Towards Selective-Alignment: Bridging the Accuracy Gap between Alignment-Based and Alignment-Free Transcript Quantification

Hirak Sarker*
Stony Brook University
Stony Brook, New York
hsarkar@cs.stonybrook.edu

Mohsen Zakeri*
Stony Brook University
Stony Brook, New York
mzakeri@cs.stonybrook.edu

Laraib Malik
Stony Brook University
Stony Brook, New York
lmalik@cs.stonybrook.edu

Rob Patro†
Stony Brook University
Stony Brook, New York
rob.patro@cs.stonybrook.edu

ABSTRACT

We introduce an algorithm for selectively aligning high-throughput sequencing reads to a transcriptome, with the goal of improving transcript-level quantification in difficult or adversarial scenarios. This algorithm attempts to bridge the gap between fast non-alignment-based algorithms and more traditional alignment procedures. We adopt a hybrid approach that is able to produce accurate alignments while still retaining much of the efficiency of non-alignment-based algorithms. To achieve this, we combine edit-distance-based verification with a highly-sensitive read mapping procedure. Additionally, unlike the strategies adopted in most aligners which first align the ends of paired-end reads independently, we introduce a notion of co-mapping. This procedure exploits relevant information between the "hits" from the left and right ends of paired-end reads before full mappings for each are generated, improving the efficiency of filtering likely-spurious alignments. Finally, we demonstrate the utility of selective alignment in improving the accuracy of efficient transcript-level quantification from RNA-seq reads. Specifically, we show that selective-alignment is able to resolve certain complex mapping scenarios that can confound existing non-alignment-based procedures, while simultaneously eliminating spurious alignments that fast mapping approaches can produce. Selective-alignment is implemented in C++11 as a part of *Salmon*, and is available as open source software, under GPL v3, at: https://github.com/COMBINE-lab/salmon/tree/selective-alignment

CCS CONCEPTS

• **Applied computing** → **Computational biology**; **Computational transcriptomics**;

KEYWORDS

Mapping, Alignment, RNA-seq, Quantification, Selective alignment

ACM Reference Format:
Hirak Sarker, Mohsen Zakeri, Laraib Malik, and Rob Patro. 2018. Towards Selective-Alignment: Bridging the Accuracy Gap between Alignment-Based and Alignment-Free Transcript Quantification. In *9th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics (ACM-BCB'18), August 29-September 1, 2018, Washington, DC, USA,* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 22 pages. https://doi.org/10.1145/3233547.3233589

1 INTRODUCTION

Since the introduction of high-throughput, short read sequencing technologies, many algorithms and tools have been designed to tackle the problem of aligning short sequenced reads to a reference genome or transcriptome accurately and efficiently. While there exist "full-sensitivity" aligners (e.g. mrFAST [1], mrsFAST [8, 33], RazerS3 [31], Masai [27]) which guarantee to find all reference positions within a given edit-distance threshold of a read sequence, the most widely-used tools ([11], [17]) employ heuristic strategies to enable much faster alignment of reads in the typical case (i.e., only a small number of easy-to-find candidate locations exist for each alignment). The common procedure followed by these tools for aligning reads can be divided into two major steps. The first is finding potential alignment locations for the read using a pre-processed index that is generated from the reference genome or transcriptome. Then, in the second step, the potential locations are filtered, and reads are aligned to the positions that pass the initial filtering, based on a variety of heuristics. The exact method for generating the initial index varies for each tool. For example, tools like Bowtie [12], Bowtie2 ([11]), BWA [15], and BWA-mem [14] use Burrows-Wheeler transformation (BWT) based indices, whereas, k-mer based indices are used by tools such as Subread-aligner [17], Maq [16], SNAP [35], and GMAP and GSNAP [32].

Similarly, the heuristic for choosing the most probable locations is also different. However, each method is based on the principle of

*Equal contribution
†Corresponding Author

ACM-BCB'18, August 29-September 1, 2018, Washington, DC, USA

ACM ISBN 978-1-4503-5794-4/18/08…$15.00
https://doi.org/10.1145/3233547.3233589

trying to find the reference loci that support the best (or near-best) alignment score between the read and the reference. Repeating this for a large number of reads comes with a considerable cost in terms of computation. Some tools, like STAR [6], considerably speed up the alignment process by combining efficient heuristics with data structures (like the uncompressed suffix array) that trade working memory for exact pattern lookup speed. Recently, tools like HISAT [10] have also demonstrated that cache-friendly compressed indices (the hierarchical FM index in this case) can provide similarly efficient pattern search, even with a very moderate memory budget. The alignment of sequenced reads to the reference is the first step in pipelines leading to various downstream studies, such as estimation of transcript abundances and differential expression analysis, calculation of splicing rates [26, 30], and detection of fusion events [5, 21].

While alignment is a staple of many genomic analyses, it sometimes represents more information than is actually necessary to address the analysis at hand. For example, recent tools like *Sailfish* [23], *RNAskim* [37], *kallisto* [3], *Salmon* [22], and *Fleximer* [9], demonstrate that accurate quantification estimates can be obtained without all of the information encoded in traditional alignments. By avoiding traditional alignment procedures, these tools are much faster than their alignment-based counterparts. Furthermore, by building the mapping phase of the analysis directly into the quantification task, they dispense with the need to write, store, and read, large intermediate alignment files. However, these non-alignment-based tools, while highly-efficient, have the disadvantage of potentially losing sensitivity or specificity in certain cases where alignment-based methods would perform well. For example, in the presence of paralogous genes, with high sequence similarity, there is an increased probability that the mapping strategies employed by such tools, and the efficient heuristics upon which they rely, will mis-map reads between the paralogs (or return a more ambiguous set of mapping locations than an aligner, which expends computational resources to verify the returned alignments) [2]. Similarly, in the case of *de novo* assemblies, poorly assembled contigs may have a larger number of mis-mapped reads due to lower sensitivity (here, the issue would be primarily due to aberrant exact matches masking the true origin of a read).

Other than suffering from spurious mappings, these fast non-alignment-based approaches can also miss true mappings of a read in rare cases where errors are positioned adversarially on the read. An obvious case of losing the true mapping is if a read contains no subsequence of sufficient length from the true transcript. In another case, the true mapping of the read might be lost from the set of potential mapping loci due to the greedy nature of the mapping procedures. For some reads, multiple positions might be found on the same transcript where the read maps. In such cases, improved heuristics are required to address these challenges. In this paper, we present a novel algorithm, selective-alignment, that extends quasi-mapping to compute and store edit distance information where necessary. The reads for alignment are chosen based on certain criteria calculated during mapping. This strikes a balance between speed and accuracy; not compromising the superior speed of non-alignment-based algorithms, while also addressing some of the challenges mentioned above. Specifically, the motivation for selective-alignment is to enhance both the sensitivity and specificity of fast mapping algorithms by reducing or eliminating cases where spurious exact matches mask true mapping locations as well as cases where small exact matches support otherwise poor alignments. Selective-alignment algorithm is built atop the framework of *RapMap* [29], which uses an index that combines a fixed-length prefix hash table and an uncompressed suffix array [19]. We introduce a coverage-based consensus scheme to identify critical read candidates for which alignment is necessary.

