Métodos Estatísticos em Bioinformática

Master's Degree in Bioinformatics and Computational Biology

June 2021

Miguel Casanova Vieira Parente

Nº 24475

Evaluation Project – Question II

RNA-sequencing analysis

2nd Semester – 2020/2021

Table of Contents

1. Introduction

Since its inception, RNA-sequencing (RNA-seq) has established itself as the dominant technology to obtain gene expression data[1–3]. The data it generates, can be used to identify differentially expressed genes (DEGs) or transcripts between different experimental groups or conditions, allowing for a myriad of applications in both fundamental and applied research. Whereas technologically, the platforms of high throughput sequencing seem to be dominated by Illumina, tools for RNA-seq analysis abound and tackle this process in multiple angles. In this never-ending battle to determine which tool is the best, many publications evaluating the robustness of these tools have been reported (see, for example, [4–6]). Unsurprisingly, no clear winner stands out and their suitability depends on the dataset and biological questions at hand.

The standard process for differential expression (DE) analysis typically starts with a "count matrix", a table in which each row indicates a gene (or an exon, or a genomic locus of interest), each column corresponds to a different sample, and each cell indicates the number of reads mapped to the respective genomic feature[3]. There are several levels of resolution for current DE analysis, including gene-, transcript-, exon-, and base-level. Here, I will focus primarily in tools that perform DE analysis at the gene-level.

DE analysis can be roughly divided in two main steps: data normalization and DEG identification. The aim of the normalization step is to transform the raw counts in such a way that non-DEGs will have similar normalized counts across all samples. These counts will then be used for DE analysis, by estimating the parameters that define the model of gene count distribution, in order to calculate p-values to test whether or not a given gene is differentially expressed. Following this general approach, different tools make assumptions about the statistical properties inherent to RNA-seq data, using a combination of normalization and analysis strategies to calculate the magnitude of a DEG result and its corresponding significance. In other words, DE tools use statistical testing to decide whether, for a given gene, an observed difference in read counts is significant and greater than what would be expected due to normal random variation of the data.

In this report, I will focus on different strategies employed by a set of DE analysis tools, for the normalization and DEG identification steps. In this sense, and with the consent of the professor, I focused my attention on the use of five different tools for read count normalization and/or differential expression analysis contained in Bioconductor[7]: edgeR[8], limma[9,10], EDASeq[11], DESeq2[12] and baySeq[13]. These tools were employed to study a private dataset (obtained from Simão Teixeira da Rocha Group, iMM) consisting of RNA-seq data of control and Angelman derived induced pluripotent stem cells (iPSCs). Angelman is a genetic neurodevelopmental disorder, caused by the loss of the maternally inherited copy of the *UBE3A* gene[14]. This incurable genetic disease is characterized by severe developmental delays, speech impairment, uncontrolled laughter and muscular ataxia. Our current understanding of the disease, is hindered by the lack of relevant human derived data, and the fact that it is an orphan disease, with little leverage on garnering support and resources for exploring its etiology and treatments.

2. Materials and Methods

2.1. Datasets

Total RNA from 3 control (TCLAB) and 3 Angelman (ASD) derived iPSCs was extracted and ribodepleted. Strand-specific, poly-A libraries were prepared and sequenced on an Illumina NovaSeq 6000, using paired-end protocols with a 150bp sequencing length (Illumina). Raw sequences were quality controlled using FastQC[15] (v.0.11.5), and sequencing adapters removed using trim_galore[16] (v.0.6.6). Paired-end alignments were generated using STAR[17] (v.2.7.5a) and the current human reference genome (hg38/GRCh38). Uniquely-mapped reads covering exons of each annotated gene were counted using featureCounts[18] (v.1.6.0) and the comprehensive gene annotation (GTF file) from Gencode (GRCh38.13).

2.2. Normalization and differential expression analysis

All differential expression analysis were performed using the same gene count matrix output from featureCounts. This matrix was processed and only genes for which GC-content and length information could be extracted, using biomaRt (v.2.48.0), were kept. In addition, genes mapping to haplotypes or unfinished scaffolds were discarded. Finally, genes were filtered to eliminate those with an average read count less than 10 across the 6 samples. The raw count matrix is provided as an annex to this report.

For all of the experimental strategies, the same design matrix, considering the difference between control and Angelman (as factors), was used[19].

2.2.1. EDASeq[11] (v.2.26.0)

Normalization for within and between samples was performed using EDASeq, following the package vignette. Normalization within samples was performed for GC-content bias in gene count and using full-quantile normalization between feature strata: *withinLaneNormalization(data,"gc", which = "full", offset = TRUE)*. Normalization for sequencing depth between samples was accomplished using full-quantile normalization: *betweenLaneNormalization(data, which = "full", offset = TRUE)*. In order to keep the count data unchanged and better preserve their inherent sampling properties, the count data was left unchanged and an offset was added to the countData object.

2.2.2. edgeR[8] (v.3.34.0)

Normalization factors for the different samples were calculated using trimmed mean values (TMM): *calcNormFactors(edgeR_DGElist, method = "TMM")*. This normalization uses a weighted trimmed mean of the log expression ratios between samples, allowing to normalize for sequencing depth, RNA composition and gene length.

For differential expression analysis, following EDASeq normalization, the gene count dispersion was first calculated, using *estimateDisp(EDASeqNorm_edgeR_DGElist, design)*. Next, a negative binomial generalized log-linear model was fit to the observed dispersion, following the coefficient provided in the design matrix:

glmFit(EDASeqNorm_edgeR_DGElist, design). Finally, genewise statistical tests for the coefficient defined in the design matrix were performed using *glmLRT(edgeR_fit)*. Differentially expressed genes were obtained using: *topTags(edgeR_LRT, n = Inf, sort.by = "P", adjust.method = "BH")*.

2.2.3. limma[9,10] (v.3.48.0)

limma normalization was performed as suggested in the package vignette, using the TMM method from edgeR (in which limma is now included). This method is advised when performing normalization of RNA-seq data, as opposed to the quantile normalization first proposed in the context of microarray data. Although limma normalization was not used, this could have been performed on raw read counts, with: *voom(countsFiltered, design, plot = TRUE, normalize.method = "quantile")*.

For DE analysis, using limma, data normalized with the edgeR package was used. Briefly, data was transformed for linear modelling, using *voom(edgeR_DGElist, design, plot = TRUE)*. This transforms counts into normalized counts per million (CPM), using the normalization factors calculated by the edgeR TMM normalization method. It then fits a linear model to the log2 CPM of each gene, calculating the residuals. This allows fitting a smoothed curve to the residual standard deviation by average expression. This curve will be finally used to obtain weights for each gene and sample, that are passed on to limma along with the log2 CPM values. Next, I use limma to fit a linear model using weighted least squares for each gene, considering the coefficient defined in the design matrix: *lmFit(voomTransformed, design = design*). Finally, empirical Bayes smoothing of standard errors is applied, allowing to compute moderated t-statistics, moderated F-statistics and log-odds of differential expression (*eBayes(voom_fit)*). DEGs are obtained with *topTable(voom_fit, coef = "sample_info$phenotypeAngelman", number = Inf, adjust.method = "BH", sort.by = "P")*.

2.2.4. DESeq2[12] (v.1.32.0)

DESeq2 normalization is achieved by calculating a median of ratios of gene counts relative to the geometric mean per gene. The counts for a gene in each sample is then divided by this mean. The median of these ratios in each sample is the estimated size factor for the respective sample. This will correct for sequencing depth and RNA composition bias, when only a small number of genes are very highly expressed in one condition but not another, for example. This normalization is achieved using: *estimateSizeFactors(dds)*.

For differential expression analysis, DESeq2 assumes that gene counts follow a Negative Binomial distribution, first calculating the dispersion estimates for gene counts for the above model. Then using the coefficient determined in the design matrix, it will apply negative binomial generalized linear models to test the significance of the coefficient for each gene, using Wald test. All of the above is accomplished with: *DESeq(dds)*. DEGs are then obtained using: *results(dds, independentFiltering = TRUE)*.

2.2.5. baySeq[13] (v.2.26.0)

Similarly to edgeR and DESeq2, baySeq assumes a negative binomial distribution for gene counts. Sequencing depth correction was performed by adding a size factor to the countData object, similarly to the TMM method employed by edgeR: *getLibsizes(CD, estimationType= "edgeR")*.

For differential expression analysis, we first estimate an empirical distribution on the parameters of the negative binomial distribution, by bootstrapping from the data, taking individual counts and finding the quasi-likelihood parameters for the negative binomial distribution. For efficient estimation of the prior distribution, a large sample size (10e5) was used: *getPriors(CD, samplesize = 10^5, cl = cl, verbose = T)*. We next acquire the posterior likelihoods, estimating the proportions of differentially expressed genes: *getLikelihoods(CD, pET = 'BIC', cl = cl, verbose = T)*. DEGs are obtained with: *topCounts(CD, group = "DE", number = Inf, normaliseData = TRUE)*.

2.3. R and Rstudio

All the analysis was performed in R studio (v.1.4.1717), using R (v. 4.1.0). Besides the Bioconductor packages for RNA-seq analysis, the following packages were used: *pheatmap* (v.1.0.12), *reshape2* (v1.4.4), *ggplot2* (v.3.3.3), *RColorBrewer* (v.1.1-2), *magrittr* (v.2.0.1), *VSN* (v.3.60.0), *NMF* (v.0.23.0), *org.Hs.eg.db* (v.3.13.0), *grDevices* (v.4.1.0), *ggrepel* (v.0.9.1), *data.table* (v.1.14.0), *biomaRt* (v.2.48.0), *plyr* (v.1.8.6), *ggvenn* (v.0.1.8) and *UpSetR* (v.1.4.0). The code for all of the results contained within this report are provided as a commented R script, annexed to this document.

