International Journal of
Advances in Scientific Research and Engineering (ijasre)

E-ISSN : 2454-8006

DOI: http://doi.org/10.31695/IJASRE.2018.32720

Volume 4, Issue 7
July - 2018

Thorough Assessment on Differential Gene Expression Analysis Methods for RNA-seq Data

Fahmida Pervin Brishty[1] and MD. Saifur Rahman[2]

Department of Information and Communication Technology, Computer Science

Bangladesh University of Engineering and Technology

Dhaka

Bangladesh

ABSTRACT

Abstract Recently, RNA profiling based on HT sequencing is replacing microarrays for the study of differential gene expression. In practice, millions of 'reads' are sequenced from random positions of the input RNAs which are computationally mapped on a reference genome to reveal a 'transcriptional map', where the number of reads aligned to each gene gives a measure of its level of expression. The powerful features of RNA-seq, such as high resolution and broad dynamic range, have boosted an unprecedented progress of transcriptomics research, producing an impressive amount of data worldwide. To deal with the different steps of data analysis several computational tools have been developed and updated at a fast pace which is in fact far more complex and consists in several processing steps. In this review, we describe the current RNA-seq analysis framework, focusing on each computational step from read preprocessing to differential expression (DE) analysis. We review the methodologies available, along with their underlying algorithmic strategies and believe this work can provide a broad overview of RNA-seq analysis and can guide users to define and implement their own processing pipeline.

Key Words: *RNA-seq, transcriptomics, DE analysis, HT sequencing,* FDR.

1. INTRODUCTION

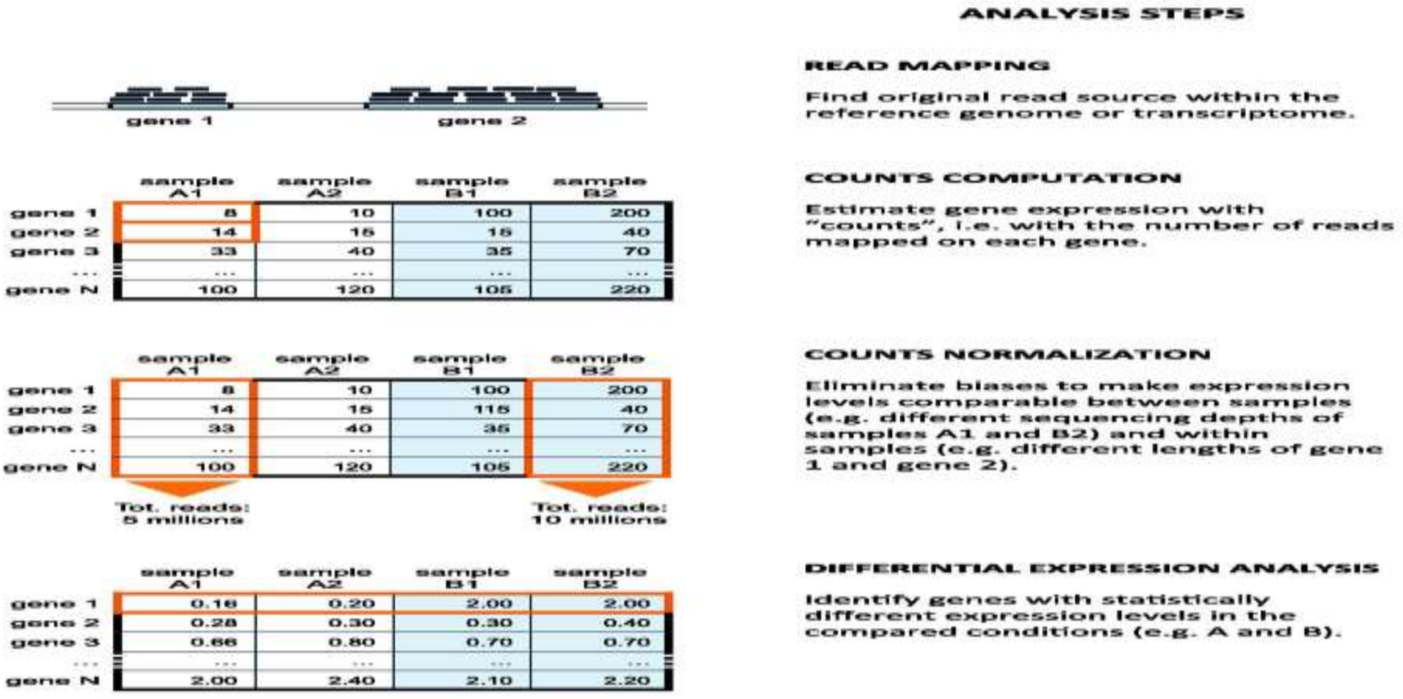
In case of high throughput (HT) sequencing, a fundamental task is the analysis of count data, such as read counts per gene in RNA-seq, for evidence of systematic changes across experimental conditions. DESeq2 is a method for differential analysis of count data, using shrinkage estimation for dispersions and fold changes to improve stability and interpretability of estimates. This enables a more quantitative analysis focused on the strength rather than the mere presence of differential expression. The DESeq2 package is available at Bioconductor. We mainly focus on the analysis of differential gene expression studies. We discuss strategies for package based analysis, non-specific and meta-data driven pre-filtering techniques, and commonly used test statistics for detecting differential expression. We show how these strategies and statistical tools are implemented and used in Bioconductor. RNA-Seq can be used to quantify and study genome-wide changes in gene expression. Such applications typically start with aligning RNA-Seq reads to a reference sequence to identify all expressed genome features. The numbers of reads per feature are then calculated to derive feature counts and infer expression levels. Finally, a statistical test is applied to normalized feature counts, followed by a collective assessment of significance based on an acceptable false discovery rate (FDR), to identify differentially expressed features with statistical significance. From this point on, we will simply refer to features as genes. While the use of RNA-Seq for quantifying gene expression is relatively straightforward to conceptualize, RNA-Seq experiments have considerable computational and statistical challenges. The massive quantities of short reads require ultra fast alignment programs that adequately address memory demands. The volume of data is also of concern if the end user desires systematic storage and management, as well as integration of data into third party software for additional analyses. Importantly, the combination of a large number of comparisons and small sample sizes causes more concern than usual about the power of the statistical test. We describe DESeq2, a simple pipeline with the appropriate statistical tests for studying genome-wide changes in gene expression. It is modular and flexible to allow the end user to use different alignment programs, easily change parameters, and use different statistical tests for analysis of differential gene expression and enriched gene ontology (GO) terms.

2. HIGH THROUGHPUT SEQUENCING

In earlier period, hybridization-based microarrays approach was mostly used for gene expression profiling and DE analysis. These technologies consist in an array of probes, whose sequences represent particular regions of the genes to be monitored. The sample under investigation is washed over the array, and RNAs are free to hybridize to the probes with a complementary sequence. A fluorescent is used to label the RNAs, so that image acquisition of the whole array enables the quantification of the expressed genes. Although widely these techniques have several limitations such as reliance on prior knowledge about the genome for probe design; possibility to monitor only some portions of the known genes and not the actual sequences of all transcribed RNAs; imperfect hybridization between quasi-complementary sequences; limited dynamic range due to background noise and signal saturation; need for normalization to compare data from different arrays. Lately, high-throughput sequencing technology has brought a revolution in differential gene expression analysis with several key advantages such as reconstruction of known and novel transcripts at single-base level; broad dynamic range, not limited by signal saturation; high levels of reproducibility. Sequencing has progressed far beyond the analysis of DNA sequences, and is now routinely used to analyze other biological components such as RNA and protein, as well as their interaction in complex networks. RNA-seq leverages on the sequencing framework to overcome the pure quantification task, enabling new applications, such as transcriptome profiling of non-model organisms, novel transcripts discovery, investigation of RNA editing and quantification of allele-specific gene expression. Currently, HT sequencing has been widely applied in multi-level application including sequence reading appliances e.g. Whole Genome Re-sequencing, Targeted Specific Sequencing, De-novo assembly as well as comparable counting based applications such as RNA-seq, CAGE, GRO-Seq, NET-seq, ChIP-seq, Ribo-Seq. In case of DE analysis, we mainly depend on HT seq counting based applications ChIP-Seq, RNA-seq. Typically, these reads are assigned to a class based on their mapping to a common region of the target genome, where each class represents a target transcript, in the case of RNA-Seq, or a binding region, in the case of ChIP-Seq. An important summary statistic is the number of reads in a class; for RNA-Seq, this read count can differ in biological conditions.

