International Journal of Foundations of Computer Science

Transcriptomics: Quantifying non-uniform read distribution using MapReduce

Jamie J. Alnasir
jamie.alnasir.2013@rhul.ac.uk

Hugh P. Shanahan

Centre for Systems and Synthetic Biology
Department of Computer Science, Royal Holloway University of London,
Egham, Surrey TW20 0EX, United Kingdom

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

RNA-seq is a high-throughput Next-sequencing technique for estimating the concentration of all transcripts in a transcriptome. The method involves complex preparatory and post-processing steps which can introduce bias, and the technique produces a large amount of data [**?** **?**]. Two important challenges in processing RNA-seq data are therefore the ability to process a vast amount of data, and methods to quantify the bias in public RNA-seq datasets. We describe a novel analysis method, based on analysing sequence motif correlations, that employs MapReduce on Apache Spark to quantify bias in Next-generation sequencing (NGS) data at the deep exon level. Our implementation is designed specifically for processing large datasets and allows for scalability and deployment on cloud service providers offering MapReduce. In investigating the wild and mutant organism types in the species *D. melanogaster* we have found that motifs with runs of Gs (or their complement) exhibit low motif-pair correlations in comparison with other motif-pairs. This is independent of the mean exon GC content in the wild type data, but there is a mild dependence in the mutant data. Hence, whilst both datasets show the same trends, there is however significant variation between the two samples.

Keywords: rna-seq; mapreduce; transcriptomics; 4-mers; motif analysis; drosophila.

1. Background

High-throughput sequencing methods have developed over the last three decades from semi-automated methods to massively-parallel next-generation sequencing [29]. In particular, RNA-seq on which this article is focused, is an NGS technique for estimating the concentration of all transcripts in a transcriptome. The transcriptome of an organism is defined as the sum total of all the messenger RNA (mRNA) molecules expressed and is therefore highly dynamic, complex and in a constant state of flux. It accommodates the cells constantly changing requirements - a result of intra- and extra-cellular stimuli as well as disease pathology. RNA-Seq has revolutionised the field of transcriptomics and has transformed our view of the

extent and complexity of the transcriptome through deep-sequencing and also as a result of the increased precision the technique offers over other methods [37].

RNA-seq provides wide coverage of the transcriptome as it involves the direct sequencing of transcripts of RNA found in the sample [37, 11]. RNA-seq can therefore be used to study various types of RNA present: total RNA, pre-mRNA, and noncoding RNA (ncRNA), such as microRNA and long ncRNA enabling it to be used to study alternative splicing events [24, 16]. Furthermore RNA-seq achieves this at a higher resolution [11] than other technologies.

Recent developments in the RNA-seq workflow, from sample preparation to sequencing have furthered our understanding of the transcriptome but have also required substantial effort for data analysis and computation. Given the complexity of RNA-seq workflow this necessitates study of the bias that can be introduced in the preparatory steps [2, 11, 25]. Characterisation and quantification of bias in RNA-seq is especially incumbent given that the method sequences and measures the transcriptome indirectly using reverse transcribed complementary DNA (cDNA) [22].

Much of the raw sequence read data from RNA-seq are being deposited in public repositories such as the Sequence Read Archive (SRA) [13] and Gene Expression Omnibus (GEO) [7]. In just a year (from 2010 to July 2011) the amount of data deposited in the SRA, just one of the big sequence data repositories, increased by an order of magnitude from 10 tera bases to surpass 100 tera bases. As of January 2017 the SRA contains over 9 peta bases ($9.377 \text{x} 10^{15}$) of sequencing data [4] and ArrayExpress contains 44.5 TB of archived data [5] - this without a doubt constitutes bigdata, which is characterised as data possessing volume, velocity and variety. This ever increasing yield of sequencing data is set to surpass other fields such as astronomy and particle physics [32]. Two important challenges in processing RNA-seq are therefore the ability to handle a vast amount of data, and to quantify and eliminate biases introduced due to the complexity of the sample preparatory steps in such data. We describe a method which employs MapReduce, a distributed programming paradigm on the Apache Spark platform to quantify deviation in read distribution of mapped reads in transcriptomics data. The input data files used for such analysis are industry standard file formats, namely the genome annotation file (GTF) and the aligned reads file (SAM) [28, 14].

MapReduce is a programming model and implementation used by both Apache Spark and Hadoop platforms to enable parallel execution of algorithms which are distributed across a cluster [12]. MapReduce is designed for processing and generating large data sets, and is based on the map and reduce functions commonly used in functional programming [10] and which are conceptually similar to the scatter and reduce functions in the Messaging Passing Interface (MPI) standard [30]. Hadoop and Spark ensures that programs that implement MapReduce are automatically parallelised and executed on the cluster in a distributed fashion. MapReduce has been designed to facilitate programmers in utilising the distributed computing resources without prior specialised expertise in concurrent programming such as with

MPI [34].

Key components of the MapReduce implementation are the map μ and reduce ρ functions. The main data structure associated with these is the key-value pair, a tuple of the form $\langle k, v \rangle$ where k is the key and v is the value. An input to a Map Reduce function is a list of key-value pairs $\langle k, v \rangle_{i=1}^{N}$ to a total size of $\sum_{i=1}^{N} |k_i| + |v_i|$. The reader is directed to a paper by Fish et al. [8] that details the map and reduce functions of the MapReduce model.

2. Methods

Sequence-specific motifs are an issue in microarray data [18, 35] and have also been shown to affect RNA-seq data [38] as well as RNA primers [9], resulting in sequence-specific deviations in the distribution of mapped reads to a reference genome [27, 15]. Furthermore GC content effects have been demonstrated in both Microarray and RNA-seq data [26]. Therefore a method that can quantify these effects by way of deep, transcript analysis is necessary. We describe a novel analysis method, based on analysing sequence motif correlations, that employs MapReduce to quantify bias in NGS data at the exon level. We will look at the input data formats used and explain the implementation of the method, which comprises of two phases, described below:

(1) a distributed phase (using MapReduce) capable of handling high-throughput transcriptomics datasets that yields motif count data for reads of all exons in the dataset.
(2) a non-distributed motif counts analysis phase that quantifies sequence-specific deviations in the distribution of mapped reads by computing correlations of the motif counts computed by the distributed first phase.

The two phases and their steps are outlined in figure 1.