3. Results

To investigate how different DE pipelines behave when dealing with RNA-seq data, a total of 4 pipelines, using a combination of five different Bioconductor packages, were evaluated for this report: EDASeq, edgeR, limma, DESeq2 and baySeq. The four pipelines consisted of: EDASeq normalization, followed by edgeR DE (EE); edgeR normalization, followed by limma DE (ErL); DESeq2 normalization and DE analysis (DD); and as a “bonus” method, baySeq normalization and DE analysis (BB).

Six datasets, corresponding to 3 control (TCLAB) and 3 Angelman syndrome (ASD) iPSCs, were used and a corresponding counts matrix for exons spanning all annotated genes was used for all downstream analysis. Genes without GC-content or length information were discarded, as well as genes with less than 10 average reads across the six samples. From the original 59609 genes (including lncRNAs, miRNAs, snoRNAs, etc.), we end up with 15639 genes. This filtered counts matrix was used as the raw input data for all of the pipelines tested in this report.

3.1. Raw Samples

During the sample preparation or sequencing process, external factors that are not of biological interest can affect the expression profile of individual samples. This can lead to dissimilar expression distributions per sample, which have to be tackled by normalization. We started by analyzing the per sample read count distribution (**Fig.1a**). In addition, the density of read counts per gene was estimated for each sample (**Fig. 1b**). This revealed that the raw read counts present a fairly similar number and dispersion across all samples, suggesting that the sequencing data is fairly similar in what concerns sequencing depth. As such, normalization might not be critical when analyzing the DE of this dataset.

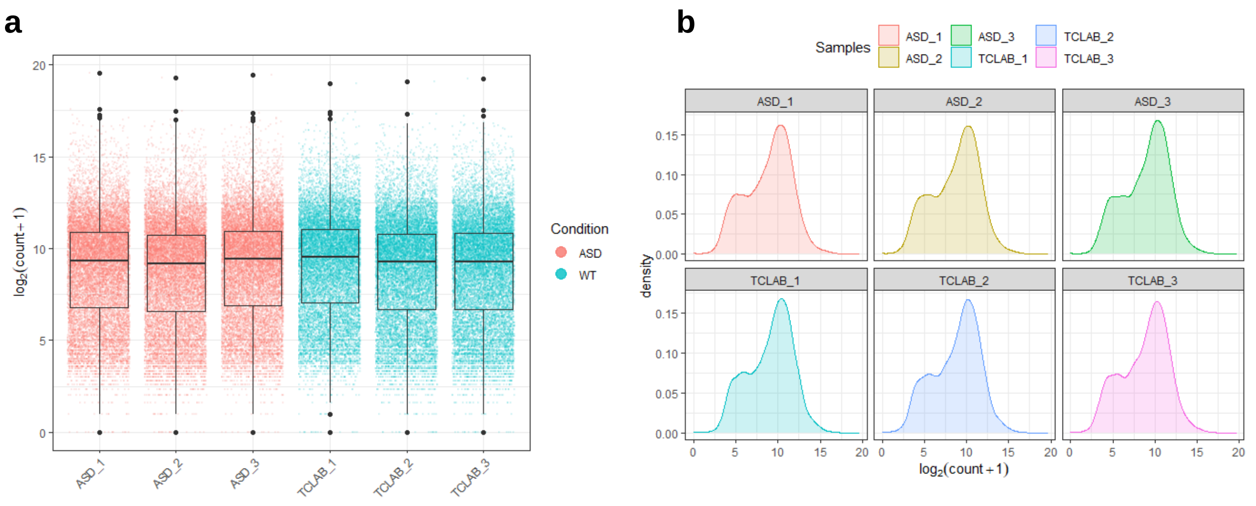

Figure 1 – Per sample distribution of read counts for ASD and TCLAB iPSCs. a- boxplots of log2-read count values showing expression distributions of unnormalised data. b- density of log2-read counts for raw data.

3.2. Normalization

The read counts for each gene are a direct consequence of its respective transcriptional activity. Nevertheless, several confounding factors can influence the read counts obtained for any given RNA-seq dataset. The main purpose of normalization is to dissect out uninteresting factors, which can introduce noise into the data. The most common confounding factors that have to be considered during normalization are: **library sequencing depth**, when comparing expression levels between samples; **gene length** and **GC-content** when comparing expression levels between different genes within the same sample; **RNA composition**, such as a few highly expressed genes, different number of expressed genes or presence of contaminants in different samples. As such, normalization is essential for DE analysis, exploratory data analysis, visualization of data and for comparing counts between or within samples.

3.2.1. EDASeq

EDASeq normalizes read counts in two ways: “within-lane” and “between-lane” level normalization. These strategies allow, dealing with within-lane gene-specific effects on read counts, such as gene length and GC-content, and with between-lane library sequencing depth differences. One could argue that for DE analysis between samples, factors like gene length and GC-content are irrelevant, as they do not change across samples. Nevertheless, some authors suggest that these factors can lead to selection biases on gene counts.

I tested the dependence of gene-level counts on GC-content, using the *biasPlot()* function in the EDASeq package (**Fig. 2a**). We can observe a slight bias towards genes with a lower GC-content. Nevertheless, the bias is reproducible across samples, suggesting that the RNA composition of the different libraries (as well as their sequencing depth) is comparable. I performed full-quantile normalization for within-lane, followed by between-lane, efficiently reducing the GC-content bias (**Fig. 2b**).

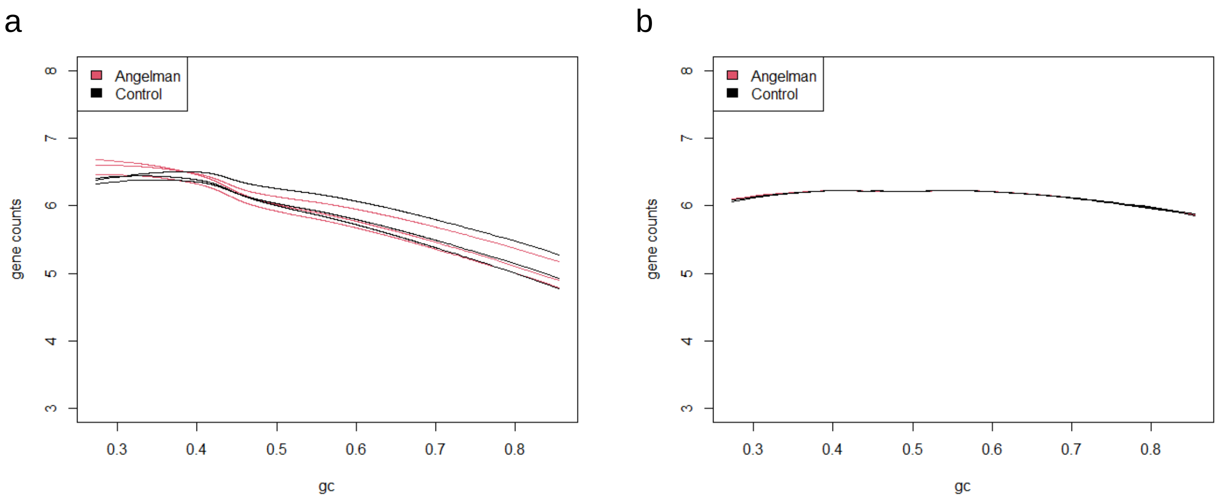

Figure 2 – EDASeq efficiently reduces GC-content bias on read counts. a- plot showing the GC-content bias of the raw filtered counts. b- plot showing the GC-content bias of the normalized counts, after within- and between-lane normalization.

Moreover, when I analyzed the distribution of read counts per gene in both raw and normalized data, I could observe a similar count distribution in between all 6 samples (**Fig. 3**).

3.2.2. edgeR

edgeR tackles normalization differently from EDASeq, taking into account differential expression analysis between samples, rather than the quantification of expression levels. In other words, it deals with relative changes in expression levels between conditions, but not absolute gene expression levels. By taking this approach, confounding factors like

GC-content and gene length are taken out of the equation, by assuming that they are similar between the different experimental conditions. As such, edgeR normalization deals essentially with sequencing depth and effective library size. Sequencing depth is self-explanatory and is related with the total number of sequenced reads per library. Effective library size is a more intricate concept related to the RNA composition of any given library. If a library has a few very highly expressed genes, reads coming from these genes will dominate the library, consuming a substantial proportion of the total library size for that particular sample. As a direct consequence, the remaining genes will be under-sampled and may appear as down-regulated.

edgeR starts with the premise that the majority of genes (more than half of the genes) are not differentially expressed between any pair of samples. Using *calcNormFactors()*, edgeR normalizes for library sizes by finding a set of scaling factors that minimize the log-fold changes between the samples for most genes. edgeR can use several methods for computing effective library sizes, but one of the most widely used is the trimmed mean of M-values (TMM) between each pair of samples. TMM is calculated as the weighted mean of log ratios between samples, after exclusion of the most expressed genes and the genes with the largest log ratios. Following the hypothesis of a low number of DEG, the TMM should be close to 1. The product of the original library size by the scaling factor gives us the effective library size, used in all downstream applications.

