BIOINFORMATICS

Vol. 30 ISMB 2014, pages i283–i292
doi:10.1093/bioinformatics/btu288

RNA-Skim: a rapid method for RNA-Seq quantification at transcript level

Zhaojun Zhang[1] and Wei Wang[2,*]

[1]Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA and [2]Department of Computer Science, University of California, Los Angeles, CA, USA

ABSTRACT

Motivation: RNA-Seq technique has been demonstrated as a revolutionary means for exploring transcriptome because it provides deep coverage and base pair-level resolution. RNA-Seq quantification is proven to be an efficient alternative to Microarray technique in gene expression study, and it is a critical component in RNA-Seq differential expression analysis. Most existing RNA-Seq quantification tools require the alignments of fragments to either a genome or a transcriptome, entailing a time-consuming and intricate alignment step. To improve the performance of RNA-Seq quantification, an alignment-free method, Sailfish, has been recently proposed to quantify transcript abundances using all k-mers in the transcriptome, demonstrating the feasibility of designing an efficient alignment-free method for transcriptome quantification. Even though Sailfish is substantially faster than alternative alignment-dependent methods such as Cufflinks, using all k-mers in the transcriptome quantification impedes the scalability of the method.

Results: We propose a novel RNA-Seq quantification method, RNA-Skim, which partitions the transcriptome into disjoint transcript clusters based on sequence similarity, and introduces the notion of sig-mers, which are a special type of k-mers uniquely associated with each cluster. We demonstrate that the sig-mer counts within a cluster are sufficient for estimating transcript abundances with accuracy comparable with any state-of-the-art method. This enables RNA-Skim to perform transcript quantification on each cluster independently, reducing a complex optimization problem into smaller optimization tasks that can be run in parallel. As a result, RNA-Skim uses $<4\%$ of the k-mers and $<10\%$ of the CPU time required by Sailfish. It is able to finish transcriptome quantification in <10 min per sample by using just a single thread on a commodity computer, which represents >100 speedup over the state-of-the-art alignment-based methods, while delivering comparable or higher accuracy.

Availability and implementation: The software is available at http://www.csbio.unc.edu/rs.

Contact: weiwang@cs.ucla.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

RNA-Seq technique has been demonstrated as a revolutionary means for examining transcriptome because it provides incomparable deep coverage and base pair-level resolution (Ozsolak and Milos, 2010). Though RNA-Seq sequencing exhibits itself as an efficient alternative to Microarray techniques in gene expression study (Wang *et al.*, 2009), it also brings unprecedented challenges, including (but not limited to) how to rapidly and effectively process the massive data produced by the proliferation of RNA-Seq high-throughput sequencing, how to build statistical model for accurate quantification of transcript abundances for transcriptome, etc.

Most of current RNA-Seq tools for RNA-Seq quantification contain two main steps: an alignment step and a quantification step. Various aligners [TopHat (Trapnell *et al.*, 2009), SpliceMap (Au *et al.*, 2010), MapSplice (Wang *et al.*, 2010)] are devised to infer the origin of each RNA-Seq fragment in the genome. The alignment step is usually time-consuming, requiring substantial computational resources and demanding hours to align even one individual's RNA-Seq data. Because there are multiple variations of RNA-Seq sequencing techniques, e.g. single-end sequencing and paired-end sequencing, to facilitate the discussion in this article, we simply refer to the *read* or the pair of reads from a RNA-Seq fragment as a *fragment*. More importantly, a significant percentage of the fragments cannot be aligned without ambiguity, which yields a complicated problem in the quantification step: how to assign the ambiguous fragments to compatible transcripts and to accurately estimate the transcript abundances. To tackle the fragment multiple-assignment problem, an expectation-maximization (EM) algorithm (Pachter, 2011) is often used to probabilistically resolve the ambiguity of fragment assignments: at each iteration, it assigns fragments to their compatible transcripts with a probability proportional to the transcript abundances, and then updates the transcript abundances to be the total weights contributed from the assigned fragments, until a convergence is reached. The EM algorithm's simplicity in its formulation and implementation makes it a popular choice in several RNA-Seq quantification methods [Cufflinks (Trapnell *et al.*, 2010), Scripture (Guttman *et al.*, 2010), RSEM (Li and Dewey, 2011), eXpress (Roberts and Pachter, 2013)]. Because all fragments and all transcripts are quantified at the same time in the EM algorithm, it usually requires considerable running time. Some studies [IsoEM (Nicolae *et al.*, 2011), MMSEQ (Turro *et al.*, 2011)] reduced the scale of the problem by collapsing reads if they can be aligned to the same set of transcripts. It is also worth mentioning that RNA-Seq quantification is an important first step for differential analysis on the transcript abundances among different samples (Trapnell *et al.*, 2012).

The alignment step is a vital step in the RNA-Seq assembly study (Trapnell *et al.*, 2010) and has become the computational bottleneck for RNA-Seq quantification tasks. If users are only interested in RNA-Seq quantification of an annotated transcriptome, aligning RNA-Seq fragments to the genome seems cumbersome: not only do the RNA-Seq aligners take a long time to align fragments to the genome by exhaustively searching all

*To whom correspondence should be addressed.

possible splice junctions in the fragments, they may also generate misaligned results owing to repetitive regions in the genome or sequencing errors, introducing errors in the quantification results (Zhang *et al.*, 2013).

From another perspective, the annotation databases of transcriptome, e.g. RefSeq (Pruitt *et al.*, 2007) and Ensembl (Flicek *et al.*, 2011), play an increasingly important role in RNA-Seq quantification. For example, TopHat/Cufflinks supports a mode that allows users to specify the transcriptome by supplying an annotation database (a GTF file). RSEM (Li and Dewey, 2011) uses bowtie (Langmead *et al.*, 2009)—a DNA sequence aligner—to align fragments directly to the transcriptome. Aligning RNA-Seq fragments to transcriptome avoids the computation to detect novel splice junctions in fragments and eliminates the non-transcriptome regions in the genome from further examination, and thus reduces the total running time of the quantification method and the number of erroneous alignments in the results.

To further improve the performance, the utility of k-mers was recently proposed. The concept of k-mers—short and consecutive sequences containing k nucleic acids—has been widely used in bioinformatics, including genome and transcriptome assembly (Fu *et al.*, 2014; Grabherr *et al.*, 2011), error correction in sequence reads (Le *et al.*, 2013), etc. Because the number of k-mers in the genome or transcriptome is enormous when k is large (e.g. $k \geq 25$), the need to store all k-mers impedes their counting. Most of existing methods save memory usage during the computation by using sophisticated algorithms and advanced data structures [bloom filter (Melsted and Pritchard, 2011), lock-free memory-efficient hash table (Marcais and Kingsford, 2011), suffix array (Kurtz *et al.*, 2008)] or relying on disk space to compensate memory space (Rizk *et al.*, 2013).

Thanks to the recent advances in both annotated transcriptome and algorithms to rapidly count k-mers, the transcriptome-based alignment-free method, Sailfish (Patro *et al.*, 2013), requires 20 times less running time and generates comparable results with alignment-dependent quantification methods. Sailfish is a lightweight method: it first builds a unique index of all k-mers that appear at least once in the transcriptome, counts the occurrences of the k-mers in the RNA-Seq fragments and quantifies the transcripts by the number of occurrences of the k-mers through an EM algorithm.