2.1. *Read Distribution in a Transcriptomic Dataset*

RNA-seq often generates short reads, the length of which is dependent on the sequencing platform, and as a result mapped reads are typically not distributed uniformly across exons (figure 2, part B) [27]. Furthermore the number of mapped reads is a function of the number of fragments sequenced and the feature length (i.e. length of the exon), for this reason a number of normalisation methods are used to quantify the number of mapped reads to a feature such as Reads per Kilobase Million (RPKM) [20], and Transcripts per million (TPM) [36]. Our method allows us to investigate the distribution of mapped reads in large datasets.

2.2. *Quantifying Deviations in Read Distribution*

Our method for quantifying sequence-specific deviations in the distribution of mapped reads across an exon was achieved by picking a short sequence motif (typically *4-mers*) which can occur at various positions within the sequence of the exon.

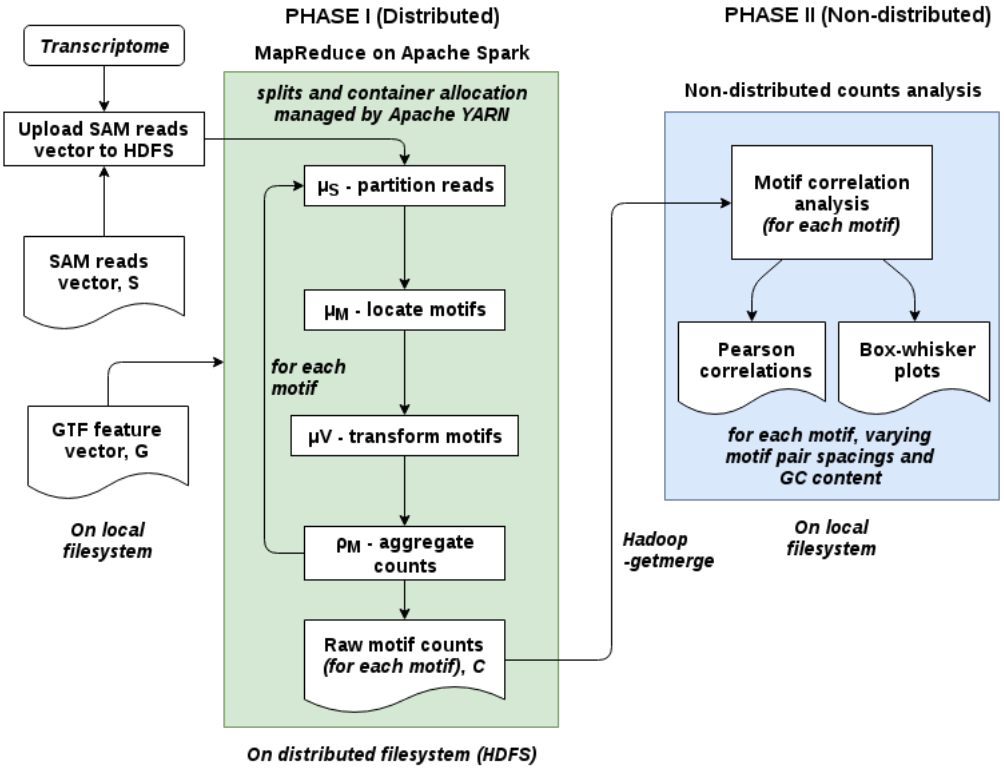

Fig. 1: Overview of method for quantifying sequence-specific deviations in read distribution. Phase I, the distributed phase, comprises 3 map steps and a reduce step on Apache Spark, with intermediate data being stored on HDFS. Phase II, the non-distributed phase, counts analysis phase utilises raw motif count and position data generated by phase I, which has been stored on the local file system.

Next, pairs of these motif occurrences were picked based on their distance apart from each other within the exon and the number of overlapping reads covering each motif position in the pair was counted (figure 3). We term the distance between the motif pairs *motif-spacing* and we have chosen to examine motif pairs that are spaced at 10, 50, 100 and 200bp apart. Uniformity of read distribution was quantified by computing the correlation of the counts for the given motif pair in all exons within the dataset by aggregating the motif pair counts at a given distance apart (motif-spacing) regardless of position within the exon (we used the Pearson correlation coefficient which we discuss later in section 3.3.2). In order to thoroughly examine the affect of sequence-specific motifs on uniformity of read distribution we analysed the correlation for all *4-mer* motifs ranging from `AAAA` to `GGGG` (i.e. 4^4 combinations). This method is implemented in MapReduce which we shall discuss

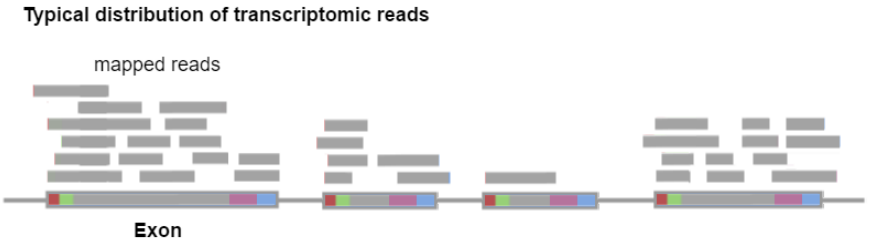

Fig. 2

Typical distribution of RNA-seq reads mapped to an exon.

later in this article.

Fig. 3: Quantification of read coverage using short pairs of sequence motifs (typically *4-mers*) within the reads shown in yellow. Motif-pairs show variable correlation that are used to quantify *sequence-specific* deviations in the distribution of mapped reads across exons.

2.3. *Transcriptomics Input Data*

Two input data files are required for our transcriptomics analysis method (their fields are described in Table 1): i) a reference genome annotation file (GTF) defining exon boundaries for the species under investigation which we store in tuple G and ii) an aligned (mapped) reads data file (SAM) produced by sequence alignment software which we store in tuple S.

Reference genome annotation data have a widely adopted standard for storage of gene structure information - the GTF (Gene Transfer Format) file format which comprises of tab-delimited fields [28]. From the input GTF data file the fields we are interested in are the chromosome name $c_{i,j}$, the feature type $t_{i,j}$, feature start

Table 1: GTF and SAM input data file fields.