As edgeR works with CPMs, I plotted the raw read count distribution of log2 CPMs, and compared it with the same distribution after applying edgeR TMM normalization (**Fig. 3**). This led to a proper normalization of count distribution between all six samples.

3.2.3. DESeq2

Similarly to edgeR, DESeq2 assumes that no genes are differentially expressed. In addition, it has been exclusively devised for DE analysis, and the factors that might introduce variability between samples, namely sequencing depth and RNA composition. In contrast with the weighted mean of log-ratios method used by edgeR, DESeq2 uses a “geometric” normalization strategy. A scaling factor for a given sample is calculated as the median of the ratios, for each gene, of its read count over its geometric mean across all samples. In practice, this means that all non-DEGs should have similar read counts across all samples, leading to a ratio of 1. As DESeq2 assumes that the majority of genes are not DEGs, the median of these ratios, for each sample, provides an estimate of the correction factor that should be applied to all read counts of the sample. By using *estimateSizeFactors()*, this factor is computed for each lane allowing to obtain normalized read counts, by dividing the raw counts by the sample-specific factor.

After DESeq2 normalization of the raw count matrix, the analysis of the distribution of read counts per gene shows a similar count distribution in between all six samples (**Fig. 3**).

An important consideration with the three normalization strategies used, is that all these tools allow DE analysis, but only EDASeq is built to allow within sample comparisons (even though, edgeR can, to a certain degree perform this analysis).

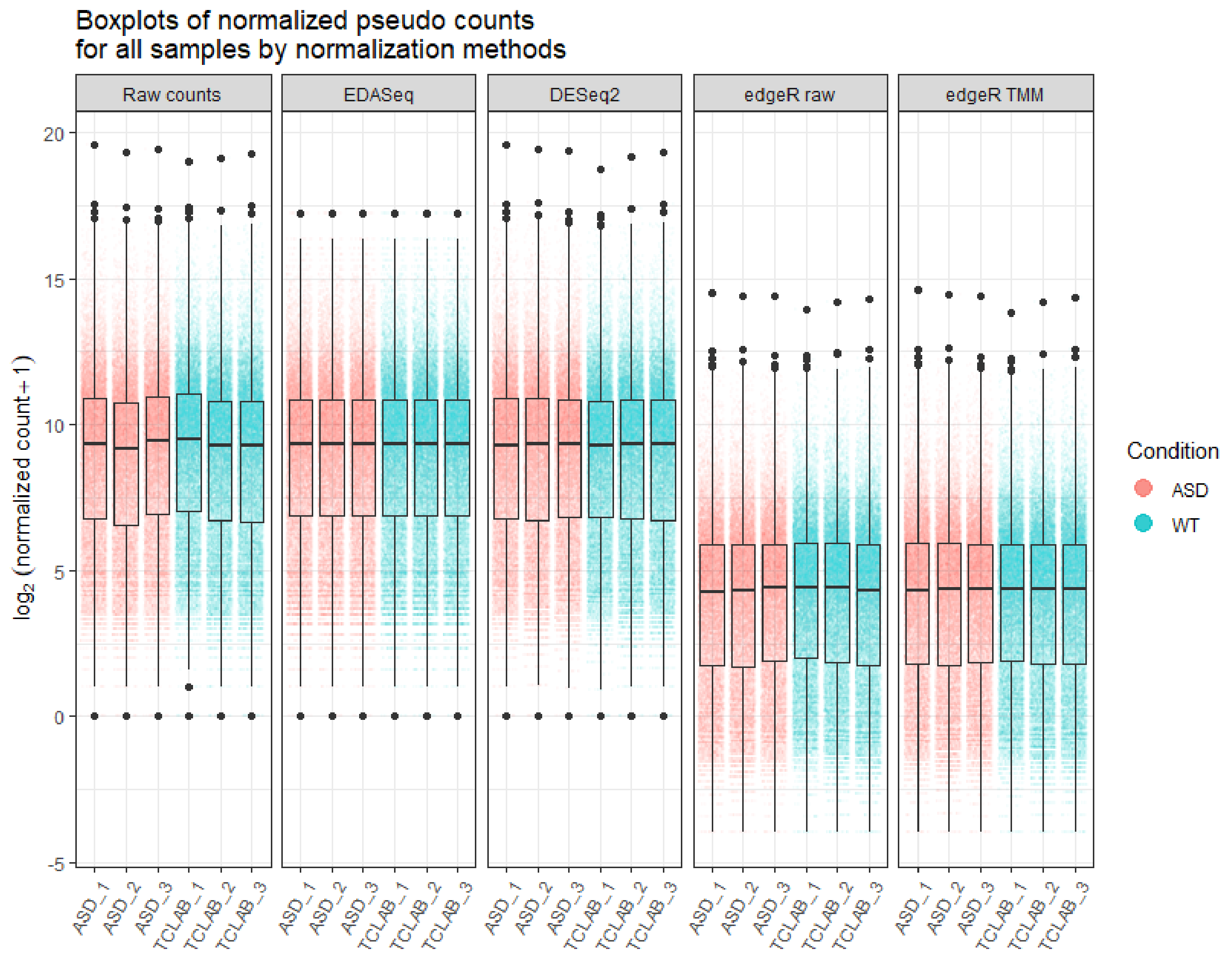

Figure 3 – Read count distribution is normalized using different normalization strategies. Boxplots showing raw read count distribution (plot 1), EDASeq normalized read count distribution (plot 2), DESeq2 normalized read count distribution (plot 3), raw CPMs distribution (plot 4) and edgeR TMM normalized CPMs distribution (plot 5).

3.3. Differential Expression Analysis

The final step in a DE analysis pipeline, is fitting the normalized counts to a determined model for gene count distribution in order to perform the statistical test for differentially expressed genes. In other words, we essentially try to determine whether the expression levels of different sample groups, for a given gene, are significantly different.

There are different methods for DE analysis, such as *edgeR*, *DESeq2* and *baySeq*, that assume that the read count distribution in RNA-seq experiments is modelled by a negative binomial distribution, or *limma-voom*, that implements linear models (using the normal distribution) to gene counts distribution, with observation weights derived from the empirical mean-variance trend.

The modelling process requires the use of a design matrix[19]. This matrix delineates the structure of the relationship between genes and explanatory variables (groups/conditions), and is used to store the values of these explanatory variables. In this exercise, I have used a simple design matrix, taking only one factor into consideration for modelling gene expression in the data: the phenotype of the patients, from which the cells were derived (Control Vs. Angelman).

3.3.1. EDASeq/edgeR

In the EE pipeline, read counts normalized with EDASeq within- and between-lane methods, were used for edgeR differential expression analysis. The EDASeq object, passed on to edgeR, contains raw read counts, plus an offset calculated during the normalization step. This offset information will be used for the statistical analysis for DE.

edgeR can use quantile-adjusted conditional maximum likelihood for experiments with single factors (qCML), or generalized linear models (GLMs), for more complex designs. Given the popularity of the GLM method, I have opted to use it for DE analysis. To calculate the differential expression in a gene-wise manner, edgeR uses the Cox-Reid profile-adjusted likelihood (CR) method, fitting generalized models to the gene counts, under the assumptions made in the design matrix. This is used to calculate the dispersion of read counts for all genes across all samples. Trended dispersion depending on the tag abundance, separate dispersion for individual tags, and common dispersion for all the tags, were calculated using the function *estimateDisp()* (**Fig. 4**).

After calculating dispersions, edgeR uses this information to fit negative binomial generalized linear models (using the *glmFit()* function), followed by testing for determining differential expression using either a quasi-likelihood (QL) F-test or a likelihood ratio test. For this exercise, I opted for using likelihood ratio testing, even though the QL test has been suggested to provide more robust and reliable error rate control when the number of replicates is small. For this, I used the *glmLRT()* function, allowing to determine whether genes are differentially expressed. Following this procedure, DEGs were extracted using *topTags()*, adjusting p-values for multiple testing using the Benjamini-Hochberg procedure. The top ten DEGs are provided in **Table 1**.

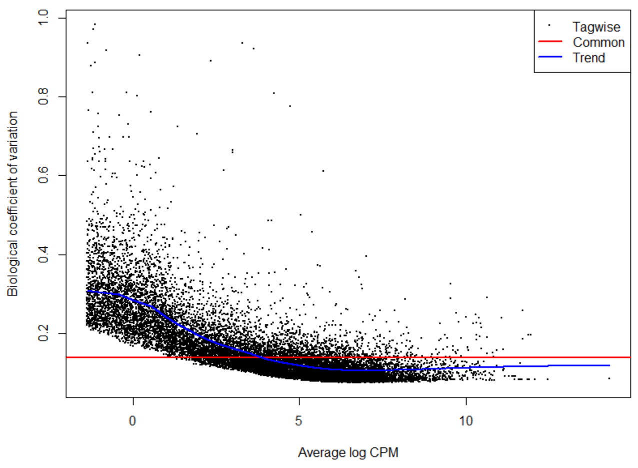

Figure 4 – Common, trended and tagwise dispersions across all tags following the EE pipeline. Biological coefficient of variation plot, showing the common (red line), trended (blue line) and tagwise (black points) dispersions after fitting a negative binomial model to the normalized count data.