Furthermore, we explore the challenging cases where the heuristics employed by fast mapping algorithms may fail to locate the correct locations for a read, while the traditional aligners do not. We do this by making a number of modifications to the underlying mapping algorithm to increase its sensitivity. We also introduce multiple filters and scoring schemes designed to eliminate spurious mappings (i.e., situations where the best mapping is unlikely to represent the true origin of the read). In this work, we focus on the effect of selective-alignment on transcript quantification estimates, and we leave a thorough evaluation of the alignment qualities themselves as future work. In particular, the evaluation of alignment qualities is considerably complicated due to prevalent multi-mapping in the transcriptome.

2 METHODS

The process of selective-alignment builds upon the basic data structures of Srivastava et al. [29], but there are a number of important algorithmic distinctions. Specifically, compared to the algorithm of RapMap, selective-alignment introduces the k-safe longest common prefix (k-safe-LCP), replaces maximum mappable prefixes (MMP) with maximum mappable safe prefixes (MMSP), increases mapping sensitivity by adopting a different consensus rule over hits, makes use of co-mapping to filter and prioritize potential mapping loci, introduces a new mechanism for selecting a mapping position for a read when multiple candidates exist on the same transcript, and, finally, introduces a fast edit distance filter (with alignment subproblem caching) to remove spurious mappings and provide quality scores for mappings that pass the filter. A block diagram of different steps used in the selective alignment pipeline is shown in S1.

Below, we recapitulate the basic data structures and concepts that will be required to explain the selective alignment algorithm. To start with, the index built on the transcriptome in selective-alignment is a combination of a suffix array and a hash table constructed from unique k-mers (substrings of length k) and suffix array intervals. The suffix array of a sequence, T— denoted $SA(T)$— is an array of starting positions of all suffixes from T in the original sequence. The values in the array are sorted lexicographically by the suffixes they represent. Therefore, all suffixes starting with the same prefix are located in adjacent positions of the suffix array. Formally, given a suffix array, $SA(T) = \Lambda$, constructed from the transcriptome sequence, T, we construct a hash table, h, that maps each k-mer, κ, to a suffix array interval, $\mathtt{I}(\kappa) = [b, e)$, if and only if all the suffixes within interval $[b, e)$ contain the k-mer κ as a prefix. We define $\Lambda[i]$, for every $0 \leq i \leq |\Lambda|$, to be the suffix $T[SA[i]]$ (i.e., the suffix of T starting from position $SA[i]$). In the selective-alignment index, in addition to suffix array intervals, we store two extra pieces of information for each interval; the longest

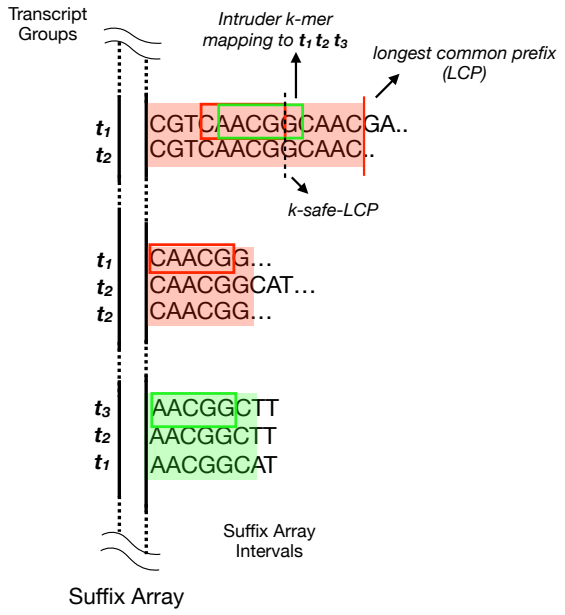

Figure 1: Calculation of k-safe-LCP from the suffix array data structure. The transcripts present in each suffix array interval determine the relevant transcript sets, and which k-mers will be considered as intruders. To determine the k-safe-LCP of the suffix array interval starting with the k-mer *CGTCA*, we check all the k-mers sequentially. Some k-mers do not yield an interval with transcripts other than t_1 and t_2, e.g., *CAACG*. Detection of a k-mer (*AACGG*) (as intruder) that maps to suffix array interval labeled (t_1, t_2, t_3) determines the k-safe-LCP here.

common prefix (LCP) and the k-safe-LCP corresponding to the interval. The longest common prefix (LCP) of any pair of suffixes in the suffix array is simply the length of the prefix that these suffixes share. Though the LCPs for the suffixes in the suffix array can be pre-computed, we instead compute them on demand using a linear scan. These methods are detailed below. As an alternate to the suffix array and the LCP array, one could make use of other data structures which also encode this information. For example, the recently-introduced method, Fleximer [9] makes use of the suffix tree for selecting informative sig-mers [37] from the transcriptome, and matching reads against them.

2.1 Defining and computing k-safe-LCPs

Here, we formally define the concept of k-safe-LCPs (see figure 1). The determination of k-safe-LCPs starts by labeling each suffix array interval with the length of its corresponding longest common prefix and the associated transcript set it represents. Formally, $\texttt{LCP}(\Lambda[b], \Lambda[e-1])$ for an interval $[b, e)$ is the length of the common prefix of the suffixes $\Lambda[b]$ and $\Lambda[e-1]$. Given k-mer κ, where $\kappa \in \mathcal{K}$ and $\mathcal{K}$ is the set of all k-mers from the reference sequence T, and the related interval $\texttt{I}(\kappa) = [b, e)$, for all $p \in [b, e)$, we consider each transcript t such that the suffix $\Lambda[p]$ starts in transcript t in the concatenated text. Then, for this interval, we can construct a set $C^{\kappa} = \{t_i, t_j, \dots\}$, which denotes the set of distinct transcripts that appear in the suffix array interval, indicated by κ. We note that this notion discards duplicate appearances of the same transcript in this interval.

