeq: Empirical Bayesian Analysis of Patterns of Differential Expression in Count Data*, 2012. R package version 2.6.0.

[18] Thomas J Hardcastle and Krystyna A Kelly. bayseq: Empirical bayesian methods for identifying differential expression in ssequence count data. *BMC Bioinformatics*, 11(1):422, 2010.

[19] Sangjin Kim and Paul Schliekelman. Prioritizing hypothesis tests for high throughput data. *Bioinformatics*, 32(6):850–858, 2015.

[20] Eija Korpelainen, Jarno Tuimala, Panu Somervuo, Mikael Huss, and Garry Wong. *RNA-seq Data Analysis: A Practical Approach*. CRC Press, 2014.

[21] Ning Leng, John A Dawson, James A Thomson, Victor Ruotti, Anna I Rissman, Bart MG Smits, Jill D Haag, Michael N Gould, Ron M Stewart, and Christina Kendziorski. Ebseq: an empirical bayes hierarchical model for inference in rna-seq experiments. *Bioinformatics*, 29(8):1035–1043, 2013.

[22] Jun Li. Using npseq (version 1.1) to discover differential expression based on sequencing data. 2011.

[23] Jun Li and Robert Tibshirani. Finding consistent patterns: A nonparametric approach for identifying differential expression in rna-seq data. *Statistical Methods in Medical Research*, 22(5):519–536, 2013.

[24] Jun Li, Daniela M Witten, Iain M Johnstone, and Robert Tibshirani. Normalization, testing, and false discovery rate estimation for rna-sequencing data. *Biostatistics*, 13(3):523–538, 2012.

[25] Yang Liao, Gordon K Smyth, and Wei Shi. featurecounts: an efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics*, 30(7):923–930, 2014.

[26] Michael I Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for rna-seq data with deseq2. *Genome Biology*, 15(12):550, 2014.

[27] Michael I Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for rna-seq data with deseq2. *Genome Biology*, 15(12):1, 2014.

[28] Jun Lu, John K Tomfohr, and Thomas B Kepler. Identifying differential expression in multiple sage libraries: an overdispersed log-linear model approach. *BMC Bioinformatics*, 6(1):165, 2005.

[29] John C Marioni, Christopher E Mason, Shrikant M Mane, Matthew Stephens, and Yoav Gilad. Rna-seq: an assessment of technical reproducibility and comparison with gene expression arrays. *Genome Research*, 18(9):1509–1517, 2008.

[30] Peter McCullagh and John A Nelder. *Generalized Linear Models*, volume 37. CRC press, 1989.

[31] Stephen B Montgomery, Micha Sammeth, Maria Gutierrez-Arcelus, Radoslaw P Lach, Catherine Ingle, James Nisbett, Roderic Guigo, and Emmanouil T Dermitzakis. Transcriptome genetics using second generation sequencing in a caucasian population. *Nature*, 464(7289), 2010.

[32] Lior Pachter. Models for transcript quantification from rna-seq. *arXiv preprint arXiv:1104.3889*, 2011.

[33] Joseph K Pickrell, John C Marioni, Athma A Pai, Jacob F Degner, Barbara E Engelhardt, Everlyne Nkadori, Jean-Baptiste Veyrieras, Matthew Stephens, Yoav Gilad, and Jonathan K Pritchard. Understanding mechanisms underlying human gene expression variation with rna sequencing. *Nature*, 464(7289):768–772, 2010.

[34] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016.

[35] Franck Rapaport, Raya Khanin, Yupu Liang, Mono Pirun, Azra Krek, Paul Zumbo, Christopher E Mason, Nicholas D Socci, and Doron Betel. Comprehensive evaluation of differential gene expression analysis methods for rna-seq data. *Genome Biol*, 14(9):R95, 2013.

[36] Andrea Rau, Melina Gallopin, Gilles Celeux, and Florence Jaffrezic. Data-based filtering for replicated high-throughput transcriptome sequencing experiments. *Bioinformatics*, 29(17):2146–2152, 2013.

[37] Matthew E Ritchie, Belinda Phipson, Di Wu, Yifang Hu, Charity W Law, Wei Shi, and Gordon K Smyth. limma powers differential expression analyses for rna-sequencing and microarray studies. *Nucleic Acids Research*, 43(7):e47–e47, 2015.

[38] Mark D Robinson. Robustly detecting differential expression in rna sequencing data using observation weights: Supplementary data, r code for simulations and analyses. http://130.60.190.4/robinson_lab/edgeR_robust/robust_simulation.R.

[39] Mark D Robinson, Davis J McCarthy, and Gordon K Smyth. edger: a bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1):139–140, 2010.

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

[41] José A Robles, Sumaira E Qureshi, Stuart J Stephen, Susan R Wilson, Conrad J Burden, and Jennifer M Taylor. Efficient experimental design and analysis strategies for the detection of differential expression using rna-sequencing. *BMC Genomics*, 13(1):484, 2012.

[42] Gordon K Smyth. Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, 3(1):1–25, 2004.

[43] Charlotte Soneson and Mauro Delorenzi. A comparison of methods for differential expression analysis of rna-seq data. *BMC Bioinformatics*, 14(1):91, 2013.

[44] Sonia Tarazona, Fernando Garcia-Alcalde, Joaquin Dopazo, Alberto Ferrer, and Ana Conesa. Differential expression in rna-seq: a matter of depth. *Genome Research*, 21(12):2213–2223, 2011.

[45] R. Tibshirani, G. Chu, Balasubramanian Narasimhan, and Jun Li. *samr: SAM: Significance Analysis of Microarrays*, 2011. R package version 2.0.

[46] Mark A Van De Wiel, Gwenaël GR Leday, Luba Pardo, Håvard Rue, Aad W Van Der Vaart, and Wessel N Van Wieringen. Bayesian analysis of rna sequencing data by estimating multiple shrinkage priors. *Biostatistics*, 14(1):113–128, 2013.

[47] Xiaobei Zhou, Helen Lindsay, and Mark D Robinson. Robustly detecting differential expression in rna sequencing data using observation weights. *Nucleic Acids Research*, 42(11):e91–e91, 2014.