Genes	logFC	logCPM	LR	PValue	FDR
POU5F1	10.544122	7.259568	2934.5138	0.000000e+00	0.000000e+00
CBS	4.379729	5.343845	953.9413	1.844986e-209	1.442687e-205
MTND2P28	4.484122	4.995175	704.8212	2.674894e-155	1.394422e-151
HLA-B	-8.985169	3.166133	628.4666	1.077251e-138	4.211783e-135
GATD3A	4.940548	3.106655	549.7829	1.403499e-121	4.389863e-118
NAP1L4	9.614190	2.946051	547.2559	4.976643e-121	1.297162e-117
PWP2	7.891243	3.047112	542.0170	6.864840e-120	1.533703e-116
CBSL	-3.898860	5.288390	523.7327	6.522977e-116	1.275161e-112
ZNF300	4.986786	4.805150	509.9004	6.666269e-113	1.158375e-109
GPX1P1	-5.449554	3.468169	398.9567	9.290648e-89	1.452964e-85

Table 1 – Top 10 DEGs following the EE pipeline (EDASeq normalization – edgeR differential expression). logFC – log2 fold change; logCPM- Average expression across all samples (log2 CPM); LR- likelihood ratio statistics; PValue- Raw p-value (based on LR) from test that logFC differs from 0; FDR- Benjamini-Hochberg false discovery rate adjusted p-value.

3.3.2. edgeR/limma

In the ErL pipeline, limma takes the filtered and normalized read count and, using the same design matrix employed for all DE analysis, will transform the data and prepare it for linear modelling, before testing for differential expression. I opted to use the edgeR TMM normalization in this pipeline, as it is the recommend method in the limma vignette, for dealing with RNA-seq data (as in microarrays, data does not follow a negative binomial distribution, but a linear/normal one).

The first step with the limma DE pipeline, is to apply *voom()*, which will effectively prepare the data for statistical testing using linear models. Using *voom()*, counts are transformed to log2 counts per million reads (CPM), based on the normalization factors calculated earlier. Then, linear models are fitted to each gene and the residuals calculated. Next, a smoothed curve is fitted to the residual standard deviation, by average expression (**Fig. 5a**) and this smoothed curved is used to obtain weights for each gene and samples that are passed into limma along with the log2 CPMs. In the case data was very noise, I could have employed a *voom* normalization method employed between microarrays, using *normalize* = *"quantile"*. This step was not performed, as the data did not demand it.

I next used *lmFit()*, to fit a linear model using weighted least squares for each gene, followed by Empirical Bayes smoothing of standard errors (using the *eBayes()* function), to shrink standard errors that are much larger or smaller than those from other genes, towards the average standard error. This allows producing a final model for gene dispersion, as seen in **Fig. 5b**. In addition, it also allows computing moderated t-statistics, moderated F-statistics, and logs-odd of differential expression. These statistics were used to extract DEGs, using limma *topTable()*, adjusting p-values for multiple testing, using the Benjamini-Hochberg method. The top ten DEGs for the ErL pipeline are presented in **Table 2**.

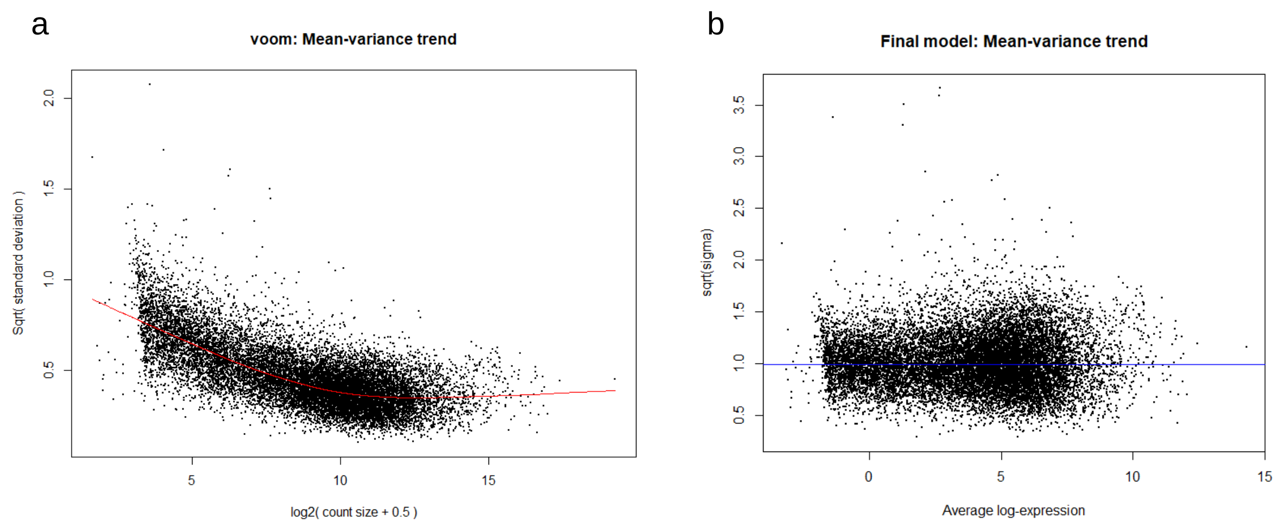

Figure 5 – Estimated mean-variance across all genes, following the ErL pipeline. a- Residual standard deviation plot per log2 CPM for all genes, showing the estimated mean-variance trend (red line) and tagwise (black points) standard deviation, after applying *voom*. b- means (x-axis) and variances (y-axis) of each gene are plotted to show how the dependence between the two are corrected after applying *voom* precision weights to the data. The log2 residual standard deviation estimated by the empirical Bayes algorithm is marked by a blue line.

Genes	logFC	AveExpr	t	P.Value	adj.P.Val	B
POU5F1	10.414141	3.0830609	33.02716	7.355970e-11	1.150400e-06	12.782722
CBSL	-4.510807	3.9429598	-28.66161	2.673735e-10	1.487606e-06	14.208635
MTND2P28	4.637326	3.6478099	27.66706	3.685565e-10	1.487606e-06	13.851852
LINC02864	-10.689556	-0.6048160	-27.57021	3.804862e-10	1.487606e-06	10.977768
ZNF208	9.722096	-1.1317484	25.69132	7.220476e-10	1.871220e-06	10.666252
XIST	4.347073	6.7019311	25.55877	7.567240e-10	1.871220e-06	13.414344
CBS	4.967919	3.8271820	25.27426	8.375562e-10	1.871220e-06	13.141297
ZNF560	8.843954	-1.5701463	24.69577	1.033101e-09	2.019584e-06	10.452145
GSTM1	8.689393	-1.6486660	23.00929	1.959603e-09	3.405136e-06	10.140050
NAP1L4	9.426075	-0.7507191	20.94943	4.569301e-09	6.762280e-06	9.770533

Table 2 – Top 10 DEGs following the ErL pipeline (edgeR normalization – limma differential expression). logFC – log2 fold change; AveExpr- Average expression across all samples (log2 CPM); t- logFC divided by its standard error; P.value- Raw p-value (based on t) from test that logFC differs from 0; adj.P.Val- Benjamini-Hochberg false discovery rate adjusted p-value; B- log-odds that gene is DE.

3.3.3. DESeq2

I next performed the DD pipeline, using DESeq2 for DE analysis. DESeq2 uses the *DEseq()* function, to apply a standard pipeline for differential expression and starts by estimating size factors, as described above for DESeq2 normalization. As normalization was already performed, this step was skipped. Next, the function obtains the dispersion estimates, assuming a negative binomial distribution of the data (using the same Cox-Reid method described above for edgeR). Finally, using the previously calculated size factors and dispersion estimates, the significance for the coefficients defined in the design matrix (Control Vs. Angelman, in this case) is tested assuming a negative binomial generalized linear model.

The dispersion estimates are, as with the previous methods, essential for the statistical inference about differential expression. The dispersion of the data after applying the NB model is displayed in **Fig. 6**.

The gene wise statistical tests performed above are used to extract DEGs using the *results()* function. As with the previous pipelines, the Benjamini-Hochberg method is used to adjust p-values for multiple testing. The top ten DEGs for the DD pipeline are presented in **Table 3**.

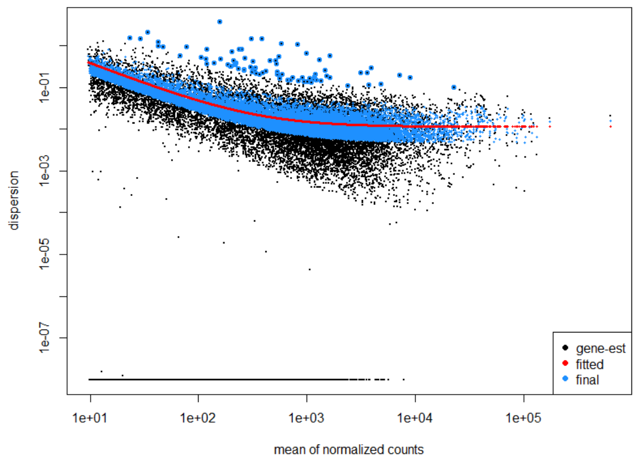

Figure 6 – Dispersion plot for gene counts across samples, following the DD pipeline. Means (x-axis) and variances (y-axis) of each gene are plotted to show how the dependence between the two are corrected after running the *DESeq()* function. The black points represent the dispersion estimates for each gene. The dispersion dependence on the mean is calculated and fit into the model (red line). Each estimate for individual genes towards the fit line are used to calculate the final estimates (blue points) that are then used in the hypothesis testing. Blue circles above the main cloud of points are genes with high gene-wise dispersion estimates, that are labelled as outliers and not fitted to the trend line.