We compute the k-safe-LCP for an interval indicated by k-mer κ_i iteratively. The initial length for the k-safe-LCP of the interval is k, length of a k-mer. We check, sequentially, each of the k-mers in the longest common prefix of the interval. For each new k-mer, the k-safe-LCP is increased by one character. We terminate the k-safe-LCP extension if any of the following conditions is encountered: (1) we reach the last k-mer contained in the LCP of this interval, (2) we encounter a k-mer κ_j such that $C^{\kappa_j} \not\subseteq C^{\kappa_i}$ or (3) we encounter a k-mer κ_j such that the reverse complement of κ_j appears elsewhere in the transcriptome. When we encounter case (2) or (3), we call the k-mer κ_j an *intruder*. That is, the k-mer will potentially alter our belief about the set of potential transcripts to which a sequence containing this k-mer maps (by strictly expanding this set), or the orientation with which it maps to the transcriptome. We denote the k-safe-LCP of a particular interval $\texttt{I}(\kappa_i)$ as $\texttt{k-safe-LCP}(\texttt{I}(\kappa_i))$.

As shown in figure 1, the k-safe-LCP determination for the top suffix array interval starts with matching k-mers within the longest common prefix. The k-mer "CAACG" maps to a suffix array interval labeled with (t_1, t_2). The next k-mer "AACGG", on the other hand, maps to a suffix array interval (shaded in green) labeled with (t_1, t_2, t_3), thereby implying the k-safe-LCP, shown as a dotted line. For each k-mer in the hash table, we store the length of the LCP and k-safe-LCP, along with the corresponding suffix array interval.

2.2 Discovering relevant suffix array intervals

As shown in figure 2, the selective-alignment approach can be broken into three major steps: collecting suffix array intervals, co-mapping, and selecting the high quality mappings. Gathering the suffix array intervals for a query read closely follows the quasi-mapping approach. It involves iterating over the read from left to right and repeating two steps. First, hashing a k-mer from the read sequence and then discovering the corresponding suffix array intervals. The process of k-mer lookup is aided by the k-safe-LCP stored in the index (discussed in 2.1). The inbuilt lexicographic ordering of the suffixes in the suffix array, and the computed k-safe-LCP values of intervals enable safely extending k-mers to longer matches without the possibility of masking potentially-informative substring matches. Given a matching k-mer, κ_r, from the read sequence r, we extend the match to find the longest substring of the read that matches within $\texttt{k-safe-LCP}(\texttt{I}(\kappa_r))$. The matched substring can be regarded as maximum mappable prefix (MMP) [6], that resides within the established k-safe-LCP. We call this a maximal mappable safe prefix (MMSP — eliding k where implied). For a k-mer, κ_r, and interval, $[b, e)$, we note that $\texttt{k-safe-LCP}(\texttt{I}(\kappa_r)) \geq \ell_{MMSP_{\kappa_r}}$, where $\ell_{MMSP_{\kappa_r}}$ is the length of $MMSP_{\kappa_r}$, the MMSP between the read's suffix starting with κ_r and the interval $\texttt{I}(\kappa_r)$. The next k-mer lookup starts from the $(MMSP_{\kappa_r} - k + 1)$-th position. By restricting our match extensions to reside within the MMSP, we ensure that we will not neglect to query any k-mer that might *expand* the set of potential transcripts where our read may map. We note here both the theoretical and practical relation between the *MMSP* matching procedure, and the concept of a uni-MEM, as introduced by Liu et al.

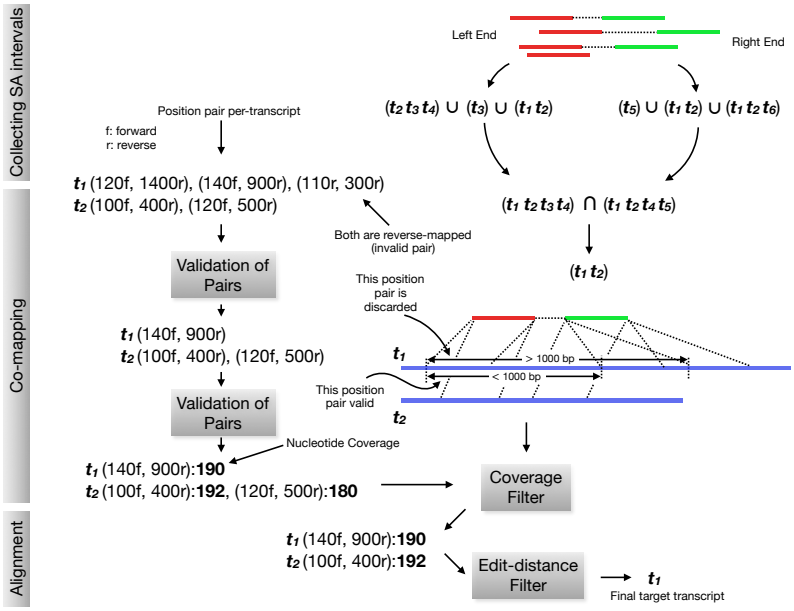

Figure 2: The three main steps of the selective-alignment process are demonstrated here. First, suffix array "hits" are collected. Then, in co-mapping, spurious mappings are removed by the orientation filter and then distance filter. At most a single locus per-transcript is selected based on the coverage filter. Finally, an edit-distance-based filter is used to select the valid target transcripts.

[18]. The k-safe-LCP for suffix array intervals are closely related to the lengths of unipaths in the reference de Bruijn graph of order k. Thus, our procedure for finding *MMSP*s, that limits match extension by the k-safe-LCP, is similar to the uni-MEM seed generation procedure described in deBGA [18], with the distinction that in our method, we only consider extending seeds in one direction, and that we also choose not to terminate the k-safe-LCP when the set of implied reference transcripts corresponding to the interval decreases in cardinality.

Given all the suffix array intervals collected for a read end (i.e. one end of a paired-end read), we take the *union* of all the transcripts they encode. Formally, if a read r maps to suffix array intervals labeled with $C^{r_1}, \ldots, C^{r_n}$, then we consider all transcripts in the set $C^{r_1} \cup C^{r_2} \cup \ldots \cup C^{r_n}$, and the associated positions implied by the suffix array intervals. As shown in figure 2; this step is done before co-mapping.

We adopt a heuristic to avoid excessive k-mer lookups when we encounter a mismatch. When extension of an MMP is no longer possible, it is most probable that the mismatch results from an error in the read. If the mismatch is due to the presence of an error, then checking each k-mer overlapping this error can be a costly process. Instead, we move forward by a distance of $k/2$ in the read, and check the k-mer from the read such that the mismatch occurs in the middle position. If this k-mer lookup leads to another suffix array interval, we continue with the MMP extension process there; otherwise, we move again to the first k-mer that does not overlap this mismatch position. We observe that, in practice, the k-safe-LCP, and hence the MMSP lengths can be quite large (Figure 3).

