Error correction enables use of Oxford Nanopore technology for reference-free transcriptome analysis

Kristoffer Sahlin[4,5]*, Botond Sipos[6]*, Phillip L James[6], Daniel J Turner[6]†, Paul Medvedev[1,2,3]†

[1]Department of Computer Science and Engineering, The Pennsylvania State University, University Park, USA
[2]Department of Biochemistry and Molecular Biology, The Pennsylvania State University, University Park, USA
[3]Center for Computational Biology and Bioinformatics, The Pennsylvania State University, University Park, USA
[4]Department of Mathematics, Science for Life Laboratory, Stockholm University, 106 91 Stockholm, Sweden
[5]Part of the work was carried out while KS was at the Pennsylvania State University and the University of Helsinki
[6]Oxford Nanopore Technologies Ltd., Oxford, UK.
*These authors contributed equally
†Co-corresponding authors

Abstract

Oxford Nanopore (ONT) is a leading long-read technology which has been revolutionizing transcriptome analysis through its capacity to sequence the majority of transcripts from end-to-end. This has greatly increased our ability to study the diversity of transcription mechanisms such as transcription initiation, termination, and alternative splicing. However, ONT still suffers from high error rates which have thus far limited it scope to reference-based analyses. When a reference is not available or is not a viable option due to reference-bias, error correction is a crucial step towards the reconstruction of the sequenced transcripts and downstream sequence analysis of transcripts. In this paper, we present a novel computational method to error-correct ONT cDNA sequencing data, called isONcorrect. IsONcorrect is able to jointly use all isoforms from a gene during error correction, thereby allowing it to correct reads at low sequencing depths. We are able to obtain an accuracy of 98.7-99.5%, demonstrating the feasibility of applying cost-effective cDNA full transcript length sequencing for reference-free transcriptome analysis.

Introduction

The sequencing of the transcriptome using long reads has proven to be a powerful method for understanding the transcriptional landscape of a cell (Wyman et al., n.d.; Bayega et al. 2018; Byrne, Cole, et al. 2019). Long-read technologies allow sequencing most transcripts end-to-end, thus overcoming the complex transcriptome assembly step required with short reads (Gordon et al. 2015; Liu et al. 2017). In particular, the Oxford Nanopore (ONT) platform is a leading technology for long read transcriptome sequencing, due to its portability, low cost, and high throughput (Sessegolo et al. 2019; Jenjaroenpun et al. 2018). It has enabled the study of alternative splicing patterns (Byrne et al. 2017), allele-specific expression (Byrne et al. 2017) or typing (Cole et al. 2019), RNA modifications (Leger et al. 2019; Sessegolo et al. 2019; Wongsurawat et al., n.d.), the discovery of novel isoforms (Workman et al. 2019; Clark et al. 2019; Sessegolo et al. 2019), and species identification in metatranscriptomic samples (Semmouri et al. 2019).

However, the scope of ONT transcriptome studies to date has been limited because of its relatively high error rate — about 14% for both direct RNA and cDNA sequencing (Workman et al. 2019). The most common approach to overcome this limitation is to align the reads against a reference transcriptome (e.g. GENCODE for human) (Wyman et al., n.d.; Workman et al. 2019). This makes the technology of limited use when a high-quality reference is not available, ruling out many non-model organisms. In addition, even when a reference is available, it does not usually capture sequence differences between individuals, cells, or environments, causing misalignment of reads from missing or highly variable loci. This has been shown to be particularly problematic in complex gene families, where a reference does not capture the high

sequence diversity between individuals (Sahlin et al. 2018). There are several experimental approaches to reducing the error rate (Lebrigand et al. 2019; Cole et al. 2019; Volden et al. 2018), but these typically come at a cost of decreased throughput and experimental overhead.

Computational error correction, on the other hand, is a highly promising approach to reduce error rates without affecting throughput or the need to customize experimental protocols. There are tools designed to correct errors in genomic reads ((Koren et al., n.d.),(Tischler and Myers, n.d.), (Salmela et al. 2016), (Xiao et al. 2017), (Chin et al. 2013)). But, transcriptomic error correction is challenging and differs from the genomic case because of structural variability within reads from the same gene or gene-family locus and because of highly variable and region-specific coverage within reads due to, e.g., alternative splicing, variable transcription start and end sites, and variable transcript abundances. In fact, a recent study found that applying error correctors designed for genomic reads to ONT transcriptome data had undesirable downstream effects, such as altering the isoform landscape by omitting or adding exons through over-correction, or by splitting reads at low coverage sites (Lima et al. 2019). To achieve the potential of error correction on ONT transcriptomic data, custom algorithms have to be designed. Recent papers have tackled clustering (Sahlin and Medvedev 2019; Marchet et al. 2019) and orientation problems for this data (Ruiz-Reche et al. 2019) but there is currently no tool available for error correction of ONT transcriptomic reads.