Genes	baseMean	log2FoldChange	lfcSE	stat	pvalue	padj
POU5F1	4899.236	10.49109	0.333102	31.4951	1.01421e-217	1.58582e-213
CBSL	1188.492	-4.48970	0.148643	-30.2046	2.06394e-200	1.61359e-196
MTND2P28	1015.913	4.64287	0.158113	29.3642	1.57262e-189	8.19648e-186
CBS	1292.363	4.95399	0.179972	27.5265	8.46983e-167	3.31086e-163
XIST	7767.756	4.33936	0.176533	24.5809	2.01964e-133	6.31582e-130
ZNF300	893.329	5.07575	0.218401	23.2405	1.77440e-119	4.62407e-116
GATD3A	272.444	4.59565	0.232777	19.7427	9.27346e-87	2.07143e-83
GPX1P1	337.262	-5.25259	0.273141	-19.2303	2.06373e-82	4.03356e-79
RPSAP58	1123.221	2.60398	0.141455	18.4086	1.12121e-75	1.94792e-72
ACTA1	1114.312	-4.59146	0.250230	-18.3490	3.36187e-75	5.25663e-72

Table 3 – Top 10 DEGs following the DD pipeline (DESeq2 normalization – DESeq2 differential expression). baseMean - Average expression across all samples (normalized read counts); log2FoldChange – log2 fold change; lfcSE- log2 fold change standard errors, stat- test statistics (Wald statistic); pvalue- Raw p-value (based on stat) from test that logFC differs from 0; padj- Benjamini-Hochberg false discovery rate adjusted p-value.

3.3.4. baySeq

As a fourth, and "extra" method, I have used the baySeq package for read count normalization and DE analysis. baySeq is an empirical Bayesian approach, that uses prior information to fit a nonlinear model that quantifies biases and uncertainties in the estimates of posterior likelihoods for each gene being tested. Similarly to edgeR and DESeq2, baySeq assumes that the read counts follow a negative binomial distribution.

As described in the methods section, the normalization I performed used the TMM method, similarly to edgeR. For differential expression analysis, I started by estimating the empirical distribution of prior parameters using the *getPriors()* function, which assumes a NB model for the data and extracts posterior parameters from this distribution using quasi-likelihood methods. Next, using the function *getLikelihoods()*, I estimated the log posterior likelihoods of each count belonging to a given group (Control or Angelman), defined in the design matrix. Finally, using *topCounts()*, the posterior likelihoods previously calculated were used to return the DEGs with the highest likelihood of association with a given group (Control of Angelman, in this case). As with all the other pipelines, False Discovery Rate (FDR) using the Benjamini-Hochberg method was used to correct for multiple testing. The top ten DEGs for the BB pipeline are represented in **Table 4**.

Genes	Control	Angelman	log2FC	likes	DE	FDR.DE	FWER.DE
POU5F1	8.00000	9651.6667	10.236562	1.0000000	2>1	0.000000e+00	0.000000e+00
XIST	769.33333	14680.6667	4.254165	1.0000000	2>1	0.000000e+00	0.000000e+00
CBSL	2344.33333	100.6667	-4.541520	1.0000000	1>2	1.670893e-09	5.012680e-09
CBS	85.33333	2475.3333	4.858370	1.0000000	2>1	3.369493e-09	1.347797e-08
MTND2P28	81.00000	1925.0000	4.570793	1.0000000	2>1	4.676157e-09	2.338079e-08
GATD3A	23.33333	518.0000	4.472488	1.0000000	2>1	1.157859e-08	6.947152e-08
LINC02864	860.00000	1.0000	-9.748193	0.9999997	1>2	4.823685e-08	3.376580e-07
GPX1P1	668.33333	18.0000	-5.214499	0.9999996	1>2	8.986348e-08	7.189077e-07
PCDHB5	52.00000	535.6667	3.364752	0.9999995	2>1	1.366313e-07	1.229681e-06
ZNF300	53.33333	1704.3333	4.998026	0.9999993	2>1	1.958055e-07	1.958053e-06

Table 4 – Top 10 DEGs following the BB pipeline (baySeq normalization – baySeq differential expression). Control- Average expression across all samples from the control group (normalized read counts); Angelman- Average expression across all samples from the Angelman group (normalized read counts); log2FC – log2 fold change; likes- likelihood; DE- Differential expression groups; FDR.DE- Benjamini-Hochberg false discovery rate adjusted p-value, FWER.DE- Family wise error rate.

4. Comparison of the different pipelines

4.1. Comparison of DEGs and log2 Fold Changes

From the DE analysis performed following the 4 proposed pipelines, we can observe that the top DEGs (**Tables 1 to 4**) are quite reproducible in between the different strategies employed in this study. To have a more global idea of the DEGs, I have extracted all genes with an adjusted p-value lower than 0.05 (FDR of 5%). To compare the list of DEGs, I have plotted a Venn diagram using the R package *ggvenn* (**Fig. 7a**), showing the common identified DEGs for each of the pipelines. In addition, I have used the R package *upset*, to visualize the set of intersections between the different pipelines (**Fig. 7b**).

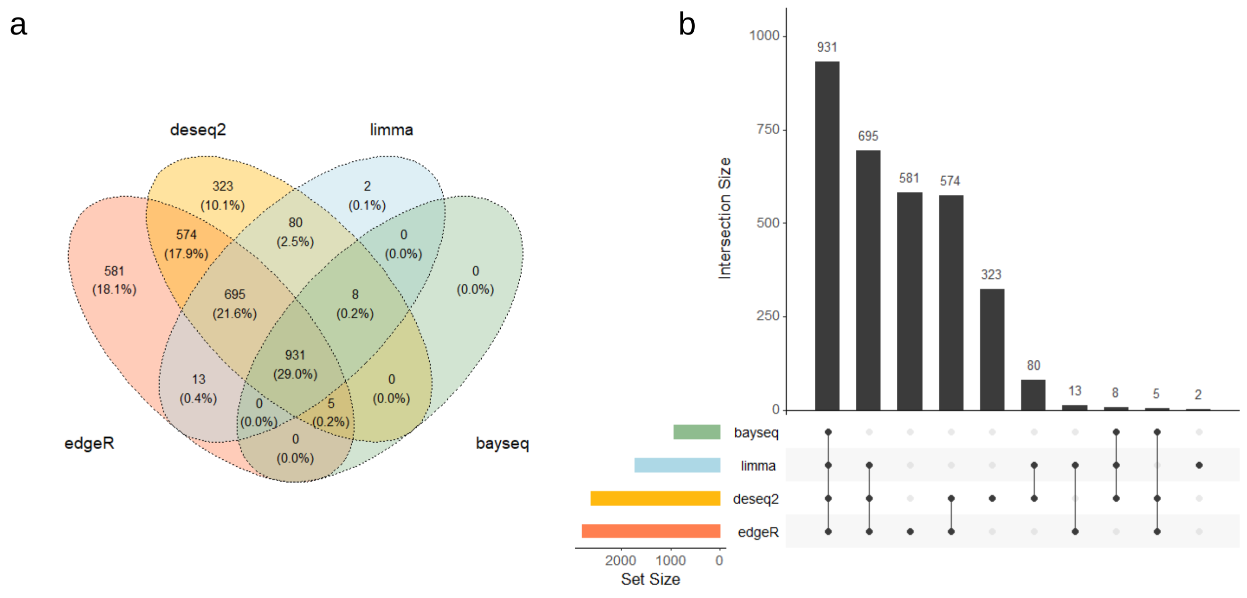

Figure 7 – Visualizing intersections of DEG set for all DE pipelines. a- Venn diagram showing intersection of DEGs for EE (red), DD (orange), ErL (blue) and BB (green) pipelines. Number of common genes and percentages are indicated. b- Equivalent UpSet plot, with a combination matrix that identifies the intersections, with bars that encode the size of each intersection.

The analysis of the plots, shows that the majority of the gene set is shared between the 4 pipelines (931 genes), followed by a gene set that is shared between the EDASeq-edgeR (EE), edgeR-limma (ErL) and DESeq2 (DD) pipelines (695 genes). Moreover, the number of DEGs identified by DD and EE were the highest, with 2616 and 2799 genes, respectively. The ErL and baySeq (BB) pipelines shown the most stringent number of DEGs, with 1723 and 944, respectively. Interestingly, only the BB pipeline had no uniquely identified DEGs, with the majority of genes identified shared between the four pipelines. Although no strong conclusion can be extracted from this crude analysis of the data, it is obvious that there is a strong correlation between the list of DEGs identified using the different methodologies. Moreover, one has to be particularly cautious with the loose definition of DEG used here, in which the only criteria chosen is an adjusted p-value below 0.05. For biological exploration of the data, more stringent criteria would have been used.

To have a more precise idea about how the normalization affects individual gene counts, I have next took the normalized read counts for the common DEGs for the four pipelines, with an adjusted p-value lower than 0.05. Using a pairs plot (*pairs()* function in R), I produced a matrix of scatterplots comparing the log2 fold-change (log2FC) of the DEGs, between all the 4 pipelines (**Fig. 8**). We can see that all normalization methodologies (EDASeq for the edgeR DE analysis, edgeR for limma DE analysis, DESeq2 and baySeq), produce quite correlated normalized log2FCs. The correlation was solid all over the log2FC ranges, with a slight deviation on both the lowest and highest values. Importantly, all tools agree on the direction and magnitude of the log2FC. These observations suggest that the normalization strategies undertaken lead to similar results which, unsurprisingly, leads to a comparable list of DEGs. It also suggests that the different normalization steps are solid and capable of producing a robust normalized count matrix for the differential expression analysis, downstream.