Regardless of being alignment-dependent or alignment-free, all methods need to recover the fragment depth—the number of fragments that cover a specific location—across the whole transcriptome as one of the initial steps. However, none of the existing methods exploit the strong redundancy of the fragment depth in RNA-Seq data. More specifically, Fig. 1 shows a strong correlation between the fragment depth of any two locations that are a certain distance apart on the transcriptome, varying the distance from 1 to 100 bp. Even when the two locations are 20 bp away from each other, the Pearson correlation score is still as high as 0.985. In other words, if an RNA-Seq quantification method that is able to recover the fragment depths for every 20 bp and quantify the abundance levels based on such information, there should be no significant accuracy loss in the result. Recently, Uziela and Honkela (2013) developed a method that simply counts the number of alignments that covers the locations of hybridization probes used in the gene expression studies.

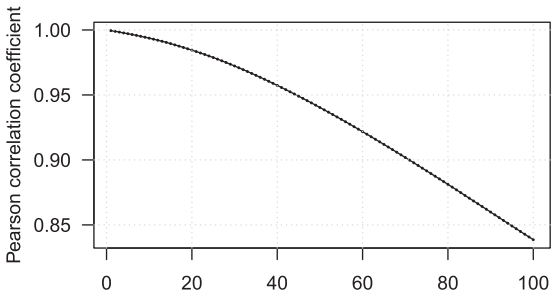

Fig. 1. This figure shows the correlations of the fragment depth of any pair of locations as a function of the distance between the two locations from 1 to 100 bp. This figure is generated based on the alignments reported by TopHat on a real RNA-Seq data

Though these probes only represent a sparse sampling on every transcript in the transcriptome, the method still provides reasonably accurate results. The observation and the method inspire us to ask the following question: what is the minimum information we need to achieve comparable accuracy in RNA-Seq quantification to the state-of-the-art methods? More specifically, does there exist a subset of k-mers that can provide accurate transcriptome quantification? And if so, how do we identify and use them to quantify transcriptome efficiently?

To answer these questions, we introduced a special type of k-mers called *sig-mers*, which only appear in a (small) subset of transcripts in the transcriptome. Based on these sig-mers, we developed a method, RNA-Skim, which is much faster than Sailfish and also maintains the same level of accuracy in the results. RNA-Skim includes two stages, *preparation* and *quantification*. In the preparation stage, RNA-Skim first partitions transcripts into clusters and uses bloom filters to discover all sig-mers for each transcript cluster, from which a small yet informative subset of sig-mers is selected to be used in the quantification stage. In the quantification stage, a rolling hash method (Karp and Rabin, 1987) is developed to rapidly count the occurrences of the selected sig-mers, and an EM algorithm is used to properly estimate the transcript abundance levels using the sig-mer counts. Because no sig-mer is shared by two transcript clusters, the task can be easily divided into many small quantification problems, which significantly reduces the scale of each EM process and also makes it trivial to be parallelized. While RNA-Skim provides similar results to those of alternative methods, it only consumes 10% of the computational resources required by Sailfish.

In this article, we first describe the RNA-Skim method, then discuss how we compared RNA-Skim with other methods, followed by the experimental results using both simulated and real data.

2 METHOD

In this section, we introduced the notion of sig-mers, which is a special type of k-mers that may serve as *signatures* of a cluster of transcripts, distinguishing them from transcripts in other clusters in the transcriptome that do not contain these k-mers.

2.1 sig-mer

In this article, an annotated *transcriptome* Θ consists of a set of T transcripts: $\Theta = \{t_1, ..., t_T\}$. A *transcript* t is an RNA sequence composed of a string of four bases 'A', 'U', 'C' and 'G'. In this article, we use the corresponding four DNA nucleotide bases 'A', 'T', 'C', 'G' to represent. The length of a transcript sequence may vary from 100 to 10 000 bp. A *partition* of a given transcriptome Θ groups all transcripts into P disjoint non-empty subsets or clusters, denoted by $(\Theta) = \{\phi_1, ..., \phi_P\}$. For example, one commonly adopted partition of transcriptome is to group transcripts into genes based on their locations on the genome. For any transcript t, we use $\phi(t)$ to denote the cluster to which t belongs.

A substring of length k from a transcript sequence, its reverse sequence, its complimentary sequence or its reverse and complimentary sequence is called a *k-mer* of the transcript. We define a function **k-mer**() to represent the set of all k-mers from a single transcript or a cluster of transcripts, denoted as **k-mer**(t) or **k-mer**(ϕ_p), respectively. For simplicity, if a string s is a k-mer of transcript t, we say $s \in$ **k-mer**(t). In this case, $s \in$ **k-mer**($\phi(t)$) is also true.

Definition. Given a length k, a transcriptome Θ and its partition (Θ), if a k-mer s only exists in one cluster ϕ_p and never appears in other clusters $\Theta \setminus \phi_p$, we call it a *sig-mer* of cluster ϕ_p. And for any given cluster ϕ_p, we denote all of its sig-mers as $\Omega(\phi_p)$. That is,

$$\Omega(\phi_p) = \{s | s \in \textbf{k-mer}(\phi_p), \forall \phi_q \in \Theta \setminus \phi_p, s \in \textbf{k-mer}(\phi_q)\}.$$

Sig-mers characterize the uniqueness of each cluster. It is obvious that the number of sig-mers heavily depends on the transcriptome partition. If transcripts with similar sequences are assigned to different clusters, k-mers from these transcripts may not qualify as sig-mers. Consequently, fewer sig-mers may be identified, and in the worst case, some cluster may not have any sig-mers.

2.2 Workflow of RNA-Skim

Because sig-mers are unique to only one cluster of transcripts, if a sig-mer occurs in some RNA-Seq reads, it indicates the sig-mer's corresponding transcripts may be expressed. Therefore, its occurrence in the RNA-Seq data may serve as an accurate and reliable indicator of the abundance levels of these transcripts. We proposed a method, RNA-Skim, for quantifying the transcripts using the sig-mer counts in RNA-Seq data. Because no sig-mer is shared between transcript clusters, the problem reduces to quantifying transcript abundances using sig-mer counts within each cluster, which can be solved much more efficiently and can be easily parallelized. This is different from Sailfish that uses all k-mers in the transcriptome. In fact, RNA-Skim can be considered as a generalization of Sailfish: if the whole transcriptome is treated as a single cluster that includes all transcripts, all k-mers become sig-mers, and RNA-Skim degenerates to the exact formulation of Sailfish.

The workflow of RNA-Skim includes two stages: *preparation* and *quantification*. In preparation, RNA-Skim clusters the transcripts based on their sequence similarities, finds all sig-mers for each transcript cluster and selects a subset of sig-mers to be used in the quantification stage. In quantification, RNA-Skim quickly counts the occurrences of sig-mers and quantifies transcripts within each cluster. The preparation stage of RNA-Skim does not require RNA-Seq read data and thus can be computed once as an offline process and be repeatedly used in the quantification stage.

2.3 Preparation stage

In the preparation stage, RNA-Skim only requires users to supply a transcriptome (including all transcript sequences and gene annotations) and specifies a desired sig-mer length to be used in RNA-Skim.