3. DIFFERENTIAL GENE EXPRESSION USING RNA SEQ

Differential expression is the assessment of differences in read counts of genes between two or more experimental conditions. Genes are differentially expressed if this difference is statistically significant. There are two samples from the same patient. One sample is from kidney tumor biopsy. The other sample is a biopsy from the patient's other kidney, which seems to be perfectly healthy tissue. Theoretically; we would expect that the two samples will have different amounts of certain messenger RNA transcripts. It would be interesting to see which transcripts from the tumor are being synthesized at a significantly higher or lower number in the tumor tissue compared to that in the healthy tissue. Nevertheless, differential Expression in RNAseq differs from Microarray and other HT Data. Microarray DE data are based on numerical intensity values while Quantitative Metabolome analysis is based on area of the peak generated by each metabolite in the sample. RNAseq is based on sequence read count distributions and provides richer information i.e. increased specificity and sensitivity for enhanced detection of differential gene expression. The transcriptome is the whole set of RNAs transcribed from the genes of a cell. Although RNAs are not the final products of the transcription–translation process, the study of gene expression and differential gene expression can unveil important aspects about the cell states under investigation. HT sequencing established RNA-seq as the preferred methodology for the study of gene expression. The RNAs in the sample of interest are initially fragmented and reverse-transcribed into complementary DNAs (cDNAs). The obtained cDNAs are then amplified and subjected to HT Seq. The millions of short reads generated can then be mapped on a reference genome and the number of reads aligned to each gene, called ‘counts’, gives a digital measure of gene expression levels



ANALYSIS STEPS

READ MAPPING

Find original read source within the reference genome or transcriptome.

COUNTS COMPUTATION

Estimate gene expression with “counts”, i.e. with the number of reads mapped on each gene.

	sample A1	sample A2	sample B1	sample B2
gene 1	8	10	100	200
gene 2	14	15	15	40
gene 3	33	40	35	70
...	...	...	...	...
gene N	100	120	105	220

COUNTS NORMALIZATION

Eliminate biases to make expression levels comparable between samples (e.g. different sequencing depths of samples A1 and B2) and within samples (e.g. different lengths of gene 1 and gene 2).

	sample A1	sample A2	sample B1	sample B2
gene 1	8	10	100	200
gene 2	14	15	115	40
gene 3	33	40	35	70
...	...	...	...	...
gene N	100	120	105	220

Tot. reads: 5 millions

Tot. reads: 10 millions

DIFFERENTIAL EXPRESSION ANALYSIS

Identify genes with statistically different expression levels in the compared conditions (e.g. A and B).

	sample A1	sample A2	sample B1	sample B2
gene 1	0.16	0.20	2.00	2.00
gene 2	0.28	0.30	0.30	0.40
gene 3	0.66	0.80	0.70	0.70
...	...	...	...	...
gene N	2.00	2.40	2.10	2.20

Figure1.1 DE analysis procedure.

Despite all these newsworthy features and apparently easy scheme of data analysis, RNA-seq studies produce large and complex data sets, whose interpretation is not straightforward. Nevertheless, if a well-annotated reference genome or transcriptome is

DOI: 10.31695/IJASRE.2018.32720

available and if the aim of an RNA-seq study is the detection of DE genes, a basic data processing pipeline consists in read mapping, counts computation, counts normalization and detection of differentially expressed genes (Figure 1). More sophisticated pipelines can be tailored on the specific need by considering the addition of pre- and post-processing modules to be used before and after read mapping.

4. TRADITIONAL FOLD CHANGE VS STATISTICAL PARAMETRIC TEST IN RNA SEQ

Fundamental to the task of DE gene expression data analysis is the need to identify genes whose patterns of expression differ according to phenotype or experimental condition. Gene expression measurements on different genes are not generally autonomous. Meaningful comparisons between samples can be made by considering the joint distribution of specific sets of genes considering their specific interactions and transcriptional controls. To ease the high dimensional exploration of differential gene expression taking into account their relationships, we can start with a gene-by-gene approach, ignoring the dependencies between genes. Fold-change criterion based analysis may give a hand but not allow the assessment of significance of expression differences in the presence of biological and experimental variation, which may differ from gene to gene. Let's say there are 50 read counts in control and 100 read counts in treatment for gene A. This means gene A is expressing twice in treatment as compared to control (100 divided by 50 =2) or fold change is 2. This works well for over expressed genes as the number directly corresponds to how many times a gene is over expressed. But when it is other way round (i.e, treatment 50, control 100), the value of fold change will be 0.5 (all under expressed genes will have values between 0 to 1, while over expressed genes will have values from 1 to infinity). To make this leveled, we use log2 for expressing the fold change. I.e, log2 of 2 is 1 and log2 of 0.5 is -1. Fold change can also be computed in unsupervised fashion, where we don't know the class labels (like case-control or type1-type2) of the samples. In that setting we can use mean expression of a gene as the base value and compute the fold change for that gene in each sample.

Calculating fold change directly can be misleading. Low counts can appear to have high fold changes while large counts are less sensitive. This is the main reason for using statistical tests to assess differential expression. Generally, one might look at various properties of the distributions of a gene’s expression levels under different conditions, though most often location parameters of these distributions, such as the mean or the median, are considered. One may distinguish between parametric tests, such as the t-test, and non-parametric tests, such as the Mann-Whitney test or permutation tests. Parametric tests usually have a higher power if the underlying model assumptions, such as normality in the case of the t test, are at least approximately fulfilled.

Data analyzing scale is much important for performing statistical analysis of DE data. Logarithmic scale may be used for symmetrically normalizing the distribution of replicated measurements per gene. A variance stabilizing transformation derived from an error model is advantageous for statistical tests that rely on variance homogeneity due to diminishing differences in variance between experimental conditions. Nevertheless, genetic variance due to specific biological reasons will remain untouched. One or two group t-test comparisons, multiple group ANOVA, and more general trend tests are all instances of linear models that are frequently used for assessing differential gene expression.

The approach of conducting a statistical test for each gene is popular, largely because it is relatively straightforward and a standard catalog of methods can be applied. However, a large number of hypothesis test needs to be carried out leading to a large number of falsely significant results. Multiple testing procedures allow one to assess the overall significance of the results of a family of hypothesis tests. They focus on specificity by controlling type I (false positive) error rates such as the family-wise error rate or the false discovery rate. Still, multiple hypothesis testing remains a problem, because an increase in specificity, as provided by p-value adjustment methods, is coupled with a loss of sensitivity, that is, a reduced chance of detecting true positives. The number of hypotheses to be tested can often be reasonably reduced by non-specific filtering procedures, discarding, e.g., genes with consistently low intensity values or low variance across the samples. This is especially relevant in the case of genome-wide arrays, as often only a minority of all genes will be expressed at all in the cell type under consideration.

Many DE experiments involve only few replicates per condition, which makes it difficult to estimate the gene-specific variances that are used, e.g., in the t-test. Different methods have been developed to exploit the variance information provided by the data of all genes; in the limma package, an Empirical Bayes approach is implemented that employs a global variance estimator, $s_0{}^2$ is computed on the basis of all genes variances. The resulting test statistic is a moderated t-statistic, where instead of the single-gene estimated variances $s_g{}^2$, a weighted average of $s_0{}^2$ and $s_g{}^2$ is used. Under certain distributional assumptions, this test statistic can be shown to follow a t-distribution under the null hypothesis with the degrees of freedom depending on the data.