Another way of looking at the log2FC for all common DEGs (FDR 5%), is to plot the log2FC for all DEGs in a clustered heatmap (using the R *pheatmap* package), for all 4 different pipelines (**Fig. 9**). This will give us a more visual way of looking at how the DEGs behave, in the different pipelines explored. By doing so, I could confirm the very close correlation between gene-wise log2FCs across the four strategies. Moreover, the pipelines separated in two clusters, EE with DD, and ErL with BB, which is not surprising, given the number of common DEGs between each of these groups of pipelines.

Another interesting observation, is that the majority of the identified DEGs show very low log2FC, and only a few seem to be highly deregulated. As mentioned, this is justified by using only adjusted p-value to define DEGs, which in a biological and functional scenario, is not a robust way of tackling differential expression. Nevertheless, this is justified for the comparison of the different pipelines in this project.

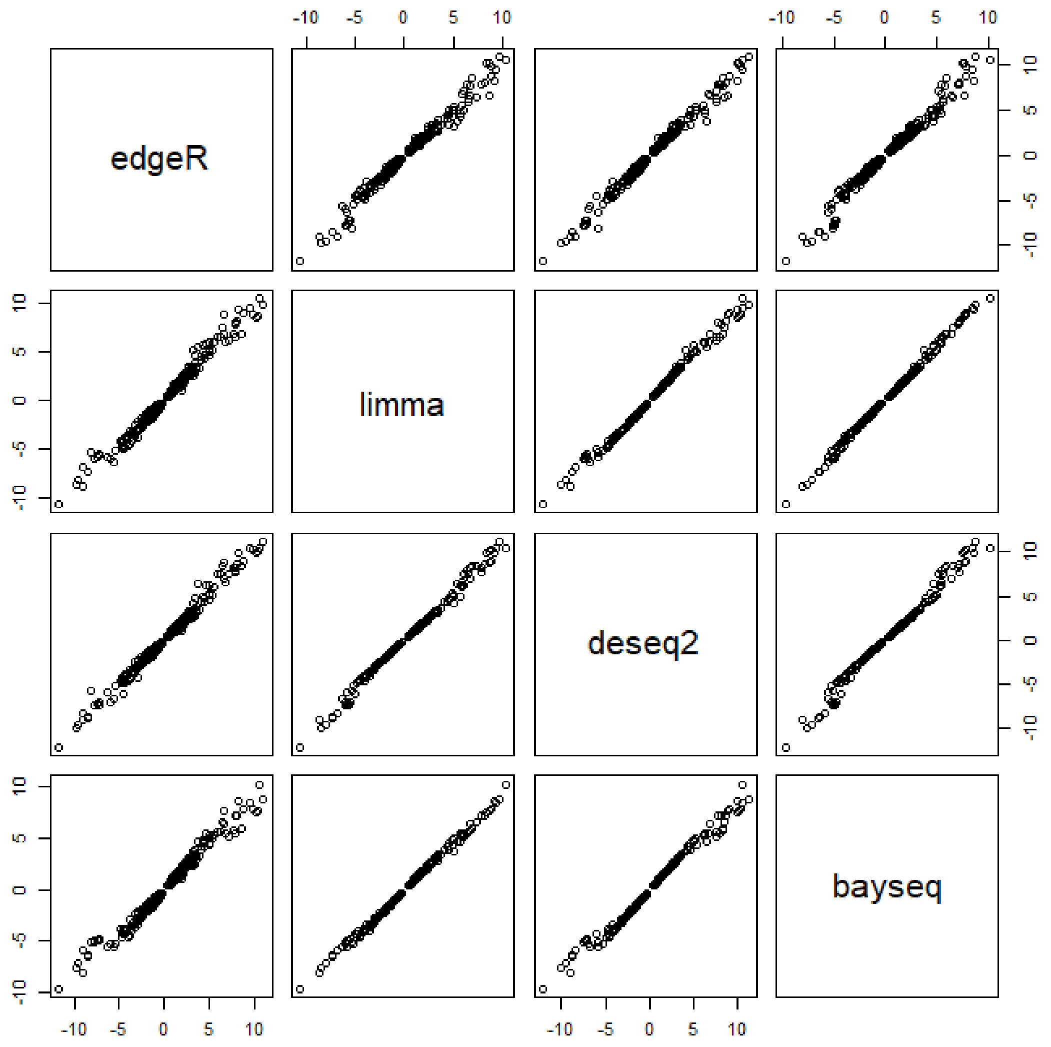

Figure 8 – Pairwise comparison of log2FCs estimates for the four DE pipelines. Pairwise scatter plots for gene expression data (log2FC) from EE, ErL, DD and BB pipelines. Panel titles specify the pipelines used as reference for pairwise comparison.

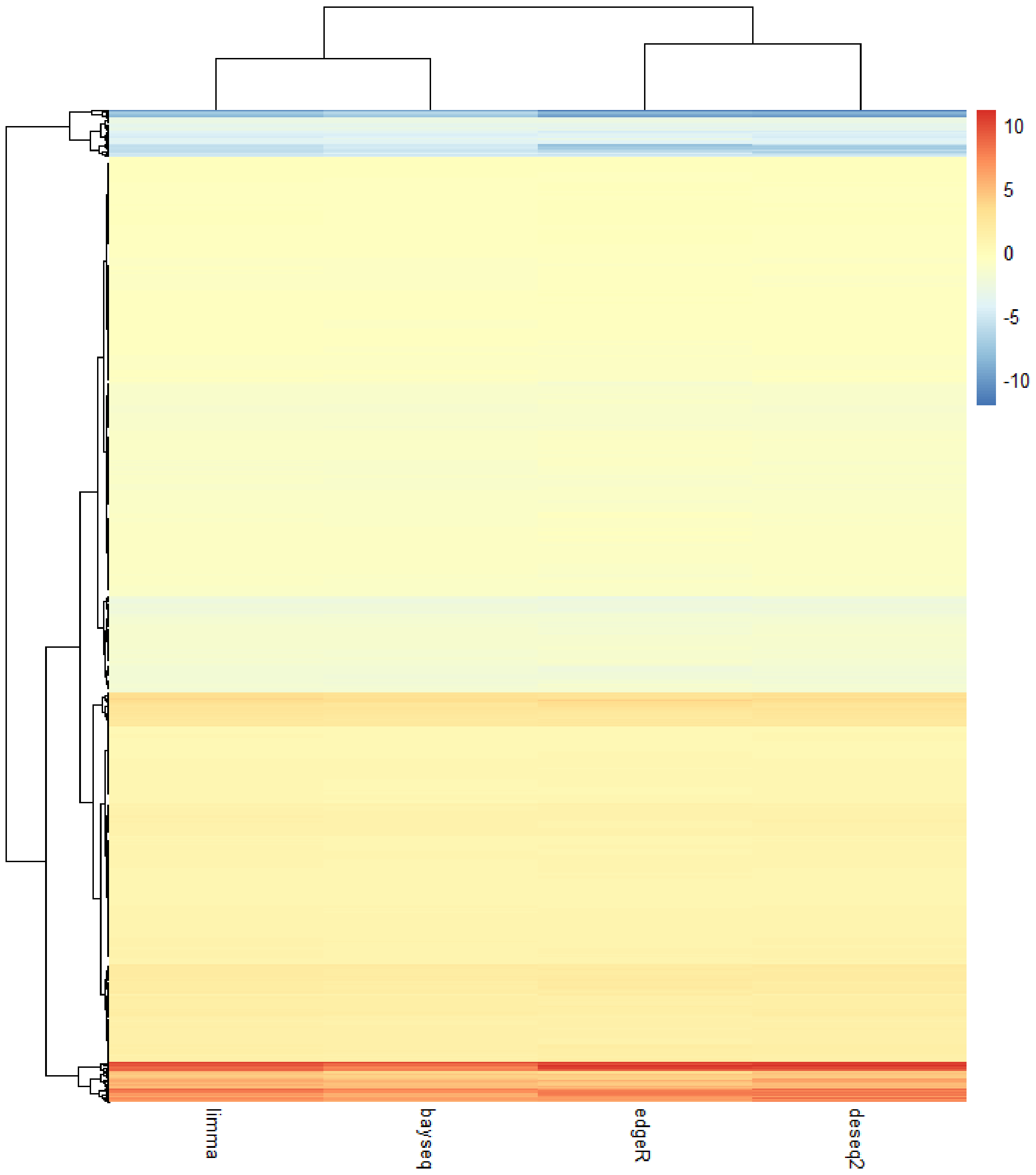

Figure 9 – Summaries of log2 fold-change estimates among the different RNA-seq pipelines. Clustered heatmap showing log2FC for all common DEGs, in the ErL, BB, EE and DD pipelines. Each row represents one of the common DEGs between the four pipelines, each column a different pipeline. Hierarchical clustering of DEG groups and pipelines is represented.