GTF (Genome Annotation) file - tuple G		
Field	**Variable**	**Description**
chromosome	$c_{i,j}$	Name of the chromosome sequence in which the feature resides, i.e. "chr1". There are multiple features per chromosome.
feature type	$t_{i,j}$	Name of the feature type, i.e. "CDS", "start_codon", "stop_codon", and "exon"
feature start	$a_{i,j}$	Start position of the feature relative to the chromosome sequence named in $c_{i,j}$, the sequence numbering for the first base starts at 1
feature end	$b_{i,j}$	End position of the feature relative to the chromosome sequence named in $c_{i,j}$, the sequence numbering for the first base starts at 1
SAM (Sequence Aligned Mapped reads) file - tuple S		
Field	**Variable**	**Description**
chromosome	d_k	Name of the chromosome in which the read occurs (referred to as RNAME in SAM specification)
read position	r_k	Leftmost mapping position of the read relative to the chromosome sequence named in d_k, the sequence numbering for the first base starts at 1
read CIGAR	h_k	CIGAR string containing read mapping information
read sequence	s_k	Sequence string of bases in the read

Note: GTF and SAM input data fields used in our analyses and their assigned variables in the input data tuples (G and S) we have defined. Both file formats are tab-delimited, with each GTF feature or SAM read occurring on it's own line. The GTF tuple G is indexed by i and j which refer to the chromosome and feature indices respectively and the SAM tuple S is indexed by k [28, 14].

position $a_{i,j}$ and feature end position $b_{i,j}$, where i, j are the indices of the chromosome and exon respectively.[a] We represent genome annotation data contained in a GTF annotation file of length IJ lines by the tuple defined below, where I is the number of chromosomes and J is the number chromosome features:

$$G_{i,j} = \langle c_{i,j}, a_{i,j}, b_{i,j} \rangle_{i,j=1}^{IJ} \tag{1}$$

SAM (Sequence Alignment/Map) is a widely adopted tab-delimited file format for storing biological sequences aligned to a reference sequence, and is a common output file of read alignment software [14]. The SAM file has a line for each read which we will index by k, and various fields for each read, of which we are interested in the read chromosome name d_k, read position r_k, sequence alignment information which is encoded in a string termed a CIGAR string (Concise Idiosyncratic Gapped Alignment Report) h_k (also discussed in further detail by Li et al.), and the sequence itself s_k, for each read S_k. To represent all the reads in the input SAM file of length N lines we define the following tuple:

[a]We only examine GTF features of type "CDS" (coding sequence). The CDS region of the exon excludes the 5' cap, 5'UTR and 3'UTR and Poly-Adenylated tail - regions that contain regulatory sequences that would bias our motif analysis.

$$S_k = \langle d_k, r_k, g_k, s_k \rangle_{k=1}^{N} \tag{2}$$

3. Phase I - Counting sequence motifs with MapReduce

MapReduce programs are implemented by way of distributed functions such as map μ and reduce ρ. The key-value pair $\langle e, v \rangle$ is the main structure in MapReduce, where e is the key and v is the value. MapReduce functions act on key-value pairs and a typical input to a function is a list of key-value pairs $\langle e, v \rangle_{i=1}^{N}$ within a *split*[b] to a total size of N.

In applying MapReduce to our method to quantify sequence-specific deviations in the distribution of mapped reads, we composed key e from the GTF tuple G that allows us to assign RNA-seq reads to a given unique exon. This MapReduce key e is composed of feature start $a_{i,j}$ and feature end $b_{i,j}$ for each exon boundary. It is constructed by concatenating the 10 digit left-padded string representation of $a_{i,j}$ with the 10 digit left-padded string representation of $b_{i,j}$, producing a final, unique 20 digit key for each exon e.

3.1. *Partitioning reads by exon - GTF-SAM map μ_S*

In order to partition transcriptomic reads in the SAM reads vector S by exon e a MapReduce map operation μ_S is performed. The output of this map step ensures that a set of reads are assigned to the exon which they are aligned to (using read alignment information from upstream read alignment software) and for each exon e comprises of a list of key-value pairs. The associated values $\langle v_1 ... v_M \rangle$ for the exon key e are the short reads S_k obtained from the SAM reads vector S. The output of the μ_S step is therefore in the form of $\langle e, S(e) \rangle$. Each sequence read has a read start position r_k (generated by the alignment software) which is relative to the start position of the chromosome named in d_k and which matches the corresponding chromosome named in $c_{i,j}$. In order to partition reads by the exon they are aligned to, the μ_S step employs a lookup function *el*. Given a read S_k the *el* function uses a binary-search (for optimisation) on the GTF annotation tuple G to return the exon key e for the exon that the read is aligned to. This is achieved using the exon's feature start position $a_{i,j}$ and feature end position $b_{i,j}$, where i, j are the indices of the chromosome and exon respectively. The positions of the reads relative to the range of the exon are were also evaluated as depicted below:

[b]We do not depict the splitting of input and output data and their distribution to MapReduce functions running on nodes of the cluster, which is covered in the Apache Spark documentation.

Fig. 4: Selection of typically distributed RNA-seq reads mapped to an exon. The reads shown in yellow straddle the left of the exon are selected, as are those in green contained within the exon and those straddling the right. Reads shown in grey are not selected for motif count analysis.

3.2. *Counting motifs within reads - MOTIF map μ_M*

The preceeding μ_S map function was used to partition reads by exon returning $\langle e, S(e)\rangle$ which is then passed into the next operation - the μ_M map step. μ_M uses the nucleic acid sequence s_k in the partitioned read and a *4-mer* search motif which is a string we designate f. It returns a tuple $\langle e, P\rangle$ consisting of an exon-key e and a vector P of the start positions at which the search motif occurs for each key and read of the input $\langle e, S(e)\rangle$. To achieve this μ_M map step employs a motif search function sf, which returns a vector of positions in which the motif occurs in s_k. Because the search function utilises positional information in the GTF tuple G and SAM reads tuple S initial values contained in P are chromosome sequence relative positions (relative to start position of chromosome sequence named in $c_{i,j}$) and are then converted into exon-relative positions (relative to $a_{i,j}$). This is achieved by computing offset o_P for the motif start position using the read start position x_k and the exon's GTF feature start position $a_{i,j}$ and applying it to the result P of the motif search function sf as follows:

$$\begin{aligned} o_P &= x_k - a_{i,j} \\ \forall p \in P : p &= p + o_P \end{aligned} \tag{3}$$

Furthermore, in applying the motif search function we ignore overlapping sequences in the string but allow for immediately consecutive occurrences of the motif f. For example when searching for the *4-mer* motif `GGGG` in a sequence read s_k containing run of six Gs (`GGGGGG`) the search function sf would return a single position ($P = \langle 1\rangle$), whereas for a sequence containing a run of eight Gs (`GGGGGGGG`) sf would return two positions ($P = \langle 1, 5\rangle$).