In this paper, we present a method for error correction transcriptome cDNA ONT data that reduces the error rate to about 1%, thereby demonstrating the feasibility of applying cost-effective cDNA full transcript length sequencing for reference-free transcriptome analysis. We are able to achieve these error rates through a novel computational error correction method called isONcorrect, which leverages the sequence regions shared between reads originating from distinct isoforms. IsONcorrect is available for download at https://github.com/ksahlin/isONcorrect. We evaluate the method using Drosophila cDNA data generated using a modified stranded PCS109 protocol, PCS109 spike-in (SIRV) data, and *in silico* data. Our method opens the door for much broader application of ONT transcriptome sequencing.

Results

We used one biological, one synthetic, and one simulated dataset (Table 1) to investigate the effects of error correction on read quality, error type, splice site accuracy. We also measured the effect of read depth and parameters on the correction algorithm's accuracy and runtime and memory usage. We present the results in this section and refer the reader to the Experimental Methods and Data Analysis Methods subsections for the relevant respective details.

Error rate analysis

We sequenced the transcriptome of a Drosophila sample using ONT, with a total of 4,350,977 reads with median length 538nt (Table 1). From these, we identified 3,747,729 reads as being end-to-end, which we call full length, and error corrected them with isONcorrect. To measure the error rate before and after correction, we aligned the reads to the Drosophila reference genome (assembly BDGP6.22) using the spliced mode of minimap2 and counted the number of mismatches (defined as any insertion, deletion, or substitution in the alignment). We compute the error rate as the number of mismatches divided by alignment length. Errors in the reads are reflected by mismatches in the alignment; however, mismatches may also result from true biological variation in the sample and from alignment errors or artifacts. Nevertheless, we expect the mismatch numbers to be a reasonable proxy for the relative improvement in error rates. Results for before and after error correction with isONcorrect are shown in Fig. 1A. The mismatch rate decreased from a median of 7.0% to a median of 1.3% (Table 1).

Due to the confounding of sequencing error with biological variation, we also generated a simulated dataset. We extracted 10,367 distinct transcripts from the ENSEMBL annotation of human chromosome 6 and simulated full length reads at controlled abundances (in the range of 1 to 100) from transcripts (Table 1) (for details of the simulations, see Supplementary Note B). Since sequencing errors were annotated as part of the simulated sequencing process, we could measure the error rate directly. As with real Drosophila data, we found that isONcorrect significantly reduces errors, with the median error rate decreasing from 6.95 to 0.6 (Table 1).

Unfortunately, while eliminating the effect of biological variability on error rate measurement, simulated data does not capture the full scope of errors and biases present in the real data. We therefore also evaluated isONcorrect on SIRV E0 (Spike-in RNA Variant Control Mixes) data. Our SIRV dataset consists of 68 synthetic transcripts from 7 different loci sequenced with ONT R9 technology (see Experimental Methods for details). The transcripts from each locus differ in their splicing pattern but not in any other mutation. With the SIRV dataset, we have the properties of real sequencing errors and eliminate the confounding effect of biological variation on measuring error rate. The downside of SIRV is that it does not represent the mutational complexity of a real genome. With these caveats in mind, we measured the error rate by aligning the reads to the sequences of the 68 true transcripts using minimap2 and assuming that any alignment mismatch is due to an error (see Data Analysis in Methods for details). Results for before and after error correction on the full SIRV dataset with isONcorrect are shown in Fig. 1B. The median error rate was 6.9% before error correction and 0.5% after (Table 1), a significant reduction.

Error profiles

We also investigated the error profiles of the datasets before and after correction. The SIRV dataset enabled us to measure the profile of sequencing errors without the confounding effect of

biological variations. We note that the overall error rate prior to correction (about 7%, Table 1) was lower than previously published cDNA ONT datasets (about 14%, (Workman et al. 2019)), likely due to improvements in the base calling software. The median substitution, insertion, and deletion rate was 2.2%, 1.9%, and 2.6%, respectively (Fig. 1C). We observed a similar distribution for Drosophila (2.3%, 1.7%, and 2.8%), with the caveat that it also includes true biological variation (Fig. 1C). Error-correction substantially reduced the error rate in each category. The median substitution, insertion, and deletion rates of SIRV reads fell to 0.0%, 0.0%, and 0.3%, respectively, after correction (Fig. 1C). Most of the remaining errors were deletions, indicating that this is the hardest error type for isONcorrect to correct.