5. RNA-SEQ ANALYSIS PROCEDURE

5.1. Read Mapping

The first computational step of the RNA-seq data analysis pipeline is read mapping: reads are aligned to a reference genome or transcriptome by identifying gene regions that match read sequences. So far, many alignment tools have been proposed. In all cases, the mapping process starts by building an index of either the reference genome or the reads, which is then used to quickly retrieve the set of positions in the reference sequence where the reads are more likely to align. Once this subset of possible mapping locations has been identified, alignment is performed in these candidate regions with slower and more sensitive algorithms. The available mapping tools can be divided into two main categories based on the methodology used to build the index: hash tables or Burrows–Wheeler transform (BWT).

The hash table is a common data structure for indexing complex data sets so to facilitate rapid string searching. Mapping tools can build hash tables either on the set of input reads or on the reference, considering all subsequences of a certain length *k*. (k-mers) contained in the considered sequences. In the hash table, the key of each entry is a k-mer, while the value is the list of all positions in the reference where the k-mer was found. The two solutions have different advantages and drawbacks. For instance, building hash tables of the reference requires constant memory, for a given reference and parameter set, regardless of the size of the input read data. Conversely, building hash tables of reads typically requires variable but smaller memory footprint, depending on the number and complexity of the read set. However, this latter solution may require longer processing time to scan the entire reference sequence when searching for hits, even if the input read set is small, and is not suited for parallelization. BWT is a reversible string rearrangement that encodes the genome into a more compact representation, leveraging on redundancy of repeated subsequences. Methods based on BWT create an index of the BWT, called ‘FM-index’, that can be used to perform fast string searching in a reduced domain of available subsequences, without scanning the whole genome. The combination of BWT and FM-index ensures both limited memory and space occupancy, but requires longer computational time for index construction than hash-based methods. However, since the index has to be constructed only once for a given reference and pre-computed indexes for several model genomes are already available, this aspect has minimum impact on the total computational time. Conversely, the strategy used to extend the first partial high-quality hits identified thanks to hash- or BWT-based indexes into full-read alignments has a major impact on algorithm performance. Usually, hash-based algorithms implement a ‘seed-and-extend’ approach leveraging on a bounded version of the Smith–Waterman (SW) algorithm. BWT-based solutions sample substrings of the reference using the FM-index and then accommodate inexact matches by tolerating some mismatches, up to a certain threshold. BWT implementations, which were developed for short (<50 nt) read alignment, impose very stringent constraints on inexact matches, which make them much faster than hash-based approaches, but less sensitive. As NGS technologies are producing increasingly longer reads (>100 nt), mapping tools are implementing hybrid solutions, which exploit the efficiency of BWT and ‘FM-index’ for seeding and then perform alignment extension with SW-like algorithms.

Unlike tools for genome-sequencing data mapping, algorithms developed for RNA-seq may have to handle ‘spliced reads’. Splicing is a post-transcriptional modification underwent by most of RNAs transcribed in eukaryotic organisms. During splicing, non-coding regions (introns) are removed and coding sequences (exons) are concatenated together. Although the order of exons is always preserved, some exons can be removed along with introns, giving rise to different RNAs. This process, called ‘alternative splicing’, enables to produce different protein isoforms starting from the same gene. Thus, RNAs in eukaryotes can give rise to spliced reads that span exon–exon junctions and that cannot be directly mapped onto the genome, where exons are separated by introns. To map these spliced reads back to the genome, algorithms for RNA-seq data analysis must handle spliced alignment (Figure 2A). Generally, simple gapped alignment is not sufficient to account for introns because they can span a wide range of lengths. To align spliced reads, many tools implement a two-step procedure: first, reads are mapped to the genome and used to identify putative exons; then, candidate exons are used to build all possible exon–exon junctions, which are considered for mapping the spliced reads that failed to map in the first step.

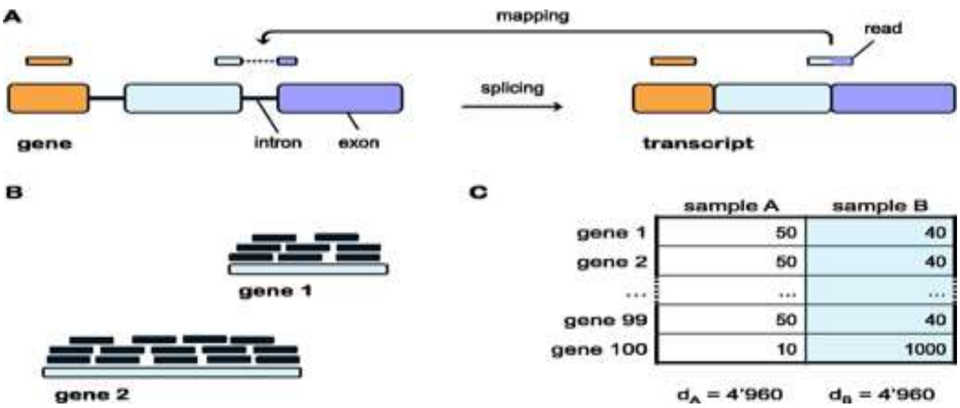


Figure 2: Read mapping foe DE analysis.

RNA-seq read mapping undoubtedly possesss some issues such as spliced-reads mapping in correspondence of exon–exon junctions, length bias, differences in library size composition. To be correctly mapped on the genome, where exons are separated by introns, spliced reads must be broken into shorter strings. Longer genes are more likely to generate more reads than shorter ones with similar expression levels. A count data set where samples A and B have the same number of reads ($d_B = d_A$), but different library compositions. The first 99 genes have the same counts in each sample (40 and 50 in sample A and B, respectively). In sample B, the reads available for most of the genes are 'consumed' by gene 100, which has very high expression. Library sizes can be computed excluding gene 100, to reflect the real sequencing state available for most of the genes in sample B ($d_B = 0.8d_A$).

5.2. Digitally Counting Gene Expression

After mapping, the reads aligned to each exon coding unit, transcript or gene expression level is estimated by computing counts. Counts computation considers the total number of reads overlapping the exons of a gene. Another strategy regards the whole length of a gene with the insertion of read counts from introns because in case of some well-annotated organisms, a fraction of reads may map outside the boundaries of known exons. After mapping, spliced reads can be used to model the abundance of different splicing isoforms of a gene. Genes with overlapping sequence can be dealt with the 'Union-Intersection gene' model considering the union of the exonic bases that do not overlap with the exons of other genes. We mainly focus on flexible 'maxcounts' approach which does not compute the sum of aligned reads, but estimates the expression level of each exon or single-isoform transcript as the maximum read coverage reached along its sequence. Combination 'maxcounts' at exon level with a measure of gene or transcript expression can be used for RNA-seq studies in eukaryotes. Gene expression from RNA-seq data is typically implemented through two computational steps: alignment of reads to a reference genome or transcriptome, and subsequent estimation of gene and isoform abundances based on aligned reads. The reads from RNA-Seq are generally much shorter than the transcripts from which they are sampled. As a consequence, in the presence of transcripts with similar sequences, it is not always possible to uniquely assign short reads to a specific gene.