The μ_M map step returns the exon-key e and above re-computed motif positions P in the tuple form $\langle e, P\rangle$, and is defined below:

$$\mu_M(\langle e, S(e)\rangle) = \langle e, P\rangle \begin{array}{l}\text{where } P \text{ is computed from } sf \\ \text{function}\end{array} \tag{4}$$

3.2.1. *Data transformation - VECTOR map μ_V and MOTIF reduce ρ_M*

The final MapReduce steps of the distributed phase are the μ_V and ρ_M steps which together transform intermediate data from the μ_M step in the tuple form $\langle e, P\rangle$ into motif count and position data in the tuple form $\langle e_{n,m}, q_{n,m}, z_{n,m}\rangle_{n,m=1}^{NM}$ which is the final output of the distributed phase I of our analysis method, this process is depicted in figure 5 below:

The penultimate operation in phase I of our analysis method is a μ_V step which operates on tuple $\langle e, P\rangle$ and maps a 1 for each position q in set of motif positions P for each key e. The output yields multiple records in the tuple form $\langle e, q_n, 1\rangle_{n=1}^{N}$ to a total size of N positions for each exon e, as follows:

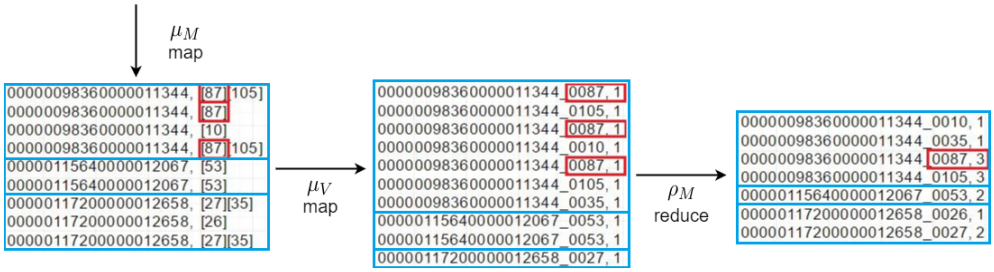

Fig. 5: Transformation of intermediate data by MapReduce steps. In this example, we illustrate that these steps have computed that three short reads have overlap of a specific motif which starts at position 87 of a specific exon

$$\mu_V(\langle e, P\rangle) = \langle e, q_n, 1\rangle \ \forall p \in P \tag{5}$$

The ρ_M is the final operation in phase I and takes for input the set of tuples $\langle e, q_n, 1\rangle_{n=1}^{N}$ produced by the preceeding μ_V step described above. ρ_M aggregates multiple records for a given exon and position into single record of motif counts for a given exon and position in the form $\langle e_{n,m}, q_{n,m}, z_{n,m}\rangle_{n,m=1}^{NM}$ as follows:

$$\rho_M(\langle e, q_n, 1\rangle_{n=1}^{N}) = e, q_n, |q| \ \forall q \in e \tag{6}$$

3.2.2. *Mean Exon GC content - EXON-GC map μ_G and EXON-GC reduce ρ_G*

The subsequent motif analysis phase II requires that the mean GC content for the reads in each exon be pre-computed. It is convenient to use MapReduce for this computation and we use two daisy-chained steps map μ_G and reduce ρ_G. They are computed once per exon, that is they are not repeated for each motif. μ_G takes output from the μ_S step, computes the GC content of each read for each exon in the key-value pair $\langle e, S(e)\rangle$ and ρ_G aggregates average GC content per read gr (as a percentage) by exon key e. The output from ρ_G is the tuple GE, which is defined below:

$$GE = \langle e, \bar{g}e\rangle \tag{7}$$

As outlined in figure 1, phase I is distributed on the Apache Spark platform, and the results are therefore stored on HDFS (Hadoop distributed file system), where each motif m in $C_{n,m}$ is stored in a separate file. The next part of our analysis method phase II is non-distributed, therefore results from the first phase

are downloaded from HDFS on the cluster to the local file system using Hadoop's *-getmerge* command [3].

3.3. *Phase II - Motif Correlations Analysis*

Phase II of our analysis method quantifies sequence-specific deviations in distribution of aligned reads to an exon using data produced by phase I. Specifically we use motif position and count data $C_{n,m}$, in the tuple form $\langle e_{n,m}, q_{n,m}, z_{n,m}\rangle_{n,m=1}^{NM}$ where n, m are the tuple index and motif index respectively, and the mean exon GC content of reads in each given exon, in the tuple form $\langle e, \bar{ge}\rangle$. We will continue to use e as the exon key, the motif position and count tuples C, and the exon mean GC tuples GE which is summarised in Table 2. Furthermore we use f_m to denote the four character string representation of the *4-mer* motif indexed by m.

Table 2: Motif counts tuple $C_{n,m}$

Motif counts tuple $C_{n,m}$	
Variable	**Description**
$e_{n,m}$	Exon key e, a unique identifier for the exon computed by μ_S
$q_{n,m}$	Position in which motif m occurs
$z_{n,m}$	Count of overlapping motif m at position $q_{n,m}$
Mean exon GC tuple GE	
e	Exon key e
$\bar{ge}$	Mean GC content of all reads in exon e

Note: Data produced by MapReduce steps employed in the distributed Phase I of our analyses method.

For our analyses we are interested in a number of properties of the data, at the transcript level, and in particular how these may cause deviation in the distribution of mapped reads. As discussed in section 2.2, in order to investigate how sequence-specific motifs affect read distribution we examined all the possible *4-mer* motifs ranging from `AAAA` to `GGGG`, indexed by m. Furthermore, as depicted in figure 3, we also examined the counts of overlapping motifs at particular base-pair (bp) spacings for all exons. We use d to refer to the spacing distance of motif-pairs in all exons in which motif m occurs, where d is defined below:

$$d \in \{10, 50, 100, 200\} \text{ bp.} \tag{8}$$

The effect of extremes of GC content in sequencing data (as well as microarray data) has been discussed in numerous studies [6, 18], and we therefore also investigate the effect of the mean GC content of reads within the exon $\bar{ge}$ and the GC content of the *4-mer* motif itself gm. In order to partition reads by mean GC content (which we will discuss later) we also define binned GC content ranges (30-40%, 40-50%, 50-60% and 60-70%) for $\bar{ge}$ as follows:

$$\bar{ge} \in \{30-40\%, 40-50\%, 50-60\%, 60-70\%\} \tag{9}$$

The GC content of a given *4-mer* motif gm is defined as being a value from the set defined below:

$$gm \in \{0\%, 25\%, 50\%, 75\%, 100\%\} \tag{10}$$

3.3.1. *Noise removal*

Biological data is rather noisy, this is especially so for transcriptomics data [33, 17] where short sequence reads may map to multiple different exons in the reference genome. For this reason we apply a noise removal process which discards low counts of overlapping motifs - we deemed low counts as those that are within the first quartile Q_1 (25th percentile) of the read counts for all exons in the motif data file for motif m.