Effect of read depth

The amount of reads generated from a transcript (i.e. its read depth or simply depth) is typically an important factor in determining whether a tool can correct the errors in the read. To explore this in isONcorrect, we first used simulated data, for which we know the precise read depth per transcript. As expected, the post-correction error rate decreased as a function of depth (Fig. 2A). Compared to the median pre-correction rate of about 6.95%, the median post-correction rate ranged from about 3% for depth of one, 2% for depth of 2 to 3, and 0.5% for depths of 10 or more. Next, we looked at the SIRV data. Since the SIRV dataset has very high coverage, we used a subsampling strategy to investigate the error rate per sampled transcript depth (details in Data Analysis in methods section) . The error rate decreased consistently for read depth up to 10, but did not improve for larger read depths (Fig. 2B).

We note that isONcorrect remains very effective at low read depths, i.e. for read depth one, the error rate is already reduced from 7% down to 4% in SIRV and to 3% in simulated data. This is due to isONcorrect's ability to jointly use all isoforms from a gene during error correction, which combines information across all the transcripts with shared or similar exons. For example, the SIRV data has 7 gene loci with several splice variants each (between 6-18), meaning that each exon will have higher coverage than any individual transcript.

Splice site accuracy and transcript recovery

One of the potential benefits of error correction is obtaining nucleotide-level resolution of splice sites. Simultaneously, correction around borders of splice junctions is known to be challenging and may alter the splice site, particularly if it is present only at low abundances (Lima et al. 2019). Since the Drosophila reference genome has high quality gene annotations, we used alignments to classify each read according to how it matches the annotated splice sites, using the terminology of (Tardaguila et al. 2018) (see Data Analysis in Methods).

As expected, we observed more reads fully matching an annotated transcript (FSM) after correction (Fig. 3A). We did not see any novel combinations of splice sites (NIC) in the reads before or after correction. This is not surprising given the high quality annotation of the Drosophila genome. However, it did underscore a positive aspect of ONT sequencing, which is

that no artificial transcripts have been constructed in the experimental steps of generating the data, such as reverse transcriptase template switching.

We did observe slightly more reference transcripts that have at least one FSM read in the original reads compared to corrected reads (13,062 and 12,990, respectively, with 173 lost and 101 gained) and investigated the lost transcripts after correction as a function of how abundant they were in the original reads (Fig. 3B). Out of the 173 transcripts that were not captured by a FSM read after correction, 108 and 36 of them had only one and two FSM original reads, respectively, and all of them occurred in less than 12 original reads. Therefore, a consequence of our correction algorithm is that the lowest abundant transcripts may be mis-corrected. However, we also observed 101 transcripts had no FSM support before correction but did after error-correction. As the error correction is reference agnostic, this is likely due to reads from annotated transcripts that were misaligned around splice sites prior to correction, and highlights the benefit of reference-free error correction.

Overcorrection

One pitfall of using an alignment-based evaluation method is when the error correction algorithm modifies non-erroneous positions of a read in a way that the read more closely aligns to the reference genome. A typical example is when there are two highly similar transcripts A and B and a read that comes from transcript A but is corrected by the algorithm to transcript B. Such overcorrection is an undesirable artifact because it misrepresents the biological sample; however, when using an alignment-based evaluation method, overcorrection can go undetected because it can actually improve the inferred error rate. Nevertheless, we were able to measure the presence of overcorrection using our simulated dataset, where the true transcript is known. We classified a read as overcorrected if the read has an edit distance smaller to a transcript other than the true transcript. This is computed by first aligning reads with minimap2, and then comparing the edit distance of minimap2's primary alignments to the edit distance to the true transcript. The overcorrected reads made up less than 1.0% (374 out of 59,440) of the total reads. Note that a small fraction of the reads, particularly from highly similar transcripts, may be included in our definition of overcorrected because initial sequencing errors made them more similar to another transcript then the original one; these are really instances of not enough correction rather than overcorrection.

To investigate further, we measured how much closer the overcorrected reads were to the incorrect transcript. We computed the *overcorrection distance* for a read as the edit distance of the read to its true transcript minus the edit distance to its closest aligned transcript. We then plotted the overcorrection distance together with the abundance of the true transcript, for the overcorrected reads (Fig. 4). We found that this distance was small for the vast majority of the reads, i.e. 5 or less positions in >76% of the overcorrected reads. In addition, the overcorrection was mostly limited to reads at low abundances, with 55% of overcorrected reads coming from transcripts with an abundance of ≤ 5. This indicates that overcorrection was mostly limited to transcripts at very low abundances, as opposed to larger exon-level miscorrections.