Transcriptome Partitioning A straightforward way to partition transcripts is based on their genome locations from an annotation database. However, the result of this partitioning approach may not be optimal because some transcripts of different genes may have similar sequences or share common subsequences. To minimize the number of common k-mers shared between clusters, RNA-Skim uses a sequence similarity-based algorithm to generate a partition of transcriptome, instead of using any existing partition. We first define the k-mer-based similarity, which is used as the sequence similarity in the algorithm.

Definition. The k-mer-based similarity of two sets of sequences ϕ_i and ϕ_j is defined as the higher of the two ratios: the number of common k-mers divided by the total number of k-mers in ϕ_i, and the number of common k-mers divided by the total number of k-mers in ϕ_j:

$$\textbf{k-mer-Similarity}(\phi_i, \phi_j) = \tag{1}$$

$$\max\left(\frac{|\textbf{k-mer}(\phi_i) \cap \textbf{k-mer}(\phi_j)|}{|\textbf{k-mer}(\phi_i)|}, \frac{|\textbf{k-mer}(\phi_i) \cap \textbf{k-mer}(\phi_j)|}{|\textbf{k-mer}(\phi_j)|}\right). \tag{2}$$

Transcripts from the same gene are likely to be similar to each other. To avoid unnecessary computation, RNA-Skim operates at the level of genes rather than transcripts. However, calculating the exact similarity between a pair of genes requires generating all k-mers appearing in each gene and taking the intersection of the two sets. This is computationally expensive. To expedite the computation, RNA-Skim uses the data structure—bloom filter (Bloom, 1970)—coupled with a sampling-based approach to approximate the similarity between two genes. The bloom filter is a space-efficient probabilistic data structure that is used to test whether an element is a member of a set, without the need of storing the set explicitly. A bloom filter includes a vector of bits and several independent hash functions. Initially, all bits are set to 0. When an element is added to the bloom filter, the bits based on the hash values of the element are set to 1. The bloom filter reports an element is in the bloom filter if its corresponding bits are all set to 1. A bloom filter may yield a small number of false positives, but no false negatives. The false-positive rate is bounded if the number of elements in the set is known. It can be maintained efficiently when new members are added to the set.

RNA-Skim first builds a bloom filter for the set of k-mers of each gene. Then, it randomly samples two subsets of k-mers—noted as $X(\phi_i)$ and $X(\phi_j)$—from the pair of genes, and the **k-mer-Similarity**(ϕ_i, ϕ_j) is approximated by $\max(\frac{|X(\phi_i) \cap \textbf{k-mer}(\phi_j)|}{|X(\phi_i)|}, \frac{|\textbf{k-mer}(\phi_i) \cap X(\phi_j)|}{|X(\phi_j)|})$ (our experiments show that we only need a small number (e.g. 10) of k-mers from each gene to achieve approximation with high accuracy). After we calculate the approximated similarities for every pair of genes, an undirected graph is built with each node representing a gene. There is an edge between two nodes if the similarity of the two corresponding genes exceeds a user-specified threshold γ. Each connected component of this graph naturally forms a cluster of nodes; each cluster of nodes forms a cluster of genes and transcripts of the genes.

Sig-mers discovery By definition, the sig-mers are essentially the k-mers occurring in only one cluster of transcripts. A brute force way to find all sig-mers is, for every k-mer in the transcriptome, to determine whether the k-mer that appears in one cluster also appears in some other cluster. Because the number of possible k-mers is in the order of billions, without any sophisticated data structure and data compression algorithms, storing the k-mers alone will easily take at least tens of gigabytes of memory space, which is way beyond the capacity of any commodity computer.

RNA-Skim again uses bloom filters to reduce memory usage. Three types of bloom filters are used: a bloom filter *BF.ALL* for checking whether a given k-mer has been examined, a bloom filter *BF.DUP* for checking whether a given k-mer appears in more than one cluster and a bloom filter $BF.S(\phi_p)$ for each cluster ϕ_p for checking whether a given k-mer is in **k-mer**(ϕ_p).

First, for each cluster ϕ_p, all distinct k-mers in it are enumerated: RNA-Skim enumerates all k-mers for every transcript in the cluster, and adds them to $BF.S(\phi_p)$; if a k-mer is already in $BF.S(\phi_p)$, it will be ignored. Second, every distinct k-mer in ϕ_p is added into *BF.ALL*, and if it is already in *BF.ALL* (that is, it was added when RNA-Skim examined other clusters), it is added into *BF.DUP*. Therefore, if a k-mer occurs in multiple clusters, it is added in *BF.DUP*. Last, every k-mer of the transcriptome is enumerated again, and if the k-mer is not in *BF.DUP*, it is reported as a sig-mer, as it only occurs in one cluster.

Because bloom filters may have false-positive reports, but never have false-negative reports, through this approach, some genuine sig-mer strings may be missed, but a non-sig-mer will never be labeled as a sig-mer. Figure 2 shows the pseudocode of our algorithm.

Sig-mers selection RNA-Skim does not use all sig-mers because they are still numerous. Instead, RNA-Skim selects a subset of sig-mers for the quantification stage. We used a simple approach to select sig-mers from all sig-mers found by the previous step: for every transcript, sig-mers are evenly chosen based on the sig-mer locations such that any two sig-mers are at least 50 base pair away from each other in the given transcript. Because some sig-mers may appear in multiple transcripts in the same cluster, for every selected sig-mer, all transcripts are re-examined, and the ones that contain the sig-mer are also recorded. Through this approach, we can guarantee that every transcript is associated with some sig-mers (as long as there exist some sig-mers). A good sig-mer coverage is crucial for accurate quantification of transcript abundance. The final output of the preparation step includes the partition of the transcriptome, selected sig-mers and their associating clusters and transcripts.

2.4 Quantification stage

The quantification stage requires users to provide RNA-Seq data (e.g. FASTQ/FASTA files) and the selected sig-mers associated with transcripts containing them from the preparation stage.

Sig-mer counting Because the number of sig-mers used in RNA-Skim is much smaller than the number of k-mers typically used by other k-mer-based approaches, all sig-mers can be stored in a hash table in memory. The number of occurrences of all sig-mers can be counted by enumerating all k-mers in the RNA-Seq reads and looking up the k-mers in the hash table to update the corresponding counters. RNA-Skim basically follows this scheme with a tweak on the hash function to further speed up the computation.

1. **foreach** partition of transcripts $\phi_p \in \Theta$
2. **foreach** location $l \in \phi_p$
3. generate the k-mer s at the location l
4. **if** $s \notin BF.S(\phi_p)$ **then**
5. Add s into $BF.S(\phi_p)$
6. **if** $s \notin BF.ALL$ **then**
7. Add s into $BF.ALL$
8. **else**
9. Add s into $BF.DUP$
10. **end foreach**
11. **end foreach**
12. **foreach** partition of transcripts $\phi_p \in \Theta$
13. **foreach** location $l \in \phi_p$
14. generate the k-mer s at the location l
15. **if** $s \notin BF.DUP$ **then**
16. **Report** s as a **sig-mer** of ϕ_p.
17. **end foreach**
18. **end foreach**

Fig. 2. The pseudocode to find all sig-mers

In a straightforward implementation of the previously described algorithm, every k-mer incurs an O(k) operation to calculate its hash value, and this hash operation can be further reduced to O(1) by the Robin–Karp pattern matching algorithm (Karp and Rabin, 1987). The Robin–Karp pattern matching algorithm requires a special hash function—rolling hash—that only uses multiplications, additions and subtractions.