3.3.2. *Pearson correlation*

To investigate the effect of the sequence-specificity of the *4-mer* motif m and their spacing distance d on the distribution of mapped reads, we compute the Pearson correlation co-efficients of counts of each motif in the motif-pair (the two sets of measurements) at varying distances for all exons. The test produces a result where a score of +1 indicates positive correlation, -1 indicates inverse correlation and 0 indicates no correlation between the two sets of measurements [19]. We designate tuple D (defined below in equation 11) to store the Pearson correlation for each motif spacing $\rho(d)$ where f_m is the string sequence of the *4-mer* motif:

$$D = \langle f_m, r(10)_m, r(50)_m, r(100)_m, r(200)_m \rangle_m^M \tag{11}$$

The Pearson correlation between the two sets of counts $v_{n,m}, w_{n,m}$ in the motif-pair for motif m on each exon e at a fixed separation of d bp was computed. Each set of counts is of the overlapping motif m at positions separated at a spacing of d bp on the exon and we allow tolerances t of ± 2 bp for 10 and 50 spaced motifs, and ± 4 bp for 100 and 200 bp spaced motifs. In order to compute the Pearson correlation coefficients to store for each spacing d in the correlations tuple D, we utilise a function mp which uses the position and counts tuple $\langle e_{n,m}, q_{n,m}, z_{n,m} \rangle_{n,m=1}^{NM}$, motif-pair spacing distance d, spacing tolerance t, and returns motif-pair counts tuple $W_{n,m}$ which takes the form below:

$$W_{n,m} = \langle e, q_{n,m}, v_{n,m}, q_{o,m}, w_{n,m} \rangle_{n,m}^{N_{match} M} \text{ where } q_{o,m} = q_{n,m} + d \pm t \tag{12}$$

The motif-pair counts tuple $W_{n,m}$ includes positional and counts information for the two motifs in the motif-pair, this is depicted in figure 6 below:

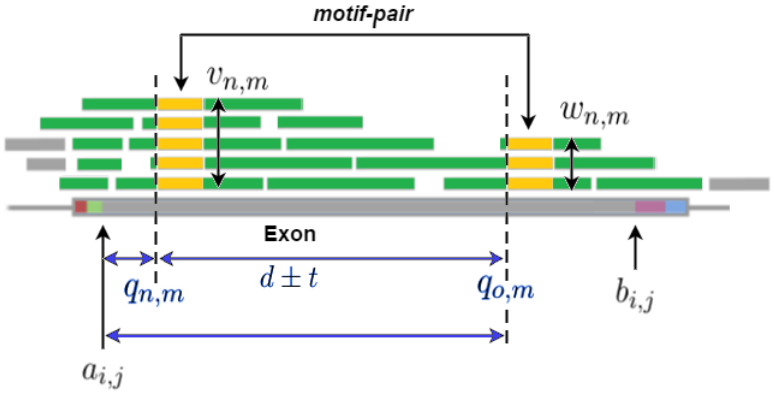

Fig. 6: Motif-pair information held in tuple $W_{n,m}$ for an exon containing mapped reads. Mapped reads are shown in green and the motif-pair in yellow. The positions of each motif $q_{n,m}, q_{o,m}$ relative to the exon feature start $a_{i,j}$ are shown in blue and by dashed lines, as are the distance between them $d \pm t$. Their respective motif counts $v_{n,m}, w_{n,m}$ are also shown by double-headed vertical arrows to represent quantification of overlapping reads containing the motif at that position.

The Pearson correlation between the two counts in the motif-pair tuple $W_{n,m}$ ($v_{n,m}$ and $w_{n,m}$) was computed for each specific motif m and distance d.

4. Results

4.1. *p-values and the Multiple-Comparisons Problem*

As we are performing a considerable number of tests on the input transcriptome there is a possibility that we may mistakingly reject a null hypothesis when it is true (i.e., a "Type I" error). For instance in testing the null hypothesis that there is *no difference in motif-pair correlation between motif-pairs with a GC content of 50% and those with 0%*, we may mistakenly reject this null hypothesis. This problem arises because the more tests we perform the greater the chance that a rare result occurs and is known as the multiple-comparisons problem. We therefore used FDR (False Discovery Rate) corrected p-values and an α value of 0.05 (5%).

4.2. *Testing and Verification with a Synthetic Dataset*

In order to verify our method for quantifying deviations in the read distribution of mapped reads we created a synthetic transcriptome consisting of an artificial GTF genome annotation and a SAM reads file of artificial reads. The SAM reads file of "synthetic" reads was created by generating random start and random end po-

sitions from which contiguous reads were constructed. These randomly generated reads were then assessed to have a mean GC content of 49.6%. As we expected the synthetic dataset had correlations of +1.0 for *all* motif pairs that were found contained within the reads. This is because although the number of reads per exon varies (which represents the expression of the exon) the reads generated are contiguous reads which are of the same length as the exon and are therefore not affected by the length of the exon.

4.3. *Read Distribution in wild type D. melanogaster*

Using MapReduce on Apache Spark[c] and employing our methods explained above we analysed two datasets for the well annotated Drosophila fruit fly species *D. melanogaster* dataset [1] – these were wild-type and mutant-r2 type (eye/antennal disc gene, [21]). Figure 7 is a scatter-matrix plot for wild-type *D. melanogaster*. We can see from the distribution of correlations (bottom right-most graph of the plot), that the majority of correlations are between 0.5 and 0.6, which demonstrates poor correlation between motif-pairs. There is a considerable variability in correlations. On closer inspection we can see that almost all of the higher correlations (>0.8) come from motif-pairs spaced at 10bp, and motifs on the same fragment likely contribute to this, however these are still very widely spread. Interestingly when examining the median exon GC content of the reads (bottom two left-most graphs of the plot) we can see that the median correlations are spread around 0.5 until the mean exon GC content increases beyond 52% whereafter they start to fall, and this is also seen with the motif GC concentration which causes correlations to fall after 75% and a significant drop of correlations for motifs with 100% GC content.