Effect of heuristics and parameters

For large clusters, isONcorrect uses a heuristic approximate algorithm (see Methods). While this reduces the runtime, it has the potential to reduce the quality of the results. We therefore investigated the accuracy between the approximate and exact mode using controlled subsampled reads from the SIRV dataset (see Data Analysis for details). As expected, we observed a decrease in accuracy in approximate mode compared to exact mode across all different k and w, with the difference in accuracy decreasing as read depth increases (Fig. S1). However, the accuracy differences between the two modes were negligible compared to the improvements over the uncorrected reads.

We also investigated the effect of parameter choices for the k-mer size k, and window size w, and the maximum anchor distance x_{max}. We observed minor effects across different k and w (Fig. S1). However, isONcorrect performs well over all the tested values of k and w, with the difference being minor compared to the overall effect of correction and of the read depth Overall, we obtained slightly better results for $k = 9$ which we set as the default value to isONcorrect. As for the maximum anchor distance, we saw a minor improvement in longer spans (80-100) compared to 40 (Fig. S2), and this informed us to set default value of $x_{max} = 80$. We generally conclude, however, that parameter values within the tested ranges have only a minor effect on accuracy.

Runtime and memory

We measured runtime and memory of isONclust and isONcorrect (Table 2). We used a machine with an x86_64 system running Linux v3.2.0-4-amd64 and equipped with 32 2-threaded cores and 512 GB RAM. We allowed isONclust to use 50 threads and isONcorrect to use 62 threads. While isONclust is relatively fast, the correction with isONcorrect takes significant time (over 2 days). Given the time investment into the sequencing protocol, we consider this time expense tolerable. However, we hope to speed up isONcorrect in the future by allowing parallelization across nodes, making it possible to speed up correction by running it on a multi-node cluster. For the full SIRV dataset, the runtime was dominated by the largest cluster with contained roughly half of the reads (597,877). In such extreme cases, the reads could be partitioned into sub-clusters and parallelized, possibly with expense to accuracy.

As for memory usage, the current memory usage require a large memory cluster to run. We note that in our simulated data, some transcripts were very long (>20,000 nucleotides). This resulted in a relatively large memory consumption given the number of reads, compared to the SIRV and Drosophila data. It is possible to decrease memory usage in several ways, such as increasing w or decreasing x_{max}, at the potential cost of accuracy. However, the memory footprint can be greatly reduced by implementing isONcorrect in C++ or storing minimizers and paired anchors in more efficient data structures (Chikhi, Holub, and Medvedev 2019).

Methods

isONcorrect algorithm

Algorithm overview

The input to our algorithm is a cluster of reads originating from transcripts of a single gene family. Such clusters can be generated from a whole-transcriptome dataset by using our previously published tool isONclust (Sahlin and Medvedev 2019). Each cluster is then processed individually and in parallel with isONcorrect. The goal of isONcorrect is to correct all the sequencing errors. The challenge that makes this problem different from error-correction of genomic data is the highly uneven coverage within different regions of the read, and structural differences of similar reads, both arising from exons differences due to alternative splicing, as well as alternative start and stop sites.

Our idea is to partition each read into intervals and then error correct each interval separately where the intervals should not cross exon and intron boundaries. Our strategy for partitioning of the read into separate intervals is based on a related idea used in the context of genomic error correction (Morisse et al. 2019), but we adapt it to the transcriptomic context. As structural differences and variable coverage is at the heart of transcriptomic error correction, we solve the partitioning problem by formulating it as a global (with respect to the read) k-mer anchor optimization problem over anchor depth.

It is desirable that each interval is found in as many reads as possible for improving the power of error correction. For a given cluster, we obtain intervals spanned by paired minimizers (as described in previous section). Then, for each read, roughly speaking, we find the set of non-overlapping intervals that jointly covers as much of the read as possible, and are found in as many of the other reads a possible. We solve this problem by solving the *Weighted Interval Scheduling* (WIS) problem.

Intuitively, by optimizing for the most common minimizers pairs, the solution is likely to contain only intervals that has (a) good support in other reads to be corrected confidently, and (b) minimizers are both correct and can be trusted as anchors in the local regions. Additionally, by optimizing over total spanned region, our correction will correct as much of the read as confidently possible. An additional feature of locally