2.3 Co-Mapping

After collecting the suffix array intervals corresponding to left and right ends of the read, we wish to exploit the paired-end information in determining which potential mapping locations might be valid. Hence, from this step onward, we use the joint information for determining the position and target transcripts. Given the suffix array intervals for individual ends of a paired-end read, the problem of aligning both ends poses a few challenges. First, a single read can map to multiple transcripts, and we wish to report all equally-best loci. Second, there can be multiple hits from a read on a single transcript (e.g., if a transcript contains repetitive sequence), and extra care must be taken to determine the correct mapping location. Finally, there may be hits that do not yield high-quality alignments (i.e. long exact matches that are nonetheless spurious). To address the first and third points, we employ an edit distance filter to discard spurious and sub-optimal alignments. To address the second challenge, we devise a consensus strategy to choose at most one unique position from each transcript.

Before applying the above mentioned strategy, we remove transcripts that do not contain hits from both the left and right ends of the read. Formally, given two ends of a read r, r^{e_1} and r^{e_2}, and the corresponding suffix array intervals labeled with $C^{r_1^{e_1}}, \ldots, C^{r_n^{e_1}}$ and $C^{r_1^{e_2}}, \ldots, C^{r_m^{e_2}}$ respectively, we only consider transcripts present in the set $(C^{r_1^{e_1}} \cup \ldots \cup C^{r_n^{e_1}}) \cap (C^{r_1^{e_2}} \cup \ldots \cup C^{r_m^{e_2}})$. We further refine this set by checking the validity of the alignments these hits might support. Currently, we use two validity checks illustrated in figure 2. First, we apply an orientation-based check, and second, we employ a distance-based check. The orientation check removes potential mappings which have an orientation inconsistent with the

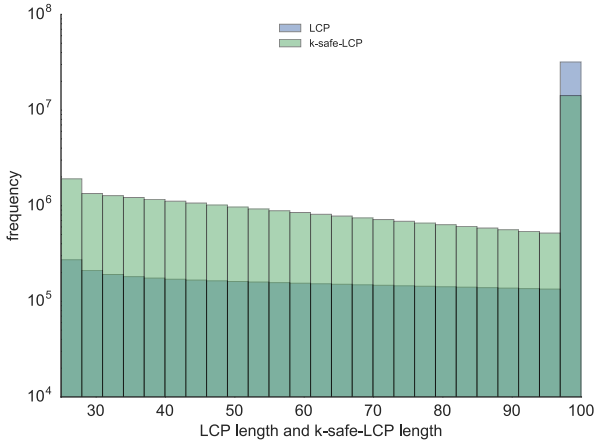

Figure 3: The distribution of k-safe-LCP lengths and LCP lengths are similar and tend to be large in practice (human transcriptome). Here, we truncate all lengths to a maximum value of 100 (so that any LCP or k-safe-LCP longer than 100 nucleotides is placed in the length 100 bin).

underlying sequencing library type (e.g., both ends of a read mapping in the same orientation). The distance check removes potential alignments where the implied distance between the read ends is larger than a given, user-defined threshold (1,000 nucleotides by default).

2.3.1 Coverage based consensus. In selective-alignment, the potential positions on a transcript are scored by their individual coverage on the target transcript. Figure 4 depicts the mechanism of choosing the best postion on a transcript from multiple probable mappings to the same transcript. The coverage mechanism employed in selective-alignment makes use of the MMSP lengths collected during a prior step of the algorithm rather than simply counting k-mers. In figure 4, the transcript t_2 has two potential mapping positions given the reads: position 10 and 20. The coverage consensus mechanism selects position 20 over position 10 due to the higher coverage by tiling MMSPs on the read.

2.3.2 Selecting the best candidate transcripts. Once the positional ambiguity within a transcript is resolved, the next step is selecting the best candidate transcripts from a set of mappings. Since mapping relies on finding exact matches, the length of the matched subsequence between the read and reference can sometimes be misguiding when comparing different candidate transcripts. That is, the transcripts with the longest exact matches do not always su A block diagram of the steps described below are depicted in Figure pport optimal alignments for a read. At this point in our procedure, we follow the approach taken by many conventional aligners, and use an existing optimal alignment algorithm to compute the edit distance, by which we select the best candidate transcripts.

When performing alignment, we assume that a given read aligns starting at the position computed in the previous steps. This helps us to reduce the search space within the transcript where we must consider aligning the read, and thereby considerably reduces the cost of alignment. To align the read at a specific position on the transcript and calculate the edit distance between them, we use *Myer's* bounded edit distance bit-vector algorithm [20], as implemented in `edlib` [28]. For a fixed maximum allowable edit distance, this algorithm is linear in the length of the read. We note that the bounded edit distance algorithm we employ will automatically terminate an alignment when the required edit distance bound is not achievable.

We remove all alignments with edit distance greater than a user-provided threshold. This is similar to the approach used by many existing aligners, and allows us to specify that even the best mapping for a given read may have too many edits to believe that it reasonably originated from a known transcript in the index. An appropriate threshold should be based on the expected error rate of the instrument generating the sequenced reads, and a very low threshold can lead to a decreased mapping rate.

2.3.3 Enhancement of quantification accuracy based on edit distance. We investigated the effect of incorporating edit distance in downstream quantification. Since we integrated the selective-alignment scheme into the quantification tool *Salmon* [22], the edit distance scores from selective-alignment can be used as a new parameter to *Salmon*'s inference algorithm.

In the framework of abundance estimation, we define the conditional probability of a generating a particular fragment, f_j, given that it comes from a specific transcript, t_i, as $P(f_j \mid t_i)$. Given the edit distance between the fragment and the transcript, we can incorporate this parameter into this conditional probability. *Soft* filtering introduces a new term in the conditional probability based on $d_{i,j}$, which is the sum of the edit distances between the read ends of fragment f_j and transcript t_i. We set this probability according to an exponential function, $P(a_j|f_j, t_i) = e^{-4d_{i,j}}$. The aggregate of threshold filtering and *soft* filtering can be described as follows:

$$\Pr\left(a_j \mid d_{i,j}, t_i\right) = \begin{cases} 0 & d_{i,j} > threshold \\ e^{-4d_{i,j}} & d_{i,j} \leq threshold \end{cases}. \quad (1)$$

2.4 Shared LCPs prevents redundant alignments

Exploiting the common subsequences in the transcriptome is instrumental to the superior speed of fast mapping, non-alignment-based tools. Reads generated from exonic sequences common to multiple transcripts from the same gene or paralogous genes are the main source of ambiguous mappings. As we rely on the suffix array data structure to obtain the initial set of transcripts to which a read maps, there are cases where exactly identical reference sequences all act as mapping targets for the read. For a suffix array interval $[b, e)$, we identify such common subsequences by examining the *longest common prefix* (LCP) of the interval. If the length of the LCP is equal or greater than the length of the read, then the actual alignment against the underlying reference at these positions will be identical. We observed (Table 1) that for almost half of the read-transcript pairs, the alignment process can be avoided. Note that if the read sequence shares a complete match with the common prefix, meaning that maximum mappable safe prefix length is equal to read length (i.e., the read matches the reference exactly at some set of