In order to investigate the potential of sequence-specific motifs to effect motif-pair counts and their correlations we examined all 4-mer sequences in the *D. melanogaster* datasets. Table 3 lists the lowest and highest ten correlations for specific *4-mer* sequences for each spacing. The results show that the spread of correlations for given spacings remain very similar with exception of the 10bp which although show the same spread have slightly higher correlations. It is noteworthy that although the *4-mer* sequence appears not to affect correlation the exception is runs of four GGGGs (and their complementary four CCCCs) which showed poorer correlations than other *4-mer* sequence motifs which also corresponds to the increased exon and motif GC content- this could potentially be due to the affect of extreme GC content which has long been recognised in both Microarray and RNA-seq data [26].

Although we observed similar overall trends in correlations for both wild and mutant-r2 type *D. melanogaster* datasets the mutant type dataset shows more variability.

[c]The cluster is a virtual machine comprised of one master and five slave nodes which each have 32GB of RAM and an Intel Xeon E3-1220v3 4-core CPU @ 3.1GHz

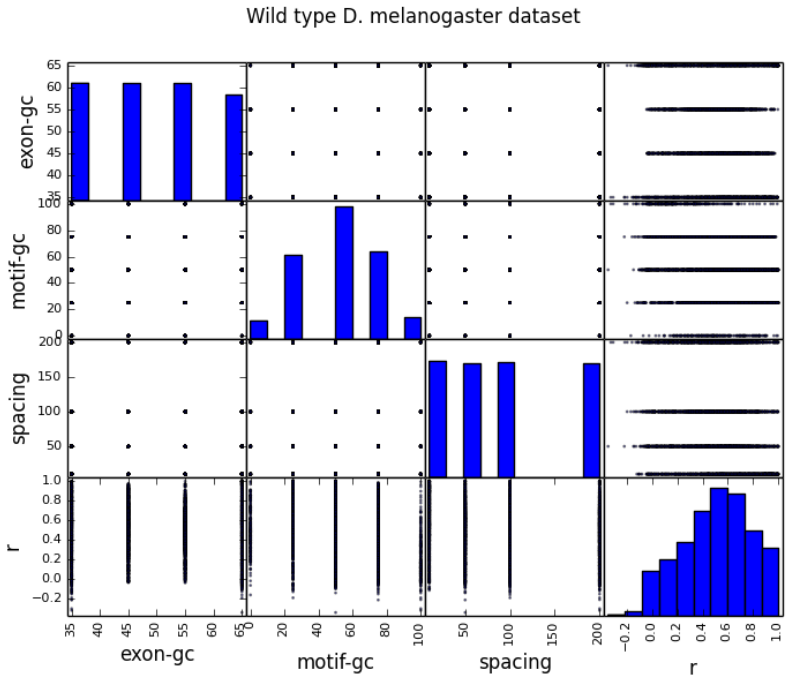

Fig. 7: Scatter-matrix plot for *wild type D. melanogaster* which depicts and compares the distributions of the Pearson correlations for the different parameters we computed correlations for (diagonally arranged blue bar histograms represent record distribution). As we computed correlations of motif-pair counts at spaces $d \in (10, 50, 100, 200)$ to investigate the effect of exon length there are correlations for these spacings, but none for 100bp which we deemed un-necessary to compute.

4.4. *Effect of Exon and Read GC content*

The effect described in section 4.3 on the low correlations associated with runs of Gs or Cs in the motif may be an artefact of the overall GC content of the exon, which has already been noted as affecting read data [26]. In order to understand this, we have compared the distribution of the correlations as a function of the GC content of the motif and exon. An example of this can be found in figure 8, where we compare the distribution of correlations for a motif GC content of 50% and 100% as a function of read GC content for the wild type and mutant-r2 *D. melanogaster* data sets. The wild type data indicates that the correlations are largely independent of the exon GC content and that the medians of the correlations for the motifs with 100% GC content are lower than those with 50% GC content. The mutant data also indicates a similar pattern for the medians but with a more noticeable dependence on the read GC content (specifically for the 50% GC content data). This trend is

Table 3: Wild-type *D. melanogaster* Pearson outliers

Wild type *D. melanogaster*			
Lowest 10 Pearson-correlation outliers and their motifs			
R(10 bp)	R(50 bp)	R(100 bp)	R(200 bp)
CCTA=-0.0064	CCCC=-0.0222	GGGG=0.0065	TAGG=-0.0203
CCCC=-0.0040	CTAG=-0.0001	ACCC=0.0129	GGGG=-0.0187
TCCC=0.0127	CCTA=0.0007	CCTA=0.0520	CCCC=0.0161
CCGG=0.0346	TCCC=0.0045	CGGG=0.0615	GGGT=0.0236
TAGG=0.0463	GGTT=0.0347	TCCC=0.0637	ACCC=0.0362
GGGG=0.1234	GCTA=0.0366	GGTT=0.0675	AGGG=0.0397
CGGG=0.1242	CCCT=0.0507	CGCG=0.0725	CCCG=0.0417
CCCG=0.1325	CGCG=0.0771	CCCC=0.0745	AACC=0.0424
GGGA=0.1351	CCGG=0.0817	TGGG=0.0752	GGGA=0.0715
GCCC=0.1478	CCCG=0.0825	CCGG=0.0762	CCGG=0.0867
Highest 10 Pearson-correlation outliers and their motifs			
R(10 bp)	R(50 bp)	R(100 bp)	R(200 bp)
GTTA=0.8928	TAGA=0.8947	ATAC=0.7652	TACG=0.8697
AGCT=0.8857	AGAT=0.8812	AGTC=0.7623	ACTG=0.8268
TTAA=0.8790	ATAG=0.7815	CTAA=0.7610	CTAC=0.7584
GAGT=0.8644	TCGA=0.7625	GACT=0.7553	GCAT=0.7476
CTGA=0.8580	CGTA=0.7601	CTGT=0.7414	AGTA=0.7260
ATTC=0.8574	ACTA=0.7573	ATTG=0.7393	TGAG=0.7183
TAAG=0.8529	ACTG=0.7499	GTGT=0.7279	ACTA=0.7115
GTAC=0.8494	GTAC=0.7345	TAGT=0.7249	GTAC=0.7057
AGTT=0.8477	GTAG=0.7337	CTTG=0.7246	GATC=0.6998
TAAA=0.8443	TTCA=0.7194	GATG=0.6937	CGTG=0.6993

Note: *D. melanogaster* wild-type Pearson correlation co-efficient outliers (top ten and lowest ten) for different *4-mer* motif sequence pairs at 10, 50, 100 and 200 bp spacings.

observed for the other possible spacings.