4.2. Visual inspection of statistically significant DEGs

To have a better insight on the results of the DE analysis, for the different pipelines, I have created volcano plots using *ggplot2* (**Fig. 10**). These scatterplots show the statistical significance (-log10 adjusted p-value or FDR) versus the magnitude of change (log2 fold change). This enables quick visual identification of genes with large fold changes that are also statistically significant, which may be the most biologically significant genes. The volcano plots show non-significant genes (gray), non-significant genes that show more than log2FC of |2| (green), genes filtered on an adjusted p-value lower than 0.05 (blue) and genes with an adjusted p-value below 0.05 and log2FC bigger than |2|. Unsurprisingly, the set of top DEGs is very similar between the different methodological approaches, further reinforcing that not a single method produces dramatically different results and that all methods worked robustly with this dataset.

This approach allows identifying top candidates that are upregulated (top right quadrant of the volcano plots) or downregulated (top left quadrant of the volcano plots) and that can be further explored, using gene set enrichment analysis, for example. Additionally, these genes are prime candidates for being related to the Angelman phenotype.

Interestingly, *UBE3A*, the gene that is mutated in the Angelman cell lines, albeit being statistically differentially expressed, shows low log2FCs. This has an easy biological explanation: this gene is maternally imprinted, being expressed mostly from the paternal allele. Given that the mutant cell line harbors a maternal deletion, its effect on the global levels of *UBE3A* are only mildly felt.

4.3. Sample-wise gene expression levels for top DEGs

To finalize my comparison, the top 30 most statistically significant DEGs were obtained for each pipeline, by sorting by adjusted p-value. The normalized read counts for each sample, were extracted and this filtered count matrix, was used to produce clustered heatmaps (using the R *pheatmap* package), representing the scaled gene expression across all samples (**Fig. 11**). This allows to quickly visualize the relative expression levels of the top 30 most significant genes, across all samples, clustering samples by columns (Control Vs. Angelman), and genes by rows (group of co-regulated genes).

Even if the list of top-30 genes is not completely identical between the different DE pipelines, several genes are found to be up- or down-regulated across the four methodologies, which reinforces the idea that the different approaches produce similar results.

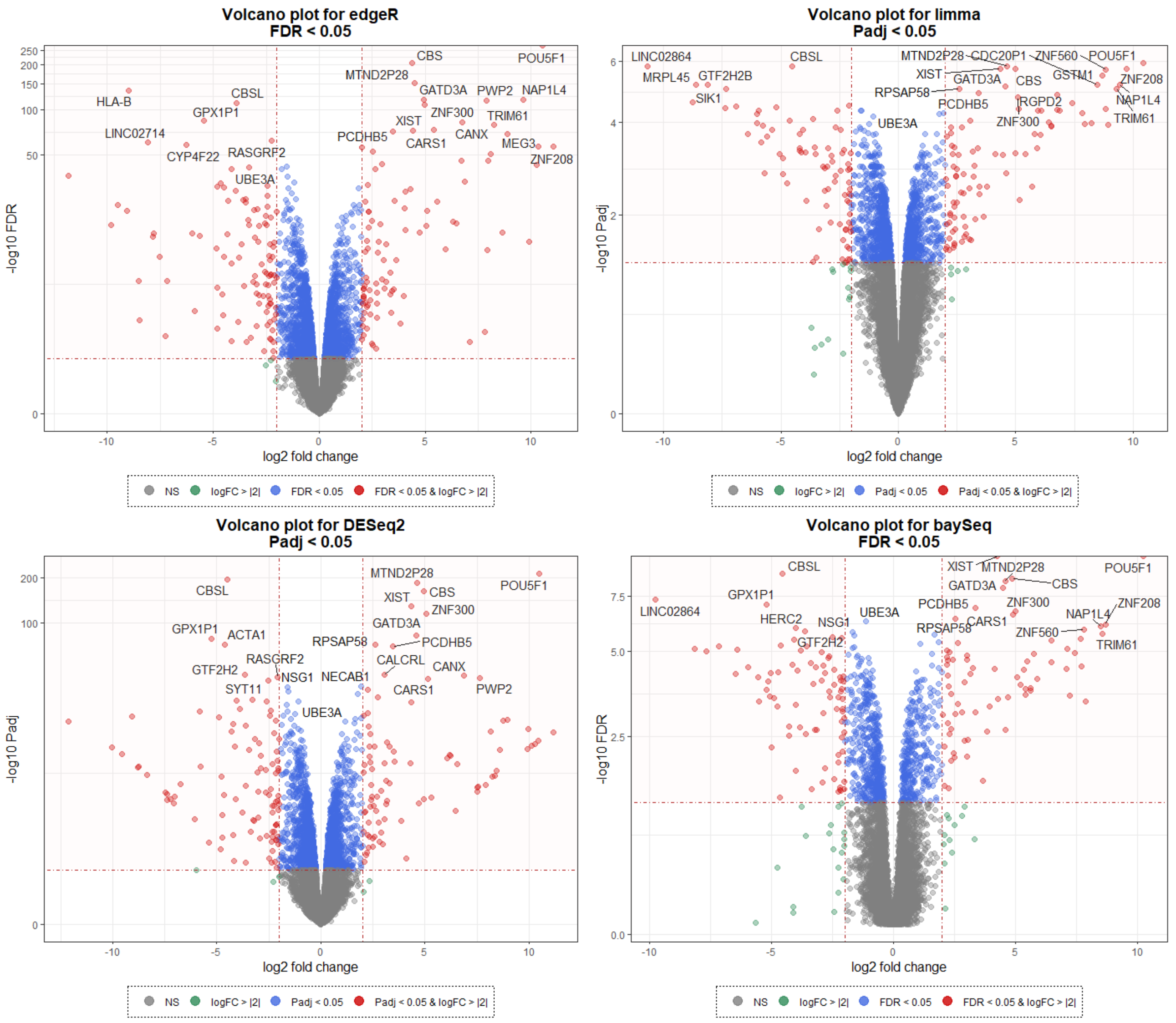

Figure 10 – RNA-seq data analysis using volcano plots. Volcano plots of individual DE analysis pipelines, showing non-significant genes (gray), non-significant genes with higher than log2FC of |2| (green), significant genes with adjusted p-value lower than 0.05 (blue) and genes with adjusted p-value lower than 0.05 and log2FC higher than |2| (red). Red dotted lines represent log2FC threshold of + and – 2, and adjusted p-value of 0.05. Top right quadrant, representing statistically significant upregulated genes and top left quadrant, representing statistically significant downregulated genes, are represented in red. From top left to bottom right: EDASeq-edgeR, edgeR-limma, DESeq2 and baySeq pipelines.