In rolling hash, the hash value $\mathbf{H}(r)$ of the first k-mer in the RNA-Seq read r is calculated by

$$\mathbf{H}(r[0, ..., k-1]) = \chi(r[0]) \times h^{k-1} + \chi(r[1]) \times h^{k-2} + ... + \chi(r[k-1]) \times h^0,$$

where h is the base of the hash function, $r[i]$ is the ith character in s and the character hash function $\chi()$ maps a character to an integer value. One way to calculate the hash value for the (sequentially ordered) second k-mer $r[1, ..., k]$ is

$$\mathbf{H}(r[1, ..., k]) = \chi(r[1]) \times h^{k-1} + \chi(r[2]) \times h^{k-2} + ... + \chi(r[k]) \times h^0.$$

But thanks to the structure of the rolling hash function, $\mathbf{H}(r[1, ..., k])$ can be calculated in a much faster way:

$$\mathbf{H}(r[1, ..., k]) = (\mathbf{H}(r[0, ..., k-1]) - \chi(r[0]) \times h^{k-1}) \times h + \chi(r[k]) \times h^0,$$

which only requires one subtraction, three multiplications and one addition. We can look up the hash value in the hash table, and if it is in the hash table, its associated counter is incremented accordingly. Because RNA-Skim uses this specially designed hash function, we implemented our own hash table in RNA-Skim using open addressing with linear probing. The base h is arbitrarily set to be a prime number 37, and the function $\chi()$ maps every character to its actual ASCII value.

Quantification Because every cluster of transcripts has a unique set of sig-mers, which are the k-mers that never appear in other transcript clusters, every cluster can be independently quantified by RNA-Skim, resulting in a set of smaller independent quantification problems, instead of one huge whole transcriptome quantification problem in other approaches.

Formally, if ϕ_p is a cluster of transcripts, the set of sig-mers of ϕ_P is denoted by $S(\phi_p)$, a sig-mer is denoted by s ($s \in S(\phi_p)$), the set of all occurrences of sig-mers is denoted by $O(\phi_p)$, an occurrence of a sig-mer in the RNA-Seq dataset is denoted by o ($o \in O(\phi_p)$) and the sig-mer of the occurrence is denoted by z_o. From the previous steps, we obtained c_s (the number of occurrences of the sig-mer s in the RNA-Seq data), $y_{s,t}$ (binary variables indicating whether the sig-mer s is contained in the sequence of transcript t) and b_t (the number of sig-mers that are contained by transcript t). C is the number of occurrences of all sig-mers ($C = \sum c_s$).

Same as in the previous study (Pachter, 2011), we define $\mathbf{\Psi} = \{\alpha_t\}_{t \in \phi_p}$ where α_t is the proportion of all selected sig-mers that are included by the reads from transcript t, and $\sum \alpha_t = 1$. For an occurrence o, $p(o \in t)$ represents the probability that o is chosen from transcript t, in a generative model,

$$p(o \in t) = y_{z_o,t} \frac{\alpha_t}{b_t} \tag{3}$$

Therefore, the likelihood of observing all occurrences of the sig-mers as a function of the parameter $\mathbf{\Psi}$ is

$$\mathcal{L}(\Psi) = \prod_{o \in O(\phi_p)} \sum_{t \in \phi_p} p(o \in t) = \prod_{o \in O(\phi_p)} \sum_{t \in \phi_p} y_{z_o,t} \frac{\alpha_t}{b_t} \tag{4}$$

$$= \prod_{s \in S(\phi_p)} \left(\sum_{t \in \phi_p} y_{s,t} \frac{\alpha_t}{b_t}\right)^{c_s}. \tag{5}$$

This is in spirit similar to the likelihood function used in other studies, except that this is the likelihood of observing sig-mers rather than fragments (Li and Dewey, 2011) or k-mers (Patro *et al*., 2013). Thus, we also used an EM algorithm to find Ψ that maximizes the likelihood: it alternates between allocating the fraction of counts of observed sig-mers to transcripts according to the proportions Ψ and updating Ψ given the amount of sig-mers assigned to transcripts. RNA-Skim also applies the same technique used in Patro *et al*. (2013), Nicolae *et al*. (2011) and Turro *et al*. (2011) to collapse sig-mers if they are contained by the same set of transcripts. (See the Supplementary Material)

RNA-Skim reports both Reads Per Kilobase per Million mapped reads (RPKM) and Transcripts Per Million as the relative abundance estimations of the transcripts, and both metrics are calculated by the way used in Sailfish (Patro *et al*., 2013).

So far, we have described both preparation and quantification stages in RNA-Skim. In the last, a toy example is provided to illustrate how each stage works in RNA-Skim in Figure 3.

3 SOFTWARE FOR COMPARISON

RNA-Skim is implemented in C++ with heavy usage of the open-source libraries bloomd (Dadgar, 2013), protobuf (Google, 2013) and an open-source class StringPiece (Hsieh, 2013). The parameter settings will be discussed in the Section 5.

We compared RNA-Skim with four different quantification methods: Sailfish (0.6.2), Cufflinks (2.1.1), RSEM (1.2.8) and eXpress (1.5.1) in both simulated and real datasets. TopHat (2.0.10) and Bowtie (1.0.0) are used as the aligners when needed.

For Sailfish, we set k-mer size to be 31 because this value gives the highest accuracy in the simulation study, among all k-mer sizes supported by Sailfish ($k \leq 31$). For other software, we followed the experiments in Patro *et al*. (2013) to set the parameters. Input to Cufflinks was generated by TopHat, which used Bowtie (–bowtie1), allowing up to three mismatches per read (-N 3 and –read-edit-dist 3). Both TopHat and Cufflinks were provided with a reference transcriptome. RSEM and eXpress directly used Bowtie to align the reads to the transcriptome, with the argument (-N 3) to allow up to three mismatches per read. The eXpress was executed in the streaming mode, to save the total quantification running time. For simulation study, we used the estimations without bias correction for Sailfish, Cufflinks and eXpress. For real datasets, the estimations with bias correction are used for these three methods. For RSEM, since it does not provide an option to control the bias correction, we did not differentiate its usage in the simulation and real data studies. Other parameters were set to default values for these methods.

All methods were run on a shared cluster managed by the Load Sharing Facility (LSF) system. The running time and CPU time of these methods are measured by LSF. Each cluster node is equipped with Intel(R) Xeon(R) 12-core 2.93 GHz CPU and at least 48 GB memory. All files were served by the Lustre file system.