5. Conclusions

In this paper we have presented a novel analysis of RNA-seq data that is based on individual reads rather than normalised estimates of exon expression. This has been implemented using Spark and more specifically the MapReduce formalism to make use of the massive parallelism to process these data sets.

We have found that motif-pair correlation is dependent on mean exon GC content and motif GC content, with high GC content showing poorer correlations. Correlations taken from motifs with runs of Gs (or their complement) were of the lowest rank. No other motif bias (particularly for high correlations) was found. Further analysis indicates that for the wild type *D. melanogaster* data set this effect is apparently independent of mean exon GC content. A mild dependence was found for the mutant *D. melanogaster* data set. A valid question is whether the systematic differences seen between these data sets is a product of the difference in the expression patterns between the wild type and mutant. In order to address this we note these are two RNA-Seq data sets that have been generated in the same lab, same species and tissue. It is reasonable to assume that the protocols to prepare the sample and performing the sequencing are the same. There are differences in the transcriptomes because of the genetic perturbation; however we expect that only to be a fraction of the overall change in exon's expression and splicing. Hence the

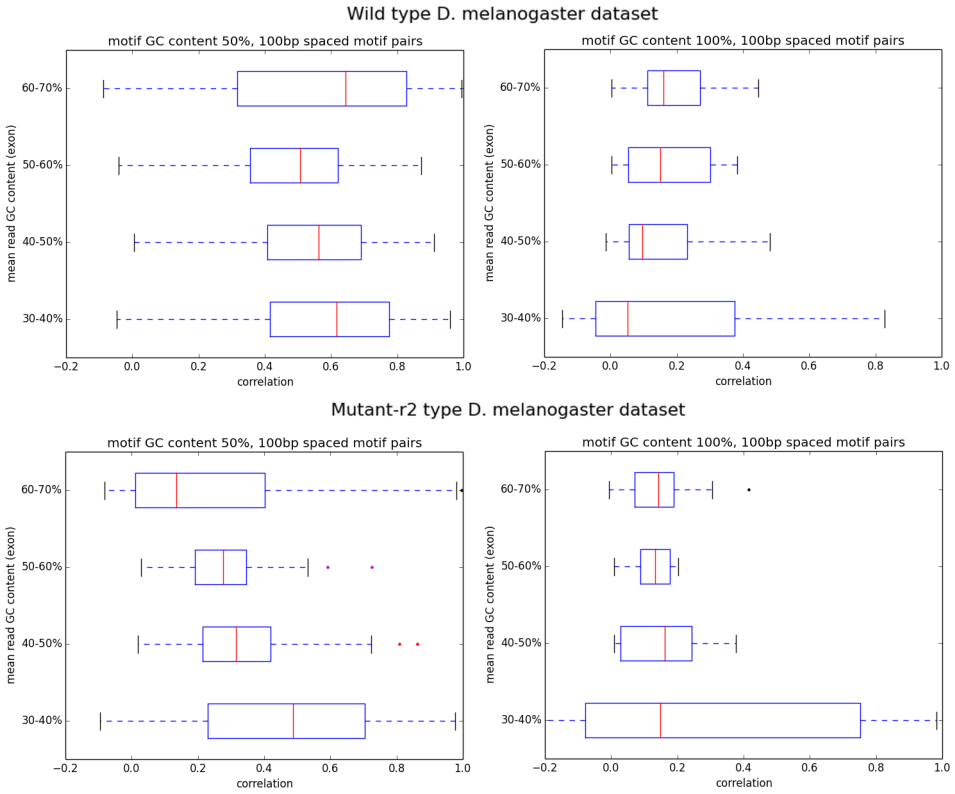

Fig. 8: Box plots for wild and mutant-r2 type *D. melanogaster* datasets comparing motif GC content of 50% vs 100% at 100bp spacing. In the wild type there is a clear dependence of correlation on motif GC content. This effect is less pronounced in the mutant-r2 type but there is still a decrease in correlation which also show a wider spread than that of wild-type.

changes in correlations should remain overall relatively small. What is observed are changes in correlations that represent a much more significant change in the distribution of the short reads between these two data sets. Hence there is a source of variance that is unaccounted for between these data sets. Future work in applying our method to different RNA-seq datasets for other species, especially those with GC extremes, would further characterise the GC bias as a source of variance in the distribution of mapped reads to exon transcripts. Interestingly, it has been found that G/C ending codons usage generally increases with increasing GC bias and decreases with increasing AT bias [23]. Our method could therefore also be applied to

investigate and model GC3 codon usage bias (that is the GC content of the third base), which is highly correlated with overall GC bias, across multiple potentially large datasets.

Our analysis method constitutes a novel and very different approach to investigating bias in RNA-seq data. Our methodology has been conceived for and designed at the molecular and transcript level, i.e. from the bottom up as the exon is the atomic i.e. "non-reducible" unit of the transcription process [31]. In light of the ever increasing amount of publicly deposited data, we have implemented the analysis using the MapReduce programming paradigm. In addition to facilitating the processing of large datasets this also allows for scalability and integration into other analysis pipelines and public datasets.

References

[1] Aerts, S. (2012). The S. Aerts Lab of Computational Biology (LCB) at ku leuven. http://gbiomed.kuleuven.be/english/research/50000622/lcb/. [Online; accessed 4-January-2017].

[2] Alnasir, J. and Shanahan, H. P. (2015). Investigation into the annotation of protocol sequencing steps in the sequence read archive. *GigaScience*, 4:23.

[3] Apache Software Foundation (2016). Hadoop 2.7 documentation. http://hadoop.apache.org/docs/r2.7.2/. [Online; accessed 06-Jan-2017].

[4] Archive), S. R. (2017). Overview of the Sequence Read Archive (SRA). https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi/. [Online; accessed 3-January-2017].

[5] ArrayExpress, EMBL-EBI (2017). ArrayExpress functional genomics data. https://www.ebi.ac.uk/arrayexpress/. [Online; accessed 3-January-2017].