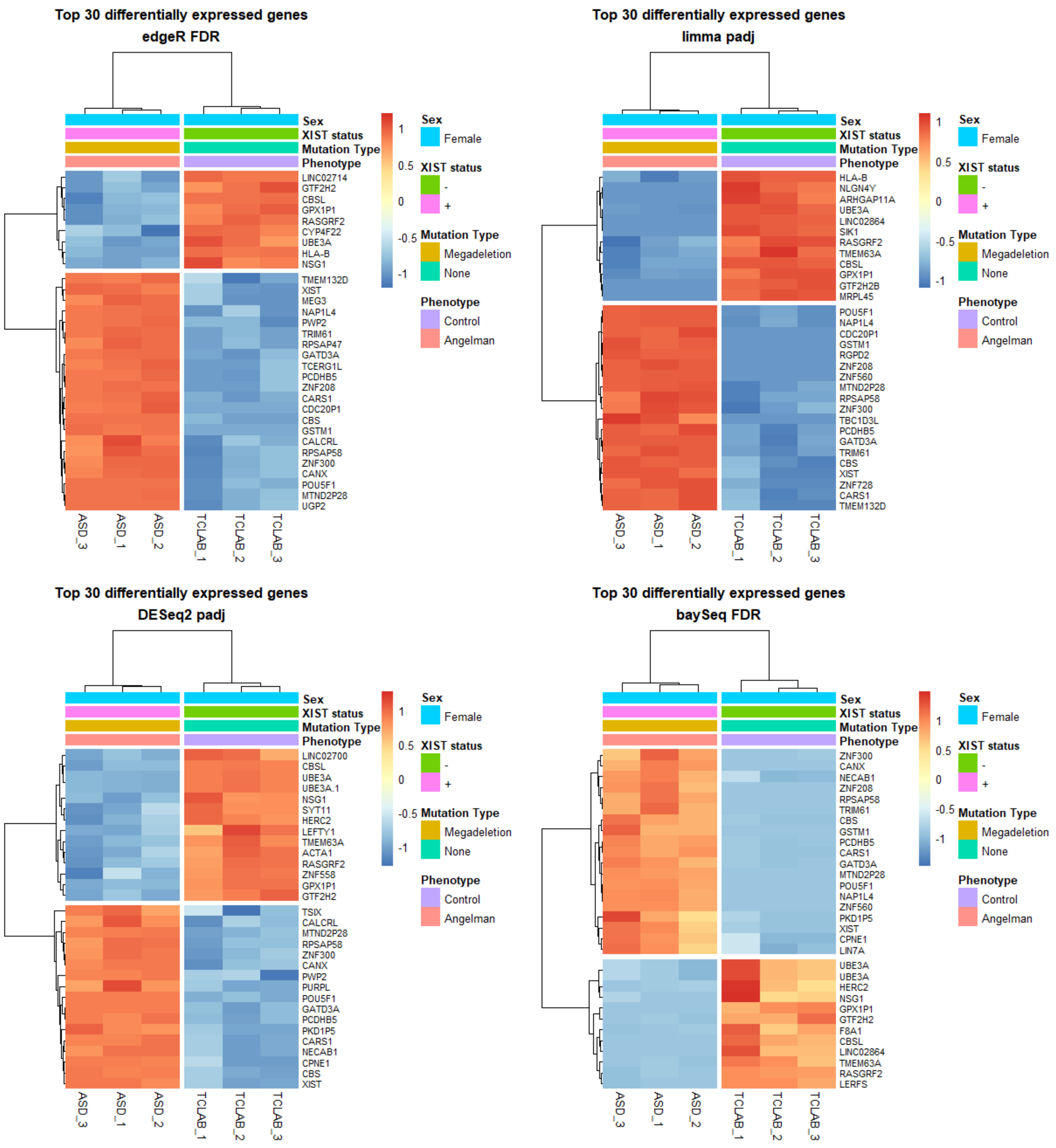

Figure 11 – Heatmaps of normalized read counts for top 30 statistically significant DEGs in the four DE analysis. Expression across each gene has been scaled per row, so that colors actually represent z-scores (obtained by subtracting the mean of read counts, to each individual sample count and then dividing by the standard deviation). Samples with relatively high expression of a given gene are marked in red and samples with relatively low expression are marked in blue. Genes and samples are reordered according to hierarchical clustering. Dendograms are shown for the sample and gene clustering. From top left to bottom right: EDASeq-edgeR, edgeR-limma, DESeq2 and baySeq pipelines.

5. Discussion

The main goal of this academic project was to compare several statistical methods for differential gene expression analysis based on RNA-seq data. I focused on frequently used methods available in Bioconductor. For normalization, I have used EDASeq non-linear full quantile normalization, edgeR trimmed mean of M-values, DESeq2 median ratios method and baySeq trimmed mean of M-values. For differential expression analysis, I have used edgeR glm, limma-Voom, DESeq2 and baySeq.

The study performed here is not an exhaustive comparison about the strengths and pitfalls of each method, as this would be beyond the scope of this exercise. Instead, I tried to use a set of well documented and commonly used strategies to analyze a RNA-seq dataset that was kindly provided by Simão Teixeira da Rocha (iMM). This RNA-seq dataset consists of 3 control iPSCs and 3 Angelman syndrome iPSCs. The quality of the sequenced data was very high for all samples. As such, whatever pitfalls the different tools might have revealed when dealing with bad quality data, have not been exposed.

For data normalization, likely justified by the good quality of the RNA-seq data, all tools led to comparable normalized read counts (as can be observed in **Fig. 3 and 8**). No single normalization strategy demonstrated to be a clear winner. Further analysis, using noisy data, would be needed to properly test the efficiency of each tool. Moreover, in a different test scenario, different normalization methods implemented in the different packages used could have been tested.

One interesting observation about the normalization methods is the full-quantile normalization employed by EDASeq. Full-quantile normalization was originally developed to deal with microarray data and generally leads to perfectly distributed data amongst samples (which can actually be observed in **Fig. 3**). Moreover, EDASeq is the only normalization strategy, which normalizes for within-sample distribution of gene counts, taking GC-content into consideration. Whereas this could have led to different results compared to other normalization strategies, this was not the case. This is likely due to the fact that small variability was observed within and across the groups of samples (control and Angelman).

As such, the different normalization strategies generated a very comparable set of normalized count matrixes, which were then used for the different DE pipelines. These pipelines included: edgeR glm, using likelihood ratio test for testing for differential expression; limma-Voom, fitting linear models and applying moderated t-test for differential expression; DESeq2, using empirical Bayes to shrink log fold changes and Wald Test, to test for differential expression; and baySeq, using the posterior distribution of the gene count distribution, to test for differential expression. After running the DE pipelines, I filtered genes using a FDR of 5% (adjusted p-value of 0.05). This revealed that the DE analyses using edgeR (on EDASeq normalized data) and DESeq2 (for both normalization and DE analysis) produces a list of 2799 and 2616 DEGs, respectively, the vast majority of these being common to both methods (**Fig. 7**). Revealing a tighter control on the FDR, limmaVoom (after edgeR normalization) and baySeq (for both normalization and DE analysis) produced a list of 1723 and 944 genes, respectively (**Fig. 7**). No formal testing was performed to verify the power of the DE testing and FDRs. Nevertheless, it is clear that edgeR and DESeq2, under the default parameters used, are less stringent when testing for differential expression. Further testing using different parameters (like edgeR robust, instead of glm, or using likelihood tests, instead of Wald test for DESeq2) could be used to give a tighter control on the obtained FDRs. Moreover, more robust testing would have been needed to compare the true FDR with the target FDR.

The similarities observed between the EE and DD pipelines are not surprising, as they are quite similar approaches for DE analysis. In contrast, the linear model approach, used by limma, and the empirical Bayesian approach, employed by baySeq, seem to provide a better control on the FDR, while maintaining a robustness in finding DEGs. This is supported by two observations: 1- the DEGs detected by each of these approaches overlaps those of the 2 other pipelines; 2- much less DEGs are detected, suggesting a stricter control on FDR estimation. Further testing on the approximation of the predicted FDR to the real FDR, for all 4 methodologies, would have been needed to test these hypotheses.

Independently of the above, we should consider that I have employed quite loose filtering options to find DEGs. In a biological context, using a threshold for adjusted p-value of 0.05, when we have thousands of DEGs, is likely not stringent enough. Moreover, an important parameter when defining DEGs is the fold-change observed between control and test groups. A statistically significant test might not always translate to a biologically relevant difference and this can be partially tackled by filtering genes that actually show a measurable fold-change. In this exercise, I filtered genes showing at least 4 times more or less read counts in the test group, compared to the control group. Doing so effectively produces a list of DEGs that are highly comparable between the different pipelines (**Fig. 10 and 11**).

Answering the question of which method produces the best results is not a trivial task. First, we should keep in mind that each tool will have advantages under different circumstances, which have not been covered in this project. Second, when determining which methodology to use, one has to keep in mind the ability of the tools to detect real DEGs and their ability to prevent false detection. Although I have not precisely tested the ability of edgeR and DESeq2 to accurately predict DEGs, I feel that using these tools will give me a robust set of DEGs, together with potential, less robust DEGs that would require further confirmation (and that would have likely been lost with the two other approaches).

Independently of the tool used, the most reasonable approach is to use a fair judgement about the list of DEGs and use these as a means to further explore a particular biological question, and not as an end for our analysis.

6. Bibliography

1. Mortazavi, A., Williams, B. A., McCue, K., Schaeffer, L. & Wold, B. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat Methods* **5**, 621–628 (2008).
2. Marioni, J. C., Mason, C. E., Mane, S. M., Stephens, M. & Gilad, Y. RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays. *Genome Res* **18**, 1509–1517 (2008).
3. Oshlack, A., Robinson, M. D. & Young, M. D. From RNA-seq reads to differential expression results. *Genome Biol* **11**, 220 (2010).
4. Osabe, T., Shimizu, K. & Kadota, K. Accurate Classification of Differential Expression Patterns in a Bayesian Framework With Robust Normalization for Multi-Group RNA-Seq Count Data. *Bioinform Biol Insights* **13**, (2019).
5. Tang, M., Sun, J., Shimizu, K. & Kadota, K. Evaluation of methods for differential expression analysis on multi-group RNA-seq count data. *BMC Bioinformatics* **16**, 360 (2015).
6. Schurch, N. J. *et al.* How many biological replicates are needed in an RNA-seq experiment and which differential expression tool should you use? *RNA* **22**, 839–851 (2016).
7. Huber, W. *et al.* Orchestrating high-throughput genomic analysis with Bioconductor. *Nat Methods* **12**, 115–121 (2015).
8. Robinson, M. D., McCarthy, D. J. & Smyth, G. K. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics* **26**, 139–140 (2010).
9. Ritchie, M. E. *et al.* limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic Acids Res* **43**, e47 (2015).
10. Law, C. W. *et al.* RNA-seq analysis is easy as 1-2-3 with limma, Glimma and edgeR. *F1000Res* **5**, (2018).
11. Risso, D., Schwartz, K., Sherlock, G. & Dudoit, S. GC-Content Normalization for RNA-Seq Data. *BMC Bioinformatics* **12**, 480 (2011).
12. Love, M. I., Huber, W. & Anders, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology* **15**, 550 (2014).
13. Hardcastle, T. J. & Kelly, K. A. baySeq: Empirical Bayesian methods for identifying differential expression in sequence count data. *BMC Bioinformatics* **11**, 422 (2010).
14. Maranga, C., Fernandes, T. G., Bekman, E. & Rocha, S. T. da. Angelman syndrome: a journey through the brain. *The FEBS Journal* **287**, 2154–2175 (2020).
15. Babraham Bioinformatics - FastQC A Quality Control tool for High Throughput Sequence Data. https://www.bioinformatics.babraham.ac.uk/projects/fastqc/.
16. Babraham Bioinformatics - Trim Galore! https://www.bioinformatics.babraham.ac.uk/projects/trim_galore/.
17. Dobin, A. *et al.* STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* **29**, 15–21 (2013).
18. Liao, Y., Smyth, G. K. & Shi, W. featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics* **30**, 923–930 (2014).
19. Law, C. W. *et al.* A guide to creating design matrices for gene expression experiments. *F1000Res* **9**, (2020).