4 MATERIALS

All materials including both simulated and real data are based on the mouse population and consist of paired-end reads with 100 bp length per read. We used C57BL/6J downloaded from Ensembl (Build 70) as the reference genome in all experiments. All methods studied in this article were provided with 74 215 protein-coding annotated transcripts from the Ensembl database. The simulation

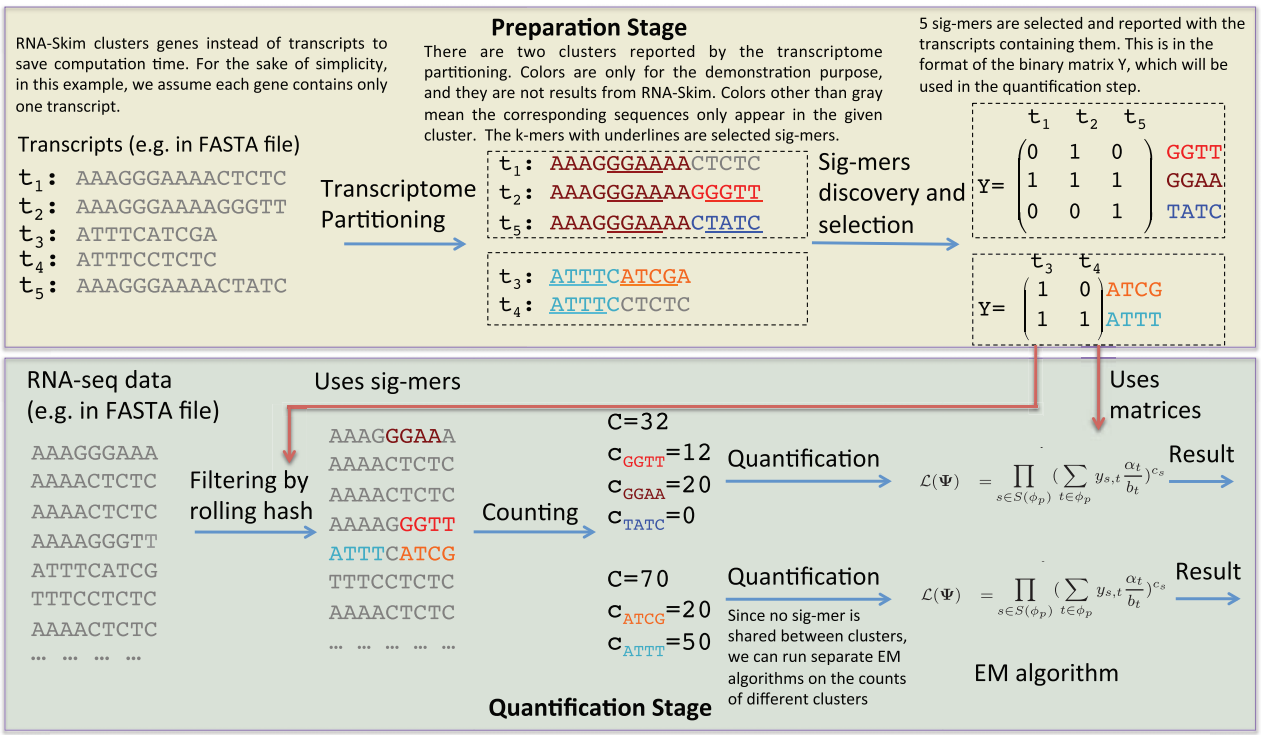

Fig. 3. An illustration of how RNA-Skim works on a toy transcriptome of five transcripts

Table 1. This table compares three different partitions

Type	Number of clusters	Average number of transcripts per cluster	Size of the largest cluster
Transcript	74 215	1	1
Gene	22 584	3.29	39
RNA-Skim	18 269	4.06	6107
Sailfish	1	74 215	74 215

Note: If the partition contains only one cluster of all transcripts, RNA-Skim degenerates to Sailfish. We thus listed it in the table for comparison.

datasets, including 100 mouse samples with the number of reads varying from 20 millions to 100 millions, were generated by the flux-simulator (Griebel *et al.*, 2012) with its default error model enabled. For real datasets, we used the RNA-Seq data from 18 inbred samples and 58 F1 samples derived from three inbred mouse strains CAST/EiJ, PWK/PhJ and WSB/EiJ. The RNA-Seq data was sequenced from mRNA extracted from brain tissues of both sexes and from all six possible crosses (including the reciprocal).

5 RESULTS

In this section, we first compared alternative partition algorithms and how they impact sig-mer selections in RNA-Skim and then furnish a comparison with four alternative methods on both simulated and real data. At last, we demonstrated that RNA-Skim is the fastest method among all considered methods.

5.1 Similarity-based partition algorithm

We compared the result of our similarity-based partition algorithm with those from two alternative ways to partition transcripts: transcript-based partition (every cluster contains a transcript) and gene-based partition (every cluster contains the transcripts from an annotated gene). The similarity threshold γ in our partition algorithm was set to be 0.2 (more details are provided later on the parameter choice). Table 1 compares these partitions on the same transcriptome. The number of clusters generated by our similarity-based partition is 20% fewer than the number of genes. The average number of transcripts per cluster is ~20% more than the average number of transcripts per gene. Most clusters only contain transcripts from a single gene, though the largest cluster contains 6107 transcripts. These transcripts in the largest cluster share a substantial number of k-mers (e.g. from paralogous genes), which need to be examined altogether to accurately estimate their abundance levels. Failing to consider them together (e.g. by using transcript-based or gene-based partitions) will compromise the number of sig-mers that help distinguish transcripts and hence impair the accuracy of transcriptome quantification. Even though this cluster contains many transcripts, it represents <10% of the total number of transcripts.

We used these three types of partitions as the input to the sig-mer discovery method. To evaluate the goodness of a partition, we measured the proportion of each transcript that is covered by

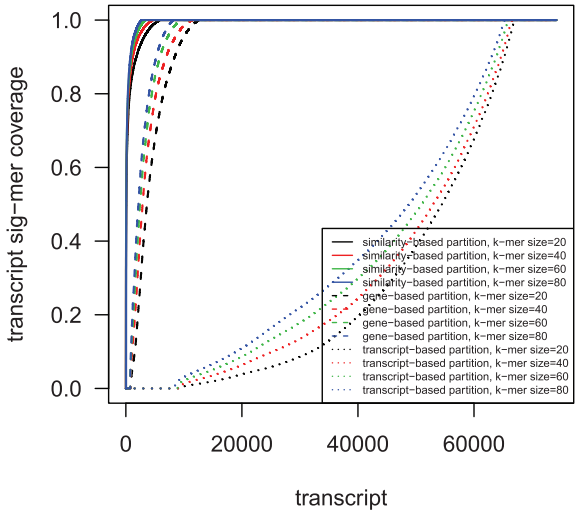

Fig. 4. The distribution sig-mer coverages across all transcripts an ascending order of the sig-mer coverage. The upper the curve is, the better the corresponding partition is

sig-mers and plot the cumulative distribution of all transcripts sorted in ascending order of their *sig-mer coverage* in Figure 4, with varying k-mer sizes. For any transcript, the higher the sig-mer coverage is, the more accurate the abundance estimation will be. Our similarity-based partition is the best: almost all transcripts have at least 80% sig-mer coverage, which pushes the curves to the upper left corner of the plot regardless of the k-mer size. The gene-based partition is slightly worse: ~95% of transcripts have at least 80% sig-mer coverage. The gene-based partition tends to result in low sig-mer coverage for genes sharing similar sequences. The transcript-based partition is the worst for an obvious reason: transcripts from the same genes may share exons and thus the number of sig-mers that can distinguish a transcript may be very small. We also observed that using longer k-mer improves the sig-mer coverage.