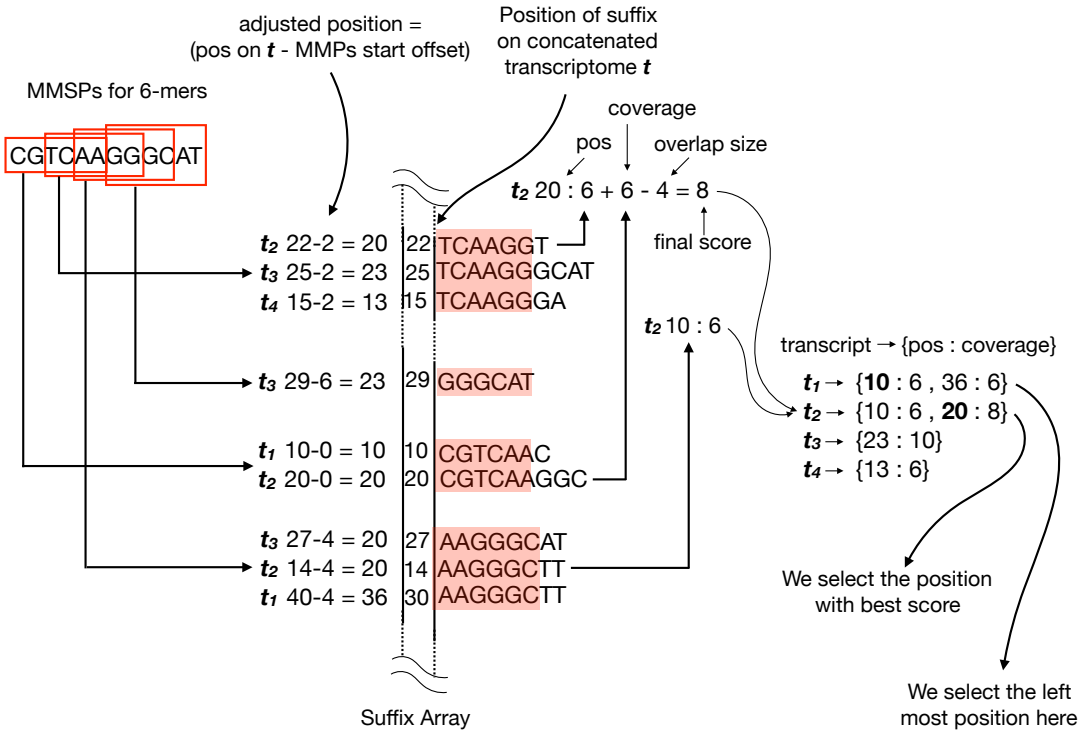

Figure 4: The MMSPs corresponding to a read are derived from multiple suffix array intervals. Here, all MMSPs happen to be of length k as LCPs are of size k. The coverage scheme finds out the exact positions on each transcript by adjusting the starting position of the MMSPs. The total score takes into account the positions where matches overlap. The final position is chosen by selecting the locus with maximum coverage.

Table 1: The percentage of hits that skip the full alignment process on five different experimental samples, due to extension by the maximum mappable safe prefix (MMSP), or projection of duplicate alignments given the longest common prefix (LCP) sequences.

Sample (SRR121)	5996	5997	5998	5999	6000
Skipped alignments	50.36%	54.85%	47.92%	48.06%	50.80%

positions), we can also bypass the Meyer's edit distance algorithm call completely.

Caching alignment sub-problems further avoids redundant work. We also extend a similar idea to the scenario where only part of the reference sequence is shared between references. Specifically, when performing an alignment between anchoring exact matches, we store the result in a hash table where the key is a tuple $(i, j, h(i', j'))$ and the associated value is the computed edit distance. Here, i and j denote the start and end of the read interval being aligned and i' and j' denote the start and end of the reference sequence; $h(i', j')$ is a hash of the corresponding reference sequence (we use `xxhash` [4]). This allows us to detect when a redundant alignment sub-problem for a read is shared between references, and to reuse the cached result in such cases.

3 RESULTS

To evaluate the effectiveness of selective-alignment, we coupled it with the quantification tool *Salmon* (branching from the v0.9.1 release). This enables us to measure the effect of different alignment based and non-alignment based algorithms on transcript-level quantification results directly, holding the statistical estimation procedure fixed. We also include *kallisto* (v0.43) in our benchmarks, which provides a perspective on pseudoalignment-based quantification. Furthermore, we compare the performance of selective-alignment with the recent, fast, hashing and alignemnt-based, abundance estimation tool (currently un-published) *Hera* [1]. We note, this is an early version of the *Hera* (v1.2) software, which is already performing very well in our testing, but is subject to changes and improvements. Given it's impressive performance (in both time and accuracy), we decided to include *Hera* in our comparisons with the consent of its authors (personal communications). We measure the Spearman correlation and Mean Absolute Relative Differences (MARD) of read counts as performance metrics when comparing the different methods (further metrics are also provided for some of the experiments in the supplementary material). All experiments were performed on an Intel(R) Xeon(R) CPU (E5-2699 v4 @2.20GHz with 44 cores and 56MB L3 cache) with 512GB RAM and a 4TB TOSHIBA MG03ACA4 ATA HDD running ubuntu 16.10 and each method was run using 16 threads.

[1] https://github.com/bioturing/hera

Mutation Rate	Correlation (Spearman)					MARD				
	Kallisto	Hera	Selective Alignment	STAR Salmon	Bowtie2 Salmon	Kallisto	Hera	Selective Alignment	STAR Salmon	Bowtie2 Salmon
0.01	0.906	0.935	0.946	0.942	0.948	0.161	0.116	0.100	0.104	0.096
0.02	0.871	0.925	0.942	0.939	0.945	0.193	0.132	0.108	0.109	0.100
0.03	0.844	0.910	0.935	0.933	0.942	0.215	0.172	0.120	0.115	0.107
0.04	0.817	0.880	0.925	0.925	0.937	0.236	0.231	0.143	0.127	0.118
0.05	0.793	0.845	0.904	0.909	0.927	0.257	0.291	0.186	0.150	0.142

Table 2: Synthetic dataset quantified against the mutated reference transcriptome with different mutation rates. The spearman correlation and MARD (mean absolute relative difference.) are calculated with respect to the ground truth.