NGS data arising from repeated regions have to be handled properly in order not to bias the results. A non-negligible fraction of RNA-seq reads are 'multireads' mapping with comparable fidelity on multiple positions of the reference. The fraction of multireads over total mapped reads depends on transcriptome complexity and read length, varying from 10 to >50%. Gene multireads can be handled by simply discarding them to estimate gene expression considering only uniquely mapping reads. Multireads filtering is a commonly used approach in the analysis of RNA-seq due to the chance of assigning multireads to the wrong genomic location. RNA-seq studies aims at reconstructing transcripts sequences and quantifying relative abundances. Ji et al. propose a sophisticated method called BM-Map to consider the mismatch profiles between the unique reads and the sequence of the genomic locations they are aligned and calculate the probability of mapping each multiread to a genomic location considering three sources of information: the sequencing error profile, the likelihood of true polymorphisms and the expression level of competing genomic locations. The mismatch profile is also taken into consideration by MMSEQ, which estimates both isoform expression and allelic imbalance, namely expression differences between different alleles of the same gene or isoform. More recent methods, such as RSEM, define a probabilistic model of RNA-Seq data and calculate maximum likelihood estimates of isoform expression levels using the Expectation-Maximization algorithm.

5.3. Preparing Count Matrices

The count-based statistical methods, such as DESeq2, edgeR, limma with the voom method, DSS, EBSeq and baySeq, expect input data from RNA-seq or HT-seq experiment as a matrix of integer values. The value in the i-th row and the j-th column of the matrix tells how many reads (or fragments, for paired-end RNA-seq) have been assigned to gene i in sample. In statistical modeling, matrix values should be counts of sequencing reads/fragments as only counts allow precise measurement. Pre-normalized counts should not be provided because the statistical model is the most powerful when applied to un-normalized counts, and is designed for diffrent library sizes. This involves several steps.

Counting RNA-seq fragments employs a newer and faster alternative pipeline to use transcript abundance quantification methods such as Salmon, Sailfish, kallisto, or RSEM, to estimate abundances without aligning reads, followed by the tximport package for assembling estimated count and offset matrices for using Bioconductor DE packages. Salmon software can be used for quantifying transcript abundance and using the steps in the tximport vignette one can build DESeqDataSet. The advantages of using the transcript abundance quantifiers in conjunction with tximport to produce gene-level count matrices and normalizing offsets, are: this approach corrects for any potential changes in gene length across samples (e.g. from differential isoform usage); some of these methods are substantially faster and require less memory and disk usage compared to alignment-based methods; and it is possible to avoid discarding those fragments that can align to multiple genes with homologous sequence. Transcript abundance quantifiers skip the generation of large files which store read alignments, instead producing smaller files which store estimated abundances, counts, and effective lengths per transcript.

Aligning reads to a reference genome starts with the computational analysis of an RNA-seq FASTQ files containing nt-seq of each read and a quality score at each position. These reads must first be aligned to a reference genome or transcriptome, or the abundances and estimated counts per transcript can be estimated without alignment using packages. In either case, it is important to know if the sequencing experiment was single-end or paired-end, as the alignment software will require the user to specify both FASTQ files for a paired-end experiment. The output of this alignment step is commonly stored in a file format called SAM/BAM. A number of software programs exist to align reads to a reference genome. We can use the STAR read aligner to align the reads for our experiment to the Ensembl reference genome. A file's each line will contain an identifier for each experiment, and we should have all the FASTQ files in a subdirectory. If you have downloaded the FASTQ files from the Sequence Read Archive, the identifiers would be SRA run IDs, e.g. SRR1039520. One should have two files for a paired-end experiment for each ID, fastq/SRR1039520_1.fastq1 and fastq/SRR1039520_2.fastq, which gives the first and second read for the paired-end fragments. If one has performed a single-end experiment, one would only have one file per ID. We have also created a subdirectory, aligned, where STAR will output its alignment files. SAMtools can be used to generate BAM files. The BAM files for a number of sequencing runs can then be used to generate count matrices.

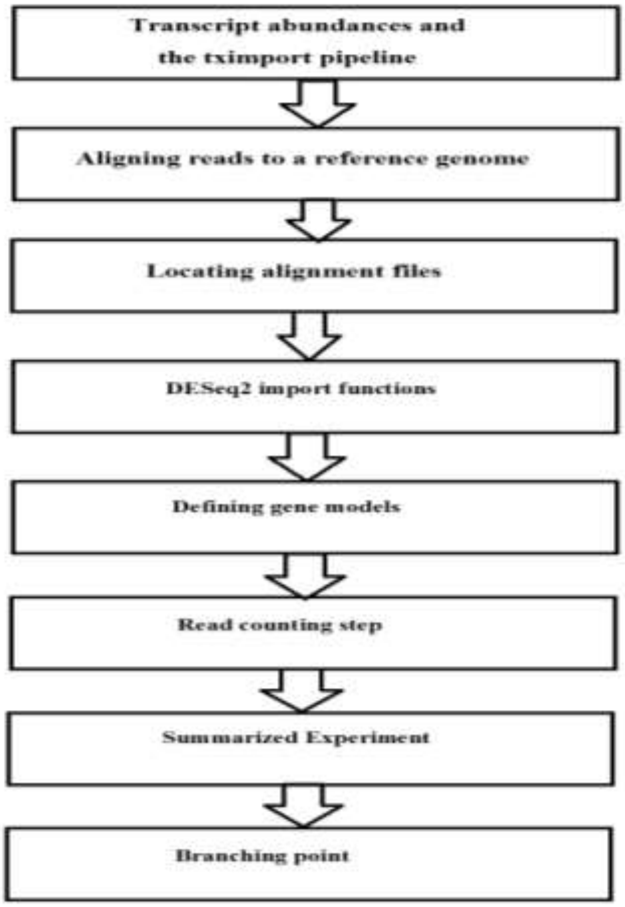


Figure 3: Steps in counting matrix preparation.

After successful alignment we have to locate the alignment files. The specific package will contain the count matrix and files with a small subset of the total number of reads in the experiment because the full alignment files are large (a few gigabytes each) which may take between 10-30 minutes to count the fragments for each sample. We will use these files to construct count matrix from BAM files. Afterwards, we will load the full count matrix corresponding to all samples and continue the analysis with that full matrix. The R function system.file can be used to find out where on computer the files from a package have been installed. In the particular directory, we may find the BAM files. Typically, we have a table with detailed information for each of our samples that links samples to the associated FASTQ and BAM files. For project help, one can create such a comma-separated value (CSV) file using a text editor or spreadsheet software such as Excel and load such a CSV. Once reads have been aligned, there are a number of tools that can be used to count the number of reads/fragments that can be assigned to genomic features for each sample. These often take as input SAM/BAM alignment files and a file specifying the genomic features, e.g. a GFF3 or GTF file specifying the gene models.

DESeq2 has several import functions which can be used generate count matrices such as summarize Overlaps, feature Counts, tximport, htseq-count.

Function	Package	Framework	Output	Deseq2 Input Function
Summarize overlaps	Genomical ignments	R/Bioconductor	Summarized experiment	Deseq dataset
Feature counts	Rsubread	R/Bioconductor	Matrix	Deseq dataset frommatrix
Txim port	Tximport	R/Bioconductor	List Of Matrices	Deseq dataset fromtximport
Htseq-Count	Htseq	Python	Files	Deseq dataset tfromhtseq

Among these we can easily proceed using summarizeOverlaps. Via the Run column in the sample table, we construct the full paths to the files we want to perform the counting operation on. We indicate in Bioconductor that these files are BAM files using the BamFileList function from the Rsamtools package that provides an R interface to BAM files. The names of the genomic features in the annotation should be consistent with the names of the reference used for read alignment. Otherwise, the scripts might fail to count any reads to features due to the mismatching names.

Gene models should be built after generating count matrix. The model will be used for counting reads/fragments. We can read the gene model from an Ensembl GTF file using makeTxDbFromGFF from the GenomicFeatures package. GTF files can be downloaded from Ensembl's FTP site or other gene model repositories. A TxDb object is a database that can be used to generate a variety of range-based objects, such as exons, transcripts, and genes. We want to make a list of exons grouped by gene for counting read/fragments. There are other options for constructing a TxDb. For the known genes track from the UCSC Genome Browser, one can use the pre-built Transcript DataBase. If the annotation file is accessible from AnnotationHub, a pre-scanned GTF file can be imported using makeTxDbFromGRanges.