In the end, RNA-Skim selects 2 586 388 sig-mers to be used in the quantification stage, and these sig-mers count for <3.5% of 74 651 849 distinguished k-mers used by Sailfish. Because RNA-Skim uses a much smaller set of sig-mers, it is able to use the rolling hash method—a very fast but memory-inefficient method—to count sig-mers in RNA-Seq reads.

5.2 Simulation study

Figure 5 compares the performance of the five methods on the simulated data using four metrics: Pearson's correlation coefficient, Spearman's rank correlation coefficient, significant false-positive rate (SFPR) and significant false-negative rate (SFNR). For brevity, we use Pearson (Truth), Spearman (Truth), SFPR and SFNR to denote these metrics, respectively. The Pearson's correlation coefficient is calculated in a logarithmic scale, using all transcripts whose true and estimated abundance values are >0.01 RPKM. This calculation is the same as that used by Sailfish (Patro *et al.*, 2013). The Spearman's rank correlation is calculated on the set of transcripts whose true abundance values are >0.01 RPKM. The SFPR and SFNR are calculated to assess the estimation distributions on the set of transcripts excluded by

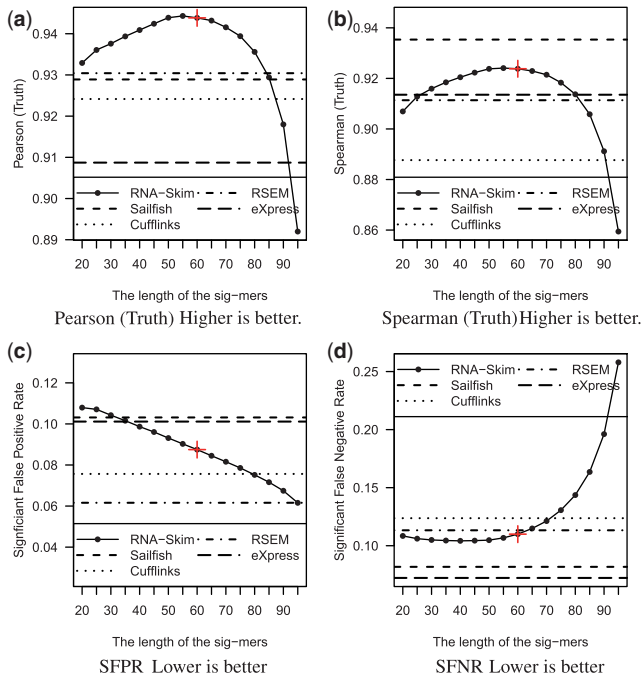

Fig. 5. (**a**), (**b**), (**c**) and (**d**) plot Pearson (Truth), Spearman (Truth), SFPR and SFNR of RNA-Skim as a function of sig-mer length, respectively. For comparison, we also plotted that of the other four methods as the horizontal lines. The reported values are the average across 100 simulated samples. The red crosses indicate the sig-mer length (i.e. 60 bp) used in other experiments in this article

the previous metrics: if a transcript's estimation is >0.1 RPKM, but its true abundance value is <0.01 RPKM (a 10-fold suppression), we call it a *significant false positive*; similarity, if a transcript's estimation is <0.01 RPKM, but its true abundance value is >0.1 RPKM (a 10-fold amplification), we call it a *significant false negative*. There are two reasons that we chose SFPR and SFNR instead of the regular false-positive rate and false-negative rate: first, we prefer the transcripts with relatively large abundance values (>0.1 RPKM) because they are accountable for 99% the RNA-Seq data; second, owing to the noisy nature of RNA-Seq, for the transcripts with small abundance values (<0.01 RPKM), it is difficult to calculate accurately, e.g. both RSEM and Sailfish set the default minimal abundance value to be 0.01 RPKM.

For RNA-Skim, we varied the sig-mer length from 20 to 95 bp. Other methods are presented as horizontal lines for comparisons. Despite the small differences by individual metrics, Figure 5 shows that these five methods exhibit comparable performance: no method outperforms the remaining methods across all metrics and the maximal difference by any metric is within 0.05.

Figure 5(a) and 5(b) show two concave curves of Pearson (Truth) and Spearman (Truth) for RNA-Skim by varying its sig-mer length. There are two factors explaining the concave curves. First, when the sig-mer length increases, sig-mers become more distinct and the sig-mer coverage increases, which improves the correlations between the truth and estimation. Second, for any fixed read length, when we increased the sig-mer length, the probability that a sig-mer is contained by a single read drops, causing the decrease in the number of sig-mers observed in the RNA-Seq data, which may exacerbate the correlations. In summary, there is a clear trade-off on the sig-mer length. Empirically, the best sig-mer length is between 55 to 60, and we thus used 60 in other experiments.

For the same reason, in Figure 5(c) and 5(d), we found that the increase in the sig-mer length affects positively on SFPR, but negatively on SFNR. When the sig-mer length equals 30, it has similar SFPR with Sailfish, but worse SFNR score than Sailfish, indicating that the complete set of k-mers still has its advantage than a small set of sig-mers. However, RNA-Skim is able to use much longer k-mers that Sailfish does not support, so RNA-Skim using longer k-mers can have a better SFPR than Sailfish. Other methods also follow the same inverse correlation: while Sailfish and eXpress are the worst in SFPR among these five methods, they are the best two in SFNR.

Figure 6 shows the Pearson (Truth), Spearman (Truth), SFPR and SFNR as a function of the number of sig-mers used in RNA-Skim. In Figure 6(a), 6(b) and 6(d), when the number of sig-mers increases, the three metrics improve substantially, though at different paces. Figure 6(c) shows no significant change in SFPR for different numbers of sig-mers. This observation suggests that we should use as many sig-mers as possible given available memory space. To ensure RNA-Skim to have similar memory usage to that of other methods, RNA-Skim uses 2.58 million sig-mers. This is also the default setting in other experiments in this article.

Table 2 shows that the metrics do not vary much when using different similarity thresholds. In the simulation study, we varied the similarity threshold γ from 0.06 to 0.28 and observed at most 0.005 change across all metrics. Owing to limited space, the detailed results for the thresholds between 0.06 and 0.28 are omitted.

Figure 7 shows a strong and clear linear correlation between the estimated RPKM scores by RNA-Skim and the true RPKM scores on one simulated sample.

In simulation study, we note that the accuracy of RNA-Skim depends on the sig-mer length and the number of sig-mers, but is insensitive to the threshold γ. When these parameters are chosen properly, RNA-Skim produces similar results to those by other methods.

5.3 Study using real RNA-Seq data

Because the flux simulator cannot simulate RNA-Seq data with bias effects, and there might also be other unknown factors in the real RNA-Seq data that the simulator fails to capture, we also compared RNA-Skim with other methods on real data. Because we do not know the ground truth on real data, we computed the Pearson correlation and Spearman correlation between the results produced by RNA-Skim and one other method, referred to as Pearson (methods) and Spearman (methods) to distinguish from the previous computed correlations between RNA-Skim result and the ground truth.