In all our experiments, reads are mapped to the transcriptome using using *Bowtie2*, *kallisto*, *Hera*, selective-alignment and *STAR*. Subsequently, transcripts are quantified by *Salmon* (v0.9.1) using the relevant mappings (from alignment or the non-alignment-based methods) as input (except in the cases of *kallisto* (0.43) and *Hera* (1.2), which include implementations of their quantification algorithms). The alignment mode of *Salmon* enables us to use *STAR* (v2.5) and *Bowtie2* (v2.3) output as a direct input to the quantification module — thereby reducing variability due to differences in the underlying methodology used for quantification. To achieve the most sensitive alignment, *Bowtie2* is run with the alignment options suggested for use with *RSEM* [13]. For aligning reads to the transcriptome using *STAR*, we used the same options described in [29]. When processing alignments, *Salmon* was run with `--rangeFactorizationBins 4` [36] and `--useErrorModel`. With selective-alignment, *Salmon* was run using the `--softFilter` flag (discussed in 2.3.3), a range factorization value of 4 and an edit distance threshold of 7. *kallisto* was run with default parameters. Both the selective-alignment and *kallisto* indices were built with $k = 25$; *Hera* does not include k-mer size as a user-defined parameter.

3.1 Quantification of simulated reads against mutated transcriptomes

We explored the performance of different alignment-based and alignment-free methods by quantifying simulated short RNA-seq reads against mutated reference sequences. The simulation process consists of two steps. In the first step, we mapped an experimental RNA-seq sample (accession number SRR5638585) to the human transcriptome (Ensembl release 80 [34]) using *Salmon*. The resulting abundance vector, in conjunction with the full transcriptome sequence generated from the full human genome and the corresponding annotations (version GRCh37.p13), is used to simulate five batches of 100bp paired-end RNA-seq samples, where each batch contains ~ 47M reads. We used the sequence simulator Polyester [7] for generating the read datasets.

While the simulated dataset enables comparison with the ground truth, the quality of the reads is high and does not show the subtle nuances that arise when mapping reads from experimental sequencing datasets. In reality, the sequenced reads could differ from the annotated reference sequence due the presence of mutations (variants) in the sequenced organism. In other cases, a reference sequence from one species could be used to analyze data from a phylogenetically closely related species, for which an annotated reference in unavailable. Therefore, to recapitulate these adversarial situations, instead of mapping the simulated reads to the exact underlying transcriptome used for read generation, we map them against references mutated at a controllable rate.

The mutated version of the transcriptome is derived from the underlying reference genome that was subject to random mutations. The nucleotides of the reference genome were randomly altered based on a Poisson process with a tunable *rate* parameter. The rate parameter enables controling the rate of mutation that we want to introduce in the reference genome. For the current manuscript we have used 5 equally spaced rate parameters from 0.01 to 0.05. The mutated genome sequences and the original annotation are used to generate the mutated reference transcriptomes. As the resulting transcriptomes contain devations from the indexed reference, we believe that mapping to these references will capture some aspects of the difficulties encountered when applying such tools to certain experimental datasets.

To evaluate the performance we have measured the quantification accuracy of different tools with respect to the ground truth provided to Polyester. As explained earlier, tools such as *kallisto*, *Hera* and selective-alignment have a quantification pipeline attached to the mapping module and are, therefore, capable of generating abundance vectors directly. On the other hand, *Bowtie2* and *STAR* generate alignment files that we have coupled with *Salmon* (run in alignment-based mode) to obtain abundance estimates.

Performance of the various methods on a simulated sample is shown in Table 2. In this case, the simulated sample is mapped against 5 different mutated transcriptomes with increasing error rates and the corresponding spearman correlation and MARD values calculated using the ground truth. As shown in Table 2, the correlation between quantification estimates using selective-alignment and the ground truth is higher than the other self-contained quantification methods, *kallisto* and *Hera*. This gap between correlation values increases as the rate of mutation in the reference transcriptome is increased, showing the ability of selective-alignment to accurately map reads against diverging transcriptomes. The MARD values for selective-alignment are lower in comparison with other non-alignment-based methods as well. Several other metrics have been shown in section S1.4 to elaborate on the performance of selective-alignment in comparison to the other approaches on these datasets. Scatter plots of these results — provided in section S1.5 — show that these relative metrics are not skewed due to outliers in the quantification estimates.

To measure the variation in quantification about a single random instance of simulated data (i.e., data generated with a particular

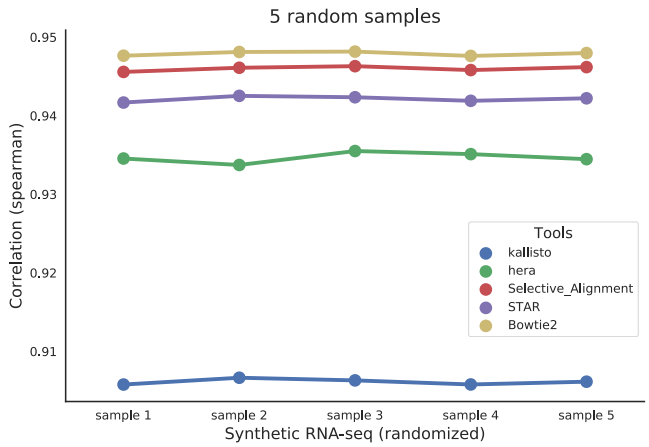

Figure 5: Performance variation of different tools on paired end read files produced with five random seeds.

random seed), we have also generated five different simulated RNA-seq datasets by passing different seeds to Polyester. To minimize external variation, we used the least mutated transcriptome (rate 0.01) as reference. By plotting the spearman correlations, as shown in Figure 5, we observe that, given that all the tools perform well on the random samples, the performance of selective-alignment is grouped with the alignment-based methods, such as *Bowtie2* and *STAR*. Further, the variation in quantification performance of all methods (i.e. the standared error) across these different simulated replicates is very small.

Note, that we also repeated this experiment by aligning the simulated reads against the original transcriptome used for the simulation. Under these circumstances, where the simulated reads and coverage profiles accord with the ideal assumed by the quantification models, and where there is no divergence between the reference being used for quantification and the sample being quantified, all the methods perform well, with small differences in quantification accuracies. The result from this analysis are presented in Section S1.3. To further investigate how sequencing error-rate (as opposed to transcriptomic variation) affects selective-alignment and other alignment-based or non-alignment-based methods, we have generated human transcriptome-wide reads with different sequencing error rates as an input to the Polyester simulator; the results and discussion of that experiment are in Section S1.2.

3.2 Experimental reads from human transcriptome

We have also benchmarked our proposed selective-alignment method on experimental data from SEQC(MAQC-III) consortium [25] samples (SRA accession `SRR1215996 - SRR1216000`). Each of the five technical replicates consists of ~11M, 100bp, paired-end reads, sequenced on an Illumina Hiseq 2000 platform.

We follow the same basic assessment methodology as discussed in Section 3.1, and report the mean Spearman correlation and MARD value for each method. However, we note that, since this is experimentally-derived data, there is no knowledge of ground truth transcript abundances. Instead, we have measured the overall concordance between different approaches. Given the results obtained in all of our other testing, we expect the *Bowtie2*-based pipeline to be the most accurate, so we are generally looking for high concordance with those quantifiaction estimates.