After all these preparations, the actual read counting is easy. The function summarizeOverlaps from the GenomicAlignments package can do this. This produces a Summarized Experiment object that contains a variety of information about the experiment, and will be described in more detail below. If it is desired to perform counting using multiple cores, one can use the register and MulticoreParam or SnowParam functions from the BiocParallel package before the counting call. We specify a number of arguments besides the features and the reads. The mode argument describes what kind of read overlaps will be counted. Fragments will be counted only once to each gene, even if they overlap multiple exons of a gene which may themselves be overlapping. In order to produce correct counts, it is important to know if the RNA-seq experiment was strand-specific or not. However, certain strand-specific protocols could have the reads align only to the opposite strand of the genes. The user must check if the experiment was strand-specific and if so, whether the reads should align to the forward or reverse strand of the genes. For various counting/quantifying tools, one specifies counting on the forward or reverse strand in different ways, although this task is currently easiest with htseq-count, featureCounts, or the transcript abundance quantifiers mentioned previously. It is always a good idea to check the column sums of the count matrix to make sure these totals match the expected of the number of reads or fragments aligning to genes. Additionally, one can visually check the read alignments using a genome visualization tool.

The main component parts of a Summarized Experiment object are assay which contains the matrix of counts, rowRanges contains information about the genomic ranges and colData contains information about the samples. A single "counts" matrix that contains the fragment counts for each gene and sample is stored in assay. The rowRanges is the GRangesList used for counting and one GRanges of exons is created for each row of the count matrix. The component parts of the SummarizedExperiment are accessed with an R function of the same name, rowRanges and colData. We can still investigate the resulting SummarizedExperiment by looking at the counts in the assay slot, the phenotypic data about the samples in colData slot, and the data about the genes in the rowRanges slot. The rowRanges, when printed, only shows the first GRanges and contains metadata about the construction of the gene model in the metadata slot. The colData slot, so far empty, should contain all the metadata. Because we used a column of sampleTable to produce the bamfiles vector, we know the columns of se are in the same order as the rows of sampleTable. We can assign the sample Table as the colData of the summarized experiment, by converting it into a DataFrame and using the assignment function.

5.4. Count Normalization and DE Analysis

After we have counted the fragments which overlap the genes in the gene model, we come to branching point where we could use a variety of Bioconductor packages for exploration and differential expression of the count data, including edgeR, limma with the voom method, DSS, EBSeq and baySeq. Compared performance of different statistical methods for RNA-seq using a large number of biological replicates can help users to decide which tools make sense to use, and how many biological replicates are necessary to obtain certain sensitivity. The Summarized Experiment object is all we need to create the data object used by DESeq2.

5.4.1. The DE-Seq Dataset Object, Sample information and the Design Formula

Software packages has general data classes (such as the Summarized Experiment) that can be used to move data between packages and use a custom class for storing data to ensure the provision of needed data slots according to the requirements. Summarized Experiment automatically subsets or reorders the associated rowRanges and colData to prevent accidental sample swaps. In DESeq2, the custom class is called DESeq DataSet which is built on top of the Summarized Experiment class, and easy to convert this class objects into DESeq DataSet objects which has an associated design formula. The experimental design is specified at the beginning of the analysis, as it will inform many of the DESeq2 functions how to treat the samples in the analysis. The design formula tells which columns in the sample information table specify the experimental design and how these factors should be used in the analysis. The simplest design formula for differential expression would be the condition in that specifies which of two or more groups the samples belongs to.

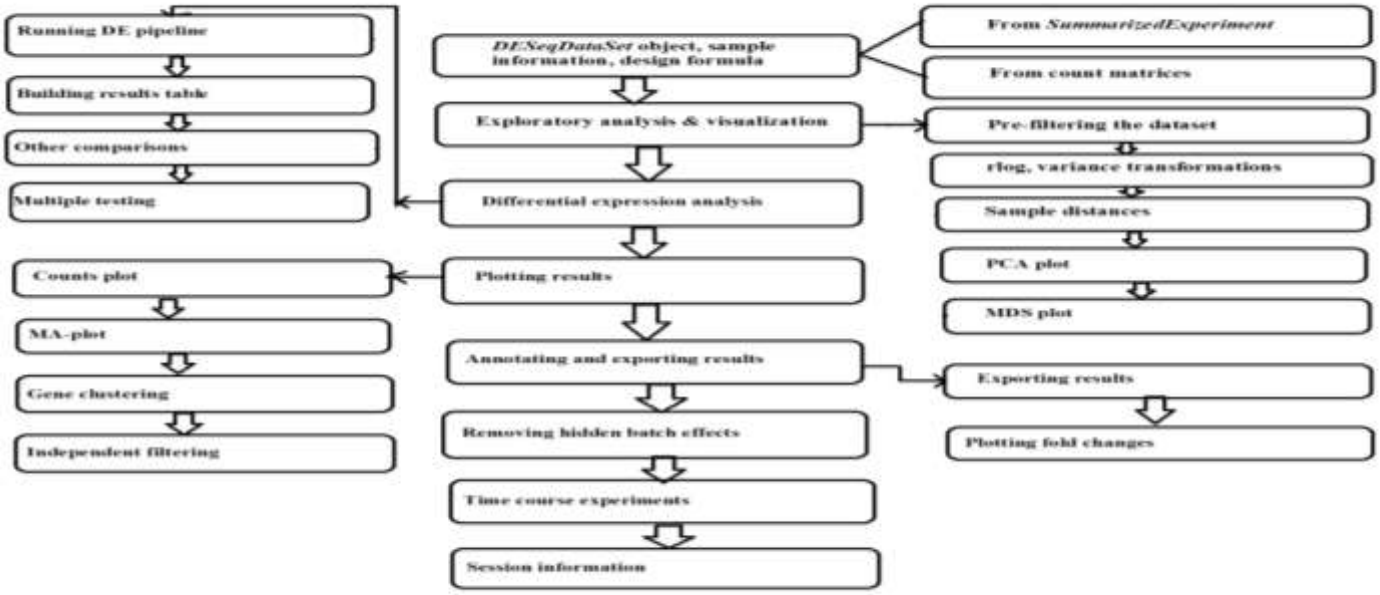


Figure 4: After counting, normalization and DE analysis steps.

Software packages has general data classes (such as the SummarizedExperiment) that can be used to move data between packages and use a custom class for storing data to ensure the provision of needed data slots according to the requirements. SummarizedExperiment automatically subsets or reorders the associated rowRanges and colData to prevent accidental sample swaps. In DESeq2, the custom class is called DESeqDataSet which is built on top of the SummarizedExperiment class, and easy to convert this class objects into DESeqDataSet objects which has an associated design formula. The experimental design is specified at the beginning of the analysis, as it will inform many of the DESeq2 functions how to treat the samples in the analysis. The design formula tells which columns in the sample information table specify the experimental design and how these factors should be used in the analysis. The simplest design formula for differential expression would be the condition in that specifies which of two or more groups the samples belongs to.

The construction of the DESeq DataSet from a prepared Summarized Experiment from the publicly available sequencing data files. We can quickly check the millions of fragments that uniquely aligned to the genes. After constructing a Summarized Experiment, we need to make sure that the object contains all the necessary information about the samples. When working we will have to add the pertinent sample / phenotypic information using comma-separated value (CSV) or tab-separated value (TSV) file exported from an Excel spreadsheet at this stage. Once we have our fully annotated Summarized Experiment object, we can construct a DESeq DataSet object from it that will then form the starting point of the analysis. Provided that we only have a count matrix and a table of sample information, we will show how can build a DESeq DataSet but we can skip the step if we have prepared a Summarized Experiment. The count matrix would either be read in from a file or perhaps generated by an R function like feature Counts from the. The information in a Summarized Experiment object can be accessed with accessor functions. In count matrix, each row represents an Ensembl gene, each column a sequenced RNA library, and the values give the raw numbers of fragments that were uniquely assigned to the respective gene in each library. We should prepare our data object in a form that is suitable for analysis count data table with the fragment counts and coldata table with information about the samples.