Figure 8 shows that the distributions of the Pearson (methods) and Spearman (methods) are not significantly different between real data and simulated data. For example, the differences between the mean values of the correlations on both simulated and real data are no more than 0.02 across all methods. This

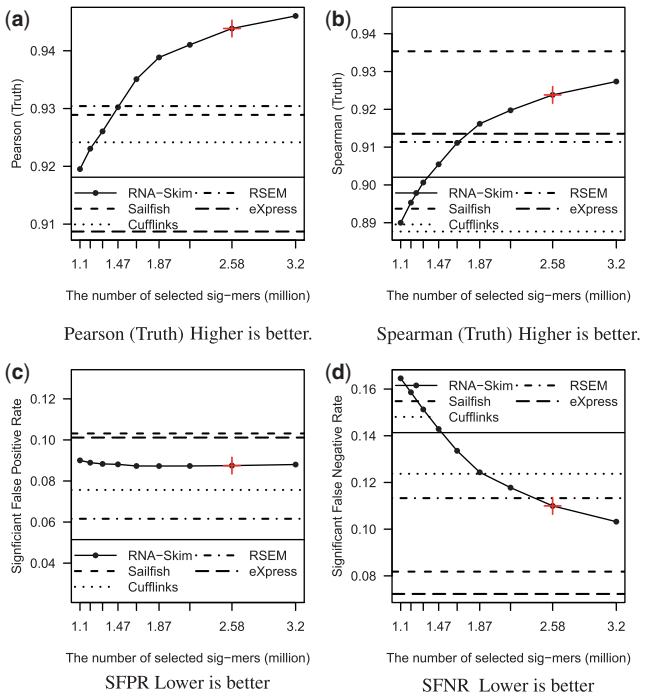

Fig. 6. **(a)**, **(b)**, **(c)** and **(d)** plot Pearson (Truth), Spearman (Truth), SFPR and SFNR as a function of the number of sig-mers used in RNA-Skim, respectively. For comparison, we also showed that of the other four methods as horizontal lines. The reported values are the average across 100 simulated samples. The red crosses indicate the number of sig-mers (i.e. 2.58 million sig-mers) used in other experiments in this article

Table 2. This table shows that the four metrics do not change much for different similarity threshold γ

γ	Pearson	Spearman	SFP	SFN
0.06	0.9438	0.9242	0.0692	0.0233
0.28	0.9440	0.9237	0.0698	0.0235

consistency suggests that the result from RNA-Skim may have similar correlations with the unobserved truth. The slightly wider distribution of the correlations in real data (than that in simulated data) suggests the real data may exhibit more diversity than simulated data. (For the comparison with gene expression data, please see the Supplementary Material).

5.4 Running time

For the preparation stage (including transcriptome partitioning and sig-mer selection), RNA-Skim takes ~3 h to finish on the mouse transcriptome by using a single thread. Most time is spent on calculating the k-mer-based similarities between different pairs of genes. It takes ~10 min to finish sig-mer discovery and selection. It is worth noting that these steps only need to be run once for one population beforehand, and after sig-mers are selected and their connections with transcripts are established, the result can be repeatedly used on quantifying the transcriptome of many samples. Therefore, the running time for the preparation stage is less critical than the running time of the quantification stage, and the one-time investment of 3 h is acceptable.

For the quantification stage, we compared both the running time and the CPU time of these five methods on a real sample with 44 millions of paired-end reads. The running time is the

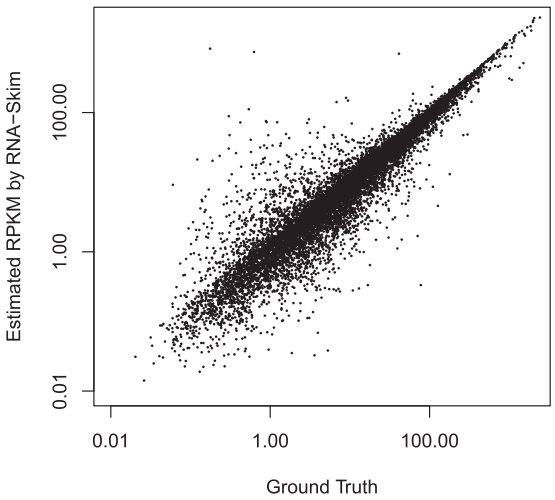

Fig. 7. The scatterplot of the estimated RPKM scores by RNA-Skim versus the true RPKM scores. Both axes are in a logarithmic scale, and all transcripts whose true RPKM or estimated RPKM is <0.01 are omitted

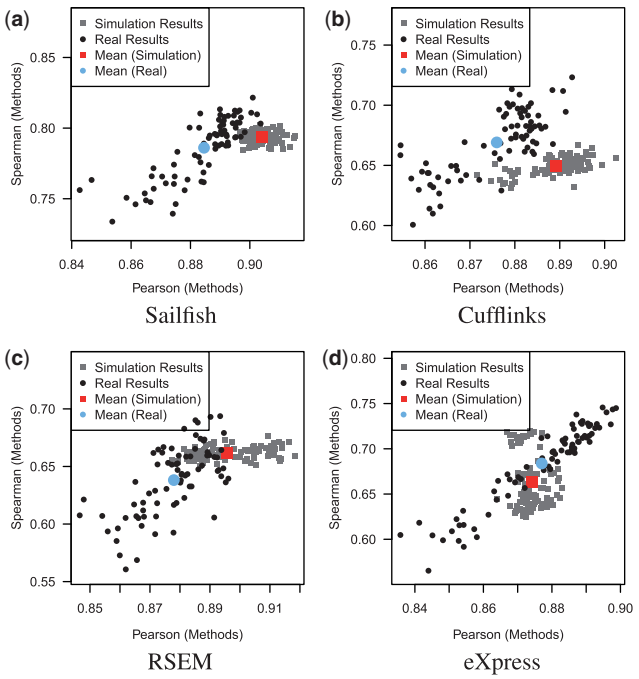

Fig. 8. **(a)**, **(b)**, **(c)** and **(d)** show the distributions of the Pearson (methods) and Spearman (methods) correlations between the results from RNA-Skim and the results from each of the remaining methods on both simulated and real data

Table 3. This table shows the running time of these five methods on a real sample with 44 millions of paired-end reads

Method	Number of threads	Running time (s)	CPU time (s)	Speedup (CPU time)
RNA-Skim	1	592	592	1x
Sailfish	8	972	7005	11.8x
TopHat + Cufflinks	8	12 480	68 834	116x
Bowtie + RSEM	8	17 60	79 222	133x
Bowtie + eXpress	8	13 800	111 273	188x

elapsed time between the start and end of a method, and the CPU time is the total time a method uses on each core of the CPU. For a single thread method, the running time is exactly the same as the CPU time. And for a multi-threading method running on a multi-core CPU, the running time is typically shorter than the CPU time. RNA-Skim is submitted as a single thread method. Sailfish, Cufflinks with TopHat as the aligner and RSEM with Bowtie as the aligner are submitted with multi-threading enabled and requiring eight threads. eXpress is an online algorithm, and it can quantify a streaming input of alignments generated by Bowtie in real time. Bowtie and eXpress use six and two threads for alignment and quantification, respectively.

Table 3 summarizes the running time of all five methods. RNA-Skim is the fastest, ~11× faster than the second best method, Sailfish, on the CPU time. Even when Sailfish uses eight threads, RNA-Skim is ~1.6× faster on the running time by just using one thread. Because the aligner usually consumes lots of computation time, RNA-Skim has >100 times speedup on the CPU time compared with Cufflinks, RSEM and eXpress.