In Table 3, we compare the quantification results produced by different methods. Each individual cell contains the average obtained across the five samples. High Spearman correlation and low MARD value between *Bowtie2* and selective-alignment show that selective-alignment produces results most similar to those based on *Bowtie2*. Interestingly, the concordance between the selective-alignment and *Bowtie2*-based pipelines is even higher than the concordance between the two pipelines based on more traditional alignment approaches (i.e. *Bowtie2* and STAR). While we cannot assess the accuracy with respect to known ground truth on these samples, we nonetheless believe assessments based on real data like this are important to perform, as the complexity of experimental data seems to be considerably higher than that of simulated data and its characteristics can be markedly different. Finally, Table 4 provides timing and memory assessments of all the methods running on sample `SRR1215996`. Since the mapping phase of selective-alignment is not distinct from the quantification phase, the memory and time footprints include the mapping part of the pipeline. Further, disk space is not comparable to alignment-based methods, since alignment files are not written directly as output of selective-alignment (rather, the selective-alignment algorithm informs the mappings and provides edit-distance-based scores — as described in Eq. (1) — directly to the quantification algorithm).

4 CONCLUSION

Recently, fast non-alignment-based approaches have been developed for mapping RNA-seq reads to transcriptomes. Rather than generating full alignments, these approaches compute "mapping" information that is often sufficient for a number of given analysis tasks (e.g., transcript quantification [3, 9, 22, 23, 37] or metagenomic abundance estimation [24]). Yet, there exist scenarios where such non-alignment-based approaches can go awry; either failing, by the greedy nature of their procedures, to find the true target of origin of a read, or by allowing spurious mappings to targets supported by exact matches that would nonetheless fail reasonable alignment scoring filters. Moreover, it is sometimes desirable to be able to produce, on demand, the edit distance or alignment

Table 3: The Spearman correlation and MARDS between transcript abundances computed by all methods on experimental data. Each number is the mean on 5 different samples; the numbers in the lower left triangle of the matrix are the Spearman correlations and the ones in upper right are the MARD values. "selective " refers to selective-alignment.

Method	*kallisto*	*Hera*	selective	*STAR*	*Bowtie2*
kallisto	1 \ 0	0.19	0.16	0.15	0.17
Hera	0.87	1 \ 0	0.13	0.15	0.14
selective	0.90	0.90	1 \ 0	0.13	**0.06**
STAR	0.90	0.90	0.91	1 \ 0	0.13
Bowtie2	0.89	0.90	**0.97**	0.91	1 \ 0

Table 4: Comparison of timing and memory foot-print of selective-alignment with other alignment and non-alignment methods on experimental sample SRR1215996. The timing performance for *STAR* and *Bowtie2* is the sum of mapping and quantification (with salmon) steps (first number is the mapping step) and memory footprint is the max memory footprint of these two steps (first number is for the mapping step).

Method	time (s)	memory (KB)
kallisto	61	4006284
Hera	38	6736576
selective-alignment	65	7994324
STAR	398+96	max(8342444,5513432)
Bowtie2	977+125	max(1020032,9949380)

that would result from a given mapping location. The recently-introduced *Hera* validates mapping quality using alignment, which resolves spurious mappings, though it still suffers a loss of sensitivity compared to traditional alignment methods, and fails to process *denovo* assembled transcriptomes. In this paper, we introduce a selective alignment algorithm that attempts to bridge the gap between these non-alignment-based algorithms and more traditional alignment approaches. Selective-alignment improves upon both the sensitivity and specificity of these non-alignment-based algorithms while making very moderate concessions with respect to the computational budget. To achieve this level of efficiency, a number of algorithmic innovations were required, some of which may be of general interest. In the future, we hope to expand upon the notion of selective alignment even further, both by improving the algorithm and implementation, and by exploring use cases where selective alignment applies. Such situations are those where fast non-alignment-based approaches are inappropriate and traditional alignment approaches are too slow. In terms of improving the method, we hope to add functionality to automatically predict the optimal edit distance threshold in the read mappings based on the quality of the alignments, and for selective-alignment to self-tune to properly handle edge cases, such as soft clipping. The selective-alignment algorithm currently implements user specified edit distance threshold for filtering spurious reads. A more data-driven choice of filter can lead to a more resilient threshold that can perform gracefully while handling both adversarial reads as well as high-quality reads in heterogeneous read samples. In high quality samples, the edit distance bound can be set lower to further speed-up the algorithm. Future work will also include support for reporting the actual CIGAR strings for applications that require this information, such as RNA-seq based variant calling or allele identification.

ACKNOWLEDGMENTS

This work has been supported by the National Science Foundation (BBSRC-NSF/BIO-156491).

REFERENCES

[1] Can Alkan, Jeffrey M Kidd, Tomas Marques-Bonet, Gozde Aksay, Francesca Antonacci, Fereydoun Hormozdiari, Jacob O Kitzman, Carl Baker, Maika Malig, Onur Mutlu, S Cenk Sahinalp, Richard A Gibbs, and Evan E Eichler. 2009. Personalized copy number and segmental duplication maps using next-generation sequencing. *Nature Genetics* 41, 10 (aug 2009), 1061–1067. https://doi.org/10.1038/ng.437

[2] Michael J Axtell. 2014. Butter: High-precision genomic alignment of small RNA-seq data. *bioRxiv* (2014), 007427.

[3] Nicolas L Bray, Harold Pimentel, Páll Melsted, and Lior Pachter. 2016. Near-optimal probabilistic RNA-seq quantification. *Nature Biotechnology* 34, 5 (2016), 525–527.

[4] Cyan. 2018. xxHash — Extremely fast hash algorithm. http://cyan4973.github.io/xxHash/. Accessed: 2018-04-30.

[5] Nadia M Davidson, Ian J Majewski, and Alicia Oshlack. 2015. JAFFA: High sensitivity transcriptome-focused fusion gene detection. *Genome medicine* 7, 1 (2015), 43.

[6] Alexander Dobin, Carrie A Davis, Felix Schlesinger, Jorg Drenkow, Chris Zaleski, Sonali Jha, Philippe Batut, Mark Chaisson, and Thomas R Gingeras. 2013. STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* 29, 1 (2013), 15–21.

[7] Alyssa C Frazee, Andrew E Jaffe, Ben Langmead, and Jeffrey T Leek. 2015. Polyester: simulating RNA-seq datasets with differential transcript expression. *Bioinformatics* 31, 17 (2015), 2778–2784.