[6] Chen, Y.-C., Liu, T., Yu, C.-H., Chiang, T.-Y., and Hwang, C.-C. (2013). Effects of gc bias in next-generation-sequencing data on de novo genome assembly. *PloS one*, 8(4):e62856.

[7] Edgar, R. (2002). Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Research*, 30(1):207–210.

[8] Fish, B., Kun, J., Lelkes, A. D., Reyzin, L., and Turán, G. (2015). On the computational complexity of mapreduce. In *International Symposium on Distributed Computing*, pages 1–15. Springer.

[9] Hansen, K. D., Brenner, S. E., and Dudoit, S. (2010). Biases in Illumina transcriptome sequencing caused by random hexamer priming. *Nucleic acids research*, 38(12):e131.

[10] Hughes, J. (1989). Why functional programming matters. *The computer journal*, 32(2):98–107.

[11] Kukurba, K. R. and Montgomery, S. B. (2015). Rna sequencing and analysis. *Cold Spring Harbor Protocols*, 2015(11):pdb–top084970.

[12] Lämmel, R. (2008). Googles mapreduce programming modelrevisited. *Science of computer programming*, 70(1):1–30.

[13] Leinonen, R., Sugawara, H., and Shumway, M. (2011). The sequence read

archive. *Nucleic acids research*, 39(Database issue):D19–21.

[14] Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R., et al. (2009). The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079.

[15] Li, J., Jiang, H., and Wong, W. H. (2010). Modeling non-uniformity in short-read rates in rna-seq data. *Genome biology*, 11(5):1.

[16] Liu, R., Loraine, A. E., and Dickerson, J. A. (2014). Comparisons of computational methods for differential alternative splicing detection using rna-seq in plant systems. *BMC bioinformatics*, 15(1):1.

[17] Lykke-Andersen, S. and Jensen, T. H. (2006). Cut it out: silencing of noise in the transcriptome. *Nature Structural and Molecular Biology*, 13(10):860.

[18] Memon, F. N., Owen, A. M., Sanchez-Graillet, O., Upton, G. J., and Harrison, A. P. (2010). Identifying the impact of g-quadruplexes on affymetrix 3'arrays using cloud computing. *Journal of integrative bioinformatics*, 7(111).

[19] Mendenhall, W. and Sincich, T. (1992). *Statistics for engineering and the sciences.* Dellen Publishing Company. San Francisco, CA, USA.

[20] Mortazavi, A., Williams, B. A., McCue, K., Schaeffer, L., and Wold, B. (2008). Mapping and quantifying mammalian transcriptomes by rna-seq. *Nature methods*, 5(7):621–628.

[21] Naval-Sánchez, M., Potier, D., Haagen, L., Sánchez, M., Munck, S., Van de Sande, B., Casares, F., Christiaens, V., and Aerts, S. (2013). Comparative motif discovery combined with comparative transcriptomics yields accurate targetome and enhancer predictions. *Genome research*, 23(1):74–88.

[22] Ozsolak, F., Platt, A. R., Jones, D. R., Reifenberger, J. G., Sass, L. E., McInerney, P., Thompson, J. F., Bowers, J., Jarosz, M., and Milos, P. M. (2009). Direct rna sequencing. *Nature*, 461(7265):814–818.

[23] Palidwor, G. A., Perkins, T. J., and Xia, X. (2010). A general model of codon bias due to gc mutational bias. *PLoS One*, 5(10):e13431.

[24] Park, J. W., Tokheim, C., Shen, S., and Xing, Y. (2013). Identifying differential alternative splicing events from rna sequencing data using rnaseq-mats. *Deep Sequencing Data Analysis*, pages 171–179.

[25] Raz, T., Kapranov, P., Lipson, D., Letovsky, S., Milos, P. M., and Thompson, J. F. (2011). Protocol dependence of sequencing-based gene expression measurements. *PloS one*, 6(5):e19287.

[26] Risso, D., Schwartz, K., Sherlock, G., and Dudoit, S. (2011). Gc-content normalization for rna-seq data. *BMC bioinformatics*, 12(1):480.

[27] Roberts, A., Trapnell, C., Donaghey, J., Rinn, J. L., and Pachter, L. (2011). Improving RNA-Seq expression estimates by correcting for fragment bias. *Genome biology*, 12(3):R22.

[28] Sanger, N. (2012). Ensembl, GTFGFF file format specification. http://www.ensembl.org/info/website/upload/gff.html. [Online; accessed 15-November-2016].

[29] Shendure, J., Mitra, R. D., Varma, C., and Church, G. M. (2004). Advanced

sequencing technologies: methods and goals. *Nature Reviews Genetics*, 5(5):335–344.

[30] Snir, M. (1998). *MPI–the Complete Reference: The MPI core*, volume 1. MIT press.

[31] Stalteri, M. A. and Harrison, A. P. (2007). Interpretation of multiple probe sets mapping to the same gene in affymetrix genechips. *BMC bioinformatics*, 8(1):1.

[32] Stephens, Z. D., Lee, S. Y., Faghri, F., Campbell, R. H., Zhai, C., Efron, M. J., Iyer, R., Schatz, M. C., Sinha, S., and Robinson, G. E. (2015). Big data: Astronomical or genomical? *PLoS Biol*, 13(7):1–11.

[33] Struhl, K. (2007). Transcriptional noise and the fidelity of initiation by rna polymerase ii. *Nature structural & molecular biology*, 14(2):103–105.

[34] Taylor, R. C. (2010). An overview of the hadoop/mapreduce/hbase framework and its current applications in bioinformatics. *BMC bioinformatics*, 11(Suppl 12):S1.

[35] Upton, G. J., Langdon, W. B., and Harrison, A. P. (2008). G-spots cause incorrect expression measurement in affymetrix microarrays. *BMC genomics*, 9(1):1.

[36] Wagner, G. P., Kin, K., and Lynch, V. J. (2012). Measurement of mrna abundance using rna-seq data: Rpkm measure is inconsistent among samples. *Theory in Biosciences*, 131(4):281–285.

[37] Wang, Z., Gerstein, M., and Snyder, M. (2009). Rna-seq: a revolutionary tool for transcriptomics. *Nature reviews genetics*, 10(1):57–63.

[38] Zheng, W., Chung, L. M., and Zhao, H. (2011). Bias detection and correction in rna-sequencing data. *BMC bioinformatics*, 12(1):1.