Overall, these results demonstrate that RNA-Skim provides comparable accuracy with other methods on both simulated and real data, using a much shorter running time.

6 DISCUSSION AND CONCLUSION

We introduced RNA-Skim, a lightweight method that can rapidly and efficiently estimate the transcript abundance levels in RNA-Seq data. RNA-Skim exploits the property of sig-mers, significantly reducing the number of k-mers used by the method and the scale of the optimization problem solved by the EM algorithm. Based on our benchmark, it is at least 10× faster than any alternative methods. To the best of our knowledge, the design principle of almost all existing methods is to use as much data as possible for RNA-Seq quantification. Our results are encouraging, in the sense that they demonstrate a different, yet promising, direction of building a much faster method by discovering and using only informative and reliable features—the counts of sig-mers in RNA-Seq data.

Currently, the annotation databases are incomplete and still under development. Aligners and alignment-dependent RNA-Seq methods are commonly used to allow unknown transcript discovery, which will further improve the completeness and accuracy of the annotation databases. The performance of tools like Sailfish and RNA-Skim depends on the quality of the annotation database. Their accuracy is likely to improve when annotation databases become complete or nearly complete in the future. They will become better choices when we have a better understanding of transcriptome and transcript discovery task becomes less important.

Because RNA-Skim is still under development, there are several directions to further improve its performance. (i) RNA-Skim uses a simple hash table implementation without any optimization on the memory usage. We will investigate advanced data structures enabling better memory utilization. (ii) Currently, the sig-mer selection algorithm in RNA-Skim only ensures uniform coverage. In the future, we will explore variable selection techniques to select fewer but more informative sig-mers. (iii) The current version of RNA-Skim does not have built-in bias correction capability, even though it already produces results comparable with the state-of-the-art methods with bias correction on real data. We plan to incorporate bias correction in the next version of RNA-Skim, which is likely to improve the performance further. In addition, we also plan to support multi-thread implementation and deploy RNA-Skim in differential expression analysis. We are optimistic that, when we add the multi-threading capability to RNA-Skim, the running time will be further improved.

ACKNOWLEDGEMENTS

We would like to thank those center members who prepared and processed samples as well as those who commented on and encouraged the development of RNA-Skim, in particular, Leonard McMillan, Vladimir Jojic, William Valdar and Yunjung Kim. We also would like to thank three anonymous reviewers for their thoughtful comments.

Funding: This work was funded by NIH R01HG006703, NIH P50 GM076468-08 and NSF IIS-1313606.

Conflict of Interest: none declared.

REFERENCES

Au,K.F. *et al.* (2010) Detection of splice junctions from paired-end RNA-seq data by SpliceMap. *Nucleic Acids Res.*, **38**, 4570–4578.

Bloom,B.H. (1970) Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, **13**, 422–426.

Dadgar,A. (2013) Bloomd library. https://github.com/armon/bloomd (14 May 2014, date last accessed).

Flicek,P. *et al.* (2011) Ensembl 2012. *Nucleic Acids Res.*, **40**, D84–D90.

Fu,C.P. *et al.* (2014) An alignment-free regression approach for estimating allele-specific expression using RNA-seq data. In Sharan,R. (ed.) *Research in Computational Molecular Biology*. Vol. 8394, Springer, pp. 69–84.

Google. (2013) Protocal buffers. https://code.google.com/p/protobuf/ (14 May 2014, date last accessed).

Grabherr,M.G. *et al.* (2011) Full-length transcriptome assembly from RNA-Seq data without a reference genome. *Nat. Biotechnol.*, **29**, 644–652.

Griebel,T. *et al.* (2012) Modelling and simulating generic rna-seq experiments with the flux simulator. *Nucleic Acids Res.*, **40**, 10073–10083.

Guttman,M. *et al.* (2010) Ab initio reconstruction of cell type–specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs. *Nat. Biotechnol.*, **28**, 503–510.

Hsieh,W. (2013) Stringpiece. https://chromium.googlesource.com/chromium/ (14 May 2014, date last accessed).

Karp,R.M. and Rabin,M.O. (1987) Efficient randomized pattern-matching algorithms. *IBM J. Res. Dev.*, **31**, 249–260.

Kurtz,S. *et al.* (2008) A new method to compute K-mer frequencies and its application to annotate large repetitive plant genomes. *BMC Genomics*, **9**, 517.
Langmead,B. *et al.* (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol.*, **10**, R25.
Le,H.S. *et al.* (2013) Probabilistic error correction for RNA sequencing. *Nucleic Acids Res.*, **41**, e109.
Li,B. and Dewey,C.N. (2011) RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. *BMC Bioinformatics*, **12**, 323.
Marcais,G. and Kingsford,C. (2011) A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. *Bioinformatics*, **27**, 764–770.
Melsted,P. and Pritchard,J.K. (2011) Efficient counting of k-mers in DNA sequences using a bloom filter. *BMC Bioinformatics*, **12**, 333.
Nicolae,M. *et al.* (2011) Estimation of alternative splicing isoform frequencies from RNA-Seq data. *Algorithms Mol. Biol.*, **6**, 9.
Ozsolak,F. and Milos,P.M. (2010) RNA sequencing: advances, challenges and opportunities. *Nat. Rev. Genet*, **12**, 87–98.
Pachter,L. (2011) Models for transcript quantification from RNA-Seq. *arXiv.org,* 1104.3889v2.
Patro,R. *et al.* (2014) Sailfish: alignment-free Isoform quantification from RNA-seq reads using lightweight algorithms. *Nat. Biotech.*, **32**, 462–464.
Pruitt,K.D. *et al.* (2007) NCBI reference sequences (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. *Nucleic Acids Res.*, **35**, D61–D65.
Rizk,G. *et al.* (2013) DSK: k-mer counting with very low memory usage. *Bioinformatics*, **29**, 652–653.
Roberts,A. and Pachter,L. (2013) Streaming fragment assignment for real-time analysis of sequencing experiments. *Nat. Methods*, **10**, 71–73.
Trapnell,C. *et al.* (2009) TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics*, **25**, 1105–1111.
Trapnell,C. *et al.* (2010) Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat. Biotechnol.*, **28**, 516–520.
Trapnell,C. *et al.* (2012) Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. *Nat. Protocols*, **7**, 562–578.
Turro,E. *et al.* (2011) Haplotype and isoform specific expression estimation using multi-mapping RNA-seq reads. *Genome Biol.*, **12**, R13.
Uziela,K. and Honkela,A. (2013) Probe region expression estimation for RNA-seq data for improved microarray comparability. *ArXiv e-prints,* 1304.1698v1.
Wang,K. *et al.* (2010) MapSplice: accurate mapping of RNA-seq reads for splice junction discovery. *Nucleic Acids Res.*, **38**, e178.
Wang,Z. *et al.* (2009) RNA-Seq: a revolutionary tool for transcriptomics. *Nat. Rev. Genet.*, **10**, 57–63.
Zhang,Z. *et al.* (2013) GeneScissors: a comprehensive approach to detecting and correcting spurious transcriptome inference owing to RNA-seq reads misalignment. *Bioinformatics*, **29**, 291–299.