DOI: 10.31695/IJASRE.2018.32720

International Journal of
Advances in Scientific Research and Engineering (ijasre)

E-ISSN : 2454-8006

DOI: http://doi.org/10.31695/IJASRE.2018.32720

Volume 4, Issue 7
July - 2018

5.4.2. Exploratory Analysis and Visualization

Transformations of the counts are used to visually explore sample relationships. Original raw counts should be used for statistical testing because the methods rely on original count data not scaled or transformed. Pre-filtering the dataset means DESeqDataSet count matrix must be reduced to increase the speed of our functions by removing the rows that have no or nearly no information about the amount of gene expression. So, we should apply the most minimal filtering rule and additional weighting/filtering can improve power at a later step in the workflow. Clustering and principal components analysis (PCA), work best for data that generally has the same range of variance at different ranges of the mean values. When the expected amount of variance is approximately the same across different mean values, the data is said to be homoskedastic. For RNA-seq counts, however, the expected variance grows with the mean. If one performs PCA directly on a matrix of counts or normalized counts by taking logarithm of the normalized count values plus a pseudocount of 1; the genes with the very lowest counts will contribute a great deal of noise to the resulting plot, because taking the logarithm of small counts actually inflates their variance. The low count genes with low signal-to-noise ratio will overly contribute to sample-sample distances and PCA plots.

A useful first step in an RNA-seq analysis is often to assess overall similarity between samples: R function dist can be used to calculate the Euclidean distance between samples. To ensure we have a roughly equal contribution from all genes, we use it on the rlog-transformed data. We need to transpose the matrix of values using t, because the dist function expects the different samples to be rows of its argument, and different genes to be columns. We visualize the distances in a heatmap using the function pheatmap from the pheatmap package. In order to plot the sample distance matrix with the rows/columns arranged by the distances in our distance matrix, we manually provide sampleDists to the clustering_distance argument of the pheatmap function. Another option for calculating sample distances is to use the Poisson Distance and it takes the original count matrix (not normalized) with samples as rows instead of columns, so we need to transpose the counts.

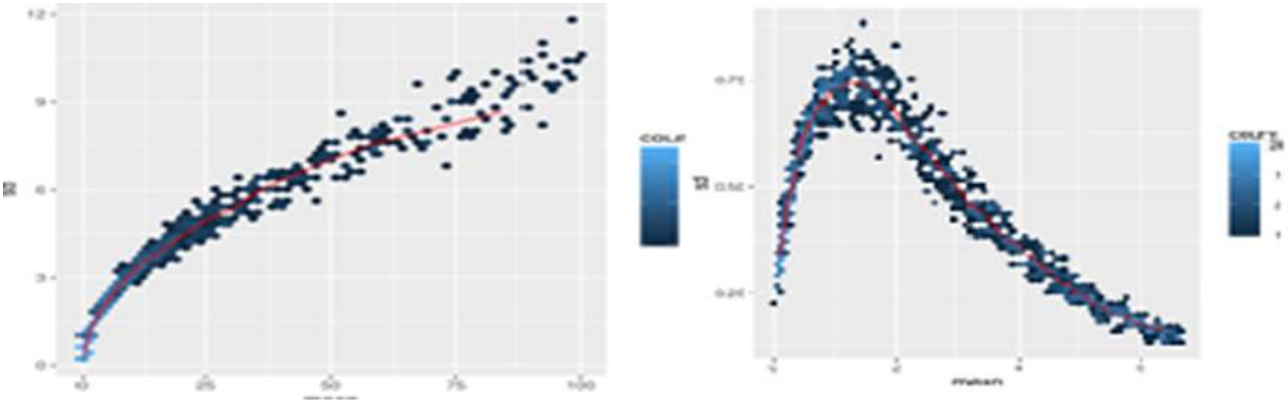


Figure 1

DESeq2 offers two transformations for count data to stabilize the variance across the mean: the regularized-logarithm transformation or rlog, and the variance stabilizing transformation (VST) for negative binomial data with a dispersion-mean. For genes with high counts, the rlog and VST will give similar result to the ordinary log2 fold transformation of normalized counts. The rlog-transformed or VST data then becomes approximately homoscedastic for lower counts, and can be used directly for computing distances between samples, making PCA plots, or as input to downstream methods which perform best with homoskedastic data. The rlog tends to work well on small datasets (n < 30), sometimes outperforming the VST when there is a large range of sequencing depth across samples. The VST is much faster to compute and is less sensitive to high count outliers than the rlog and recommended for large dataset .

A useful first step in an RNA-seq analysis is often to assess overall similarity between samples by using R function dist to calculate the Euclidean distance between samples. To ensure we have a roughly equal contribution from all genes, we use it on the rlog-transformed data. We need to transpose the matrix of values using t, because the dist function expects the different samples to be rows of its argument, and different dimensions to be columns. Besides heatmap, principal components analysis (PCA) can be used to visualize sample-to-sample distances. In this ordination method, the data points are projected onto the 2D plane such that they spread out in the two directions that explain most of the differences. The x-axis is the direction that separates the data points the most. The values of the samples in this direction are written PC1. The y-axis is a direction (it must be orthogonal to the first direction) that separates the data the second most. The values of the samples in this direction are written PC2. The percent of the total variance that is contained in the direction is printed in the axis label. These percentages do not add to 100%, because there are more dimensions that contain the remaining variance.

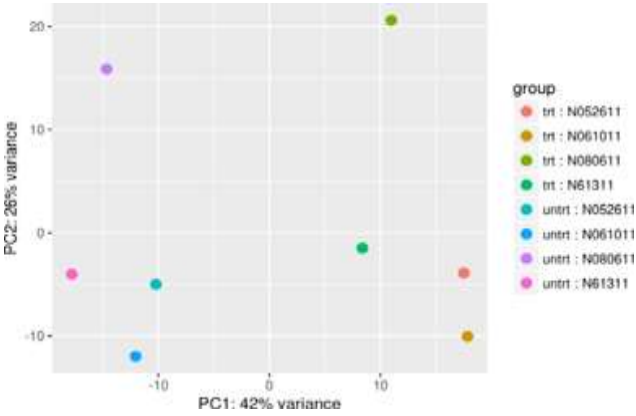


Figure 2 PCA Plot for showing distance

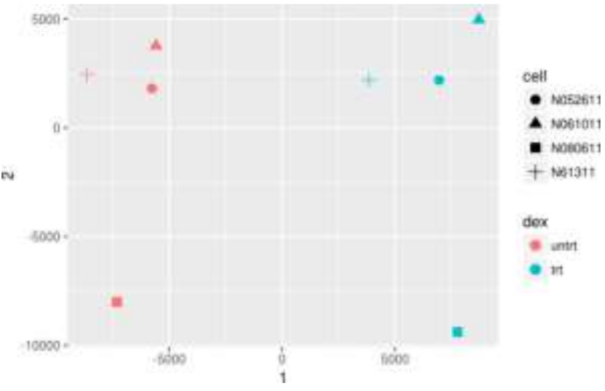


Figure 3: MDS plot using poissions distance

Function plotPCA is included in from DESeq2 or it can be used from scratch using the ggplot2 package by asking the plotPCA function to return the data used for plotting rather than building the plot. From the PCA plot, we see that the differences between features are considerable, and that's why it will be important to account for this in differential testing. Another plot, very similar to the PCA plot, can be made using the multidimensional scaling (MDS) function in base R. This is useful when we don't have a matrix of data, but only a matrix of distances. Here we compute the MDS for the distances calculated from the rlog transformed counts and plot these.