[8] Faraz Hach, Fereydoun Hormozdiari, Can Alkan, Farhad Hormozdiari, Inanc Birol, Evan E Eichler, and S Cenk Sahinalp. 2010. mrsFAST: a cache-oblivious algorithm for short-read mapping. *Nature Methods* 7, 8 (aug 2010), 576–577. https://doi.org/10.1038/nmeth0810-576

[9] Chelsea J-T Ju, Ruirui Li, Zhengliang Wu, Jyun-Yu Jiang, Zhao Yang, and Wei Wang. 2017. Fleximer: Accurate Quantification of RNA-Seq via Variable-Length k-mers. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*. ACM, 263–272.

[10] Daehwan Kim, Ben Langmead, and Steven L Salzberg. 2015. HISAT: a fast spliced aligner with low memory requirements. *Nature methods* 12, 4 (2015), 357–360.

[11] Ben Langmead and Steven L Salzberg. 2012. Fast gapped-read alignment with Bowtie 2. *Nature methods* 9, 4 (2012), 357–359.

[12] Ben Langmead, Cole Trapnell, Mihai Pop, Steven L Salzberg, et al. 2009. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology* 10, 3 (2009), R25.

[13] Bo Li and Colin N Dewey. 2011. RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. *BMC bioinformatics* 12, 1 (2011), 323.

[14] Heng Li. 2013. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM.

[15] Heng Li and Richard Durbin. 2009. Fast and accurate short read alignment with Burrows–Wheeler transform. *Bioinformatics* 25, 14 (2009), 1754–1760.

[16] Heng Li, Jue Ruan, and Richard Durbin. 2008. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome research* 18, 11 (2008), 1851–1858.

[17] Yang Liao, Gordon K Smyth, and Wei Shi. 2013. The Subread aligner: fast, accurate and scalable read mapping by seed-and-vote. *Nucleic acids research* 41, 10 (2013), e108–e108.

[18] Bo Liu, Hongzhe Guo, Michael Brudno, and Yadong Wang. 2016. deBGA: read alignment with de Bruijn graph-based seed and extension. *Bioinformatics* 32, 21 (jul 2016), 3224–3232. https://doi.org/10.1093/bioinformatics/btw371

[19] Udi Manber and Gene Myers. 1993. Suffix arrays: a new method for on-line string searches. *siam Journal on Computing* 22, 5 (1993), 935–948.

[20] Gene Myers. 1999. A fast bit-vector algorithm for approximate string matching based on dynamic programming. *Journal of the ACM (JACM)* 46, 3 (1999), 395–415.

[21] Daniel Nicorici, Mihaela Satalan, Henrik Edgren, Sara Kangaspeska, Astrid Murumagi, Olli Kallioniemi, Sami Virtanen, and Olavi Kilkku. 2014. FusionCatcher-a tool for finding somatic fusion genes in paired-end RNA-sequencing data. *bioRxiv* (2014), 011650.

[22] Rob Patro, Geet Duggal, Michael I Love, Rafael A Irizarry, and Carl Kingsford. 2017. Salmon provides fast and bias-aware quantification of transcript expression. *Nature Methods* (2017).

[23] Rob Patro, Stephen M Mount, and Carl Kingsford. 2014. Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms. *Nature biotechnology* 32, 5 (2014), 462–464.

[24] L. Schaeffer, H. Pimentel, N. Bray, P. Melsted, and L. Pachter. 2017. Pseudoalignment for metagenomic read assignment. *Bioinformatics* (Feb 2017). https://doi.org/10.1093/bioinformatics/btx106

[25] SEQC/MAQC-III Consortium and others. 2014. A comprehensive assessment of RNA-seq accuracy, reproducibility and information content by the Sequencing Quality Control Consortium. *Nature Biotechnology* 32, 9 (2014), 903–914.

[26] Shihao Shen, Juw Won Park, Zhi-xiang Lu, Lan Lin, Michael D Henry, Ying Nian Wu, Qing Zhou, and Yi Xing. 2014. rMATS: robust and flexible detection of differential alternative splicing from replicate RNA-Seq data. *Proceedings of the National Academy of Sciences* 111, 51 (2014), E5593–E5601.

[27] Enrico Siragusa, David Weese, and Knut Reinert. 2013. Fast and accurate read mapping with approximate seeds and multiple backtracking. *Nucleic acids research* 41, 7 (2013), e78–e78.

[28] Martin Šošić and Mile Šikić. 2017. Edlib: a C/C++ library for fast, exact sequence alignment using edit distance. *Bioinformatics* 33, 9 (2017), 1394.

[29] Avi Srivastava, Hirak Sarkar, Nitish Gupta, and Rob Patro. 2016. RapMap: a rapid, sensitive and accurate tool for mapping RNA-seq reads to transcriptomes. *Bioinformatics* 32, 12 (2016), i192–i200.

[30] Jorge Vaquero-Garcia, Alejandro Barrera, Matthew R Gazzara, Juan Gonzalez-Vallinas, Nicholas F Lahens, John B Hogenesch, Kristen W Lynch, and Yoseph Barash. 2016. A new view of transcriptome complexity and regulation through the lens of local splicing variations. *Elife* 5 (2016), e11752.

[31] David Weese, Manuel Holtgrewe, and Knut Reinert. 2012. RazerS 3: faster, fully sensitive read mapping. *Bioinformatics* 28, 20 (2012), 2592–2599.

[32] Thomas D Wu and Serban Nacu. 2010. Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics* 26, 7 (2010), 873–881.

[33] Hongyi Xin, Donghyuk Lee, Farhad Hormozdiari, Samihan Yedkar, Onur Mutlu, and Can Alkan. 2013. Accelerating read mapping with FastHASH. *BMC genomics* 14, 1 (2013), S13.

[34] Andrew Yates, Wasiu Akanni, M Ridwan Amode, Daniel Barrell, Konstantinos Billis, Denise Carvalho-Silva, Carla Cummins, Peter Clapham, Stephen Fitzgerald, Laurent Gil, et al. 2015. Ensembl 2016. *Nucleic acids research* (2015), gkv1157.

[35] Matei Zaharia, William J Bolosky, Kristal Curtis, Armando Fox, David Patterson, Scott Shenker, Ion Stoica, Richard M Karp, and Taylor Sittler. 2011. Faster and more accurate sequence alignment with SNAP. *arXiv preprint arXiv:1111.5572* (2011).

[36] Mohsen Zakeri, Avi Srivastava, Fatemeh Almodaresi, and Rob Patro. 2017. Improved data-driven likelihood factorizations for transcript abundance estimation. *Bioinformatics* 33, 14 (jul 2017), i142–i151. https://doi.org/10.1093/bioinformatics/btx262

[37] Zhaojun Zhang and Wei Wang. 2014. RNA-Skim: a rapid method for RNA-Seq quantification at transcript level. *Bioinformatics* 30, 12 (2014), i283–i292.