5.5 Differential expression analysis

Running the differential expression pipeline starts from DESeq DataSet raw counts with a single call to the function DESeq that prints out a message for the various steps it performs such as the estimation of size factors, the estimation of dispersion values for each gene, and fitting a generalized linear model. DESeq DataSet contains all the fitted parameters within it and we can extract out results tables of interest from this object. Calling results without any arguments will extract the estimated log2 fold changes and p values in the design formula. log2FoldChange is the effect size estimate and tell us how much the gene's expression seems in comparison to samples. This value is reported on a logarithmic scale to base 2 but has an uncertainty associated called standard error estimate for the log2 fold change. The purpose of a test for differential expression is to test whether the data provides sufficient evidence to conclude that this value is really different from zero. DESeq2 performs for each gene a hypothesis test to see whether evidence is sufficient to decide against the null hypothesis that there is zero effect of the treatment on the gene and that the observed difference between treatment and control was merely caused by experimental variability. As usual in statistics, the result of this test is reported as a p value, and it is found in the column p value that indicates the probability that a fold change as strong as the observed one, or even stronger, would be seen under the situation described by the null hypothesis. There are two ways to be stricter about which set of genes are considered significant by lowering the false discovery rate threshold, raise the log2 fold change threshold from 0. Sometimes a subset of the p values in result will be "not available" reporting that all counts for this gene were zero, and hence no test was applied. In addition, p values can be assigned NA if the gene was excluded from analysis because it contained an extreme count outlier.

Results for a comparison of any two levels of a variable can be extracted using the contrast argument to results. The user should specify three values: the name of the variable, the name of the level for the numerator, and the name of the level for the denominator. There are additional ways to build results tables for certain comparisons after running DESeq once. If results for an interaction term are desired, the name argument of results should be used

HT-seq, avoids using direct use of p values as evidence against the null, but to correct for multiple testing. Assuming the null hypothesis is true for all genes, with the definition of the p value, we expect up to 5% of the genes to have a p value below 0.05. DESeq2 uses the Benjamini-Hochberg (BH) adjustment as implemented in the base R p.adjust function; in brief, this method calculates for each gene an adjusted p value that answers the following question: if one called significant all genes with an adjusted p value less than or equal to this gene's adjusted p value threshold, what would be the fraction of false positives (the false discovery rate, FDR) among them. The FDR is a useful statistic for many high-throughput experiments, as we are often interested in reporting or focusing on a set of interesting genes, and we would like to put an upper bound on the percent of false positives in this set.

5.4.3. Exploratory Analysis and Visualization

Running the differential expression pipeline starts from DESeq DataSet raw counts with a single call to the function DESeq that prints out a message for the various steps it performs such as the estimation of size factors, the estimation of dispersion values for each gene, and fitting a generalized linear model. DESeq DataSet contains all the fitted parameters within it and we can extract out results tables of interest from this object. Calling results without any arguments will extract the estimated log2 fold changes and p values in the design formula. log2 Fold Change is the effect size estimate and tell us how much the gene's expression seems in comparison to samples. This value is reported on a logarithmic scale to base 2 but has an uncertainty associated called standard error estimate for the log2 fold change. The purpose of a test for differential expression is to test whether the data provides sufficient evidence to conclude that this value is really different from zero. DESeq2 performs for each gene a hypothesis test to see whether evidence is sufficient to decide against the null hypothesis that there is zero effect of the treatment on the gene and that the observed difference between treatment and control was merely caused by experimental variability. As usual in statistics, the result of this test is reported as a p value, and it is found in the column p value that indicates the probability that a fold change as strong as the observed one, or even stronger, would be seen under the situation described by the null hypothesis. There are two ways to be stricter about which set of genes are considered significant by lowering the false discovery rate threshold, raise the log2 fold change threshold from 0. Sometimes a subset of the p values in result will be “not available” reporting that all counts for this gene were zero, and hence no test was applied. In addition, p values can be assigned NA if the gene was excluded from analysis because it contained an extreme count outlier.

Results for a comparison of any two levels of a variable can be extracted using the contrast argument to results. The user should specify three values: the name of the variable, the name of the level for the numerator, and the name of the level for the denominator. There are additional ways to build results tables for certain comparisons after running DESeq once. If results for an interaction term are desired, the name argument of results should be used

HT-seq, avoids using direct use of p values as evidence against the null, but to correct for multiple testing. Assuming the null hypothesis is true for all genes, with the definition of the p value, we expect up to 5% of the genes to have a p value below 0.05. DESeq2 uses the Benjamini-Hochberg (BH) adjustment as implemented in the base R p.adjust function; in brief, this method calculates for each gene an adjusted p value that answers the following question: if one called significant all genes with an adjusted p value less than or equal to this gene's adjusted p value threshold, what would be the fraction of false positives (the false discovery rate, FDR) among them. The FDR is a useful statistic for many high-throughput experiments, as we are often interested in reporting or focusing on a set of interesting genes, and we would like to put an upper bound on the percent of false positives in this set.

5.4.3. Plotting results

Assuming the null Counts plot is a quick way to visualize the counts for a particular gene and it uses the plotCounts function that takes as arguments the DESeqDataSet, a gene name, and the group over which to plot the counts. We can also make custom plots using the ggplot function from the ggplot2 package.

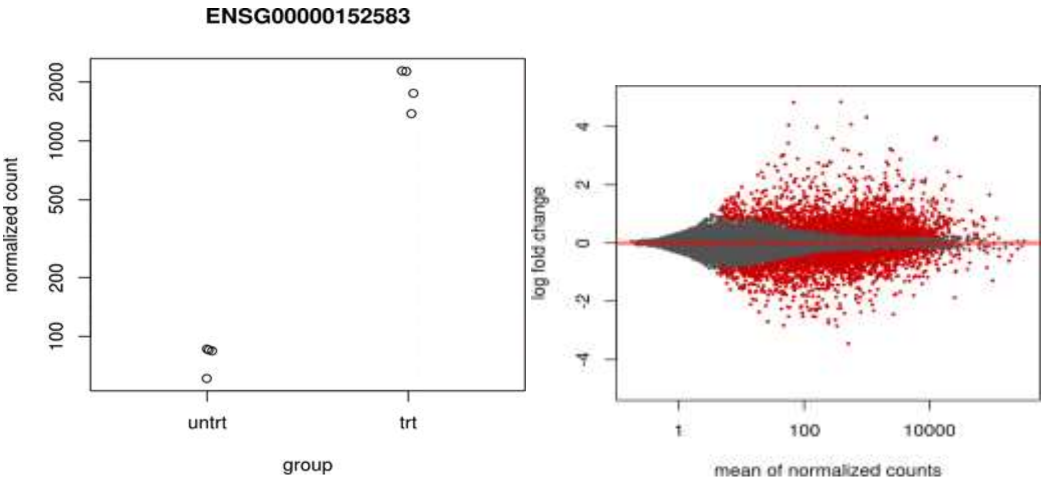
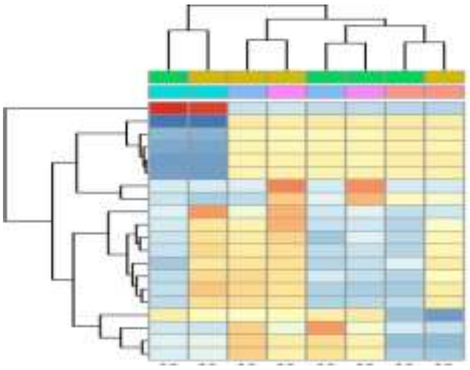


Figure 4. Count Plot **MA Plot**

Besides, MA-plot mean-difference plot, or a Bland-Altman plot provides a useful overview for the comparisons of interest, across all genes. On the y-axis, the “M” stands for “minus” – subtraction of log values is equivalent to the log of the ratio – and on the x-axis, the “A” stands for “average”. Another useful diagnostic plot is the histogram of the p values which is best formed by excluding genes with very small counts, which otherwise generate spikes in the histogram.

In the sample distance heatmap, the dendrogram at the side shows us a hierarchical clustering of the samples. Such a clustering can also be performed for the genes. Since the clustering is only relevant for genes that actually carry a signal, one usually would only cluster a subset of the most highly variable genes. The heatmap becomes more interesting if we do not look at absolute expression strength but rather at the amount by which each gene deviates in a specific sample from the gene’s average across all samples. Hence, we center each genes’ values across samples, and plot a heatmap.



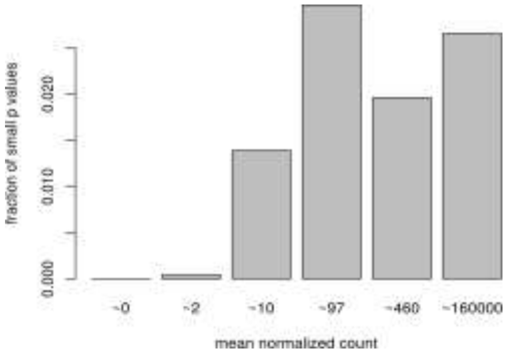


Figure 5: Gene Clustering **Figure 6: Independent Flitering**

Such a clustering can The MA plot highlights an important property of RNA-seq data. For weakly expressed genes, we have no chance of seeing differential expression, because the low read counts suffer from such high Poisson noise that any biological effect is drowned in the uncertainties from the sampling at a low rate. We can also show this by examining the ratio of small p values (say, less than 0.05) for genes binned by mean normalized count. We will use the results table subjected to the threshold to show what this looks like in a case when there are few tests with small p value. The ratio of small p values for genes binned by mean normalized count. The p values are from a test of log2 fold change greater than 1 or less than -1. This plot demonstrates that genes with very low mean count have little or no power, and are best excluded from testing. At first sight, there may seem to be little benefit in filtering out these genes. After all, the test found them to be non-significant anyway. However, these genes have an influence on the multiple testing adjustment, whose performance improves if such genes are removed. By removing the low count genes from the input to the FDR procedure, we can find more genes to be significant among those that we keep, and so improved the power of our test. This approach is known as independent filtering. The DESeq2 software automatically performs independent filtering that maximizes the number of genes with adjusted p value less than a critical value 0.1. Filtering is permissible only if the statistic that we filter on is independent of the actual test statistic, the p value under the null hypothesis. Otherwise, the filtering would invalidate the test and consequently the assumptions of the BH procedure. The independent filtering software used inside DESeq2 comes from the genefilter package

5.4.4. Annotating and exporting results

Result table so far only contains the Ensembl gene IDs, but alternative gene names may be more informative for interpretation. Bioconductor's annotation packages help with mapping various ID schemes to each other. We load the AnnotationDbi package and the annotation package org.Hs.eg.db. We can use the mapIds function to add individual columns to our results table. We can easily save the results table in a CSV file that can be shared or loaded with a spreadsheet program such as Excel. The call to as.data.frame is necessary to convert the DataFrame object from IRanges package to a data.frame object that can be processed by write.csv. A more sophisticated way for exporting results the Bioconductor package ReportingTools which automatically generates dynamic HTML documents, including links to external databases using gene identifiers and boxplots summarizing the normalized counts across groups.

If we have used the summarizeOverlaps function to count the reads, then our DESeqDataSet object is built on top of ready-to-use Bioconductor objects specifying the genomic coordinates of the genes. We can therefore easily plot our differential expression results in genomic space. While the results function by default returns a DataFrame, using the format argument, we can ask for GRanges or GRangesList output. We will use the Gviz package for plotting the GRanges and associated metadata.

5.4.5. Removing hidden batch effects

We can use statistical methods designed for RNA-seq from the sva package to detect groupings of the samples, and then we can add these to the DESeqDataSet design, in order to account for batch effect. The SVA package uses the term surrogate variables for the estimated variables that we want to account for in our analysis. Another package for detecting hidden batches is the RUVSeq package, with the acronym "Remove Unwanted Variation". We obtain a matrix of normalized counts for which the average count across samples is larger than 1and try to recover any hidden batch effects, supposing that we do not know basic information. So we use a full model matrix and a reduced, or null, model matrix with only an intercept term. Finally we specify that we want to estimate 2 surrogate variables.

5.4.6. Time course experiments and Session information

DESeq2 can be used to analyze time course experiments, for example to find those genes that react in a condition-specific manner over time, compared to a set of baseline samples. We use a design formula that models the difference at time 0, the difference over time, and any differences over time. A likelihood ratio test should be done where we remove specific differences over time. Genes with small p values from this test are those which at one or more time points after time 0 showed a specific effect. This is just one of the tests that can be applied to time series data. Another option would be to model the counts as a smooth function of time, and to include an interaction term of the condition with the smooth function. It is possible to build such a model using spline basis functions within R, and another, more modern approach is using Gaussian processes. We can plot the counts for the groups over time using ggplot2, for the gene with the smallest adjusted p value, testing for condition-dependent time profile and accounting for differences at time 0. As the last part of RNA-seq DE analysis, we call the function session Info, which reports the version numbers of R and all the packages used in this session. It is good practice to always keep such a record of this as it will help to track down what has happened in case an R script ceases to work or gives different results because the functions have been changed in a newer version of one of your packages. By including it at the bottom of a script, reports will become more reproducible.

6. CONCLUSION

In this study we performed a detailed comparative analysis of a number of methods for differential expression analysis from RNA-seq data. For the various methods, our comparison focused on the performance of the normalization, control of false positives, effect of sequencing depth and replication, and on the subset of gene expressed exclusively in one condition. RNA seq provides with a richer information, whereas microarray provides only probe specific information. We should not calculate fold change directly for RNA seq Data. DEseq2 uses the negative binomial distribution but other distributions can be used. CuffDiff2 seems to work worse than others. Possibly becauseit has extra statistics to deal with isoform. Limma, DESeq and EdgeR are pretty similar. Biological replicates are better than more depth so when dealing with large datasets it is very important to adjust the p-values to avoid type 1 errors. In contrast to other approaches, which rely on simulated data generated by specific statistical distribution or limited experimental datasets. Overall, no single method emerged as favorable in all comparisons but it is apparent that methods based on negative binomial modeling have improved specificity and sensitivities as well as good control of false positive errors with comparable performance.

REFERENCES

[1] Michael I., Wolfgang Huber, and Simon Anders. "Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2." Genome biology 15.12 (2014): 550.

[2] "MAnorm: a robust model for quantitative comparison of ChIP-Seq data sets." Genome biology 13.3 (2012): R16.

[4] "limma powers differential expression analyses for RNA-sequencing and microarray studies." Nucleic acids research 43.7 (2015): e47-e47.

[5] Ignatiadis, Nikolaos, et al. "Data-driven hypothesis weighting increases detection power in genome-scale multiple testing." Nature methods 13.7 (2016): 577.

[6] Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface." *IEEE Transl. J. Magn. Japan*, August 1987, vol. 2, pp.740-741. Leek, Jeffrey T., and John D. Storey. "Capturing heterogeneity in gene expression studies by surrogate variable analysis." PLoS genetics 3.9 (2007): e161.

[7] Leek, Jeffrey T., and John D. Storey. "A general framework for multiple testing dependence." Proceedings of the National Academy of Sciences 105.48 (2008): 18718-18723.

[8] Anders, Simon, and Wolfgang Huber. "Differential expression analysis for sequence count data." Genome biology 11.10 (2010): R106.

[9] Scholtens, D., and A. Von Heydebreck. "Analysis of differential gene expression studies." Bioinformatics and computational biology solutions using R and Bioconductor. Springer, New York, NY, 2005. 229-248.

[10] Yin, Tengfei, et al. "Visual Mining Methods for RNA-Seq Data: Data Structure, Dispersion Estimation and Significance Testing." Journal of Data Mining in Genomics & Proteomics 4.139 (2013): 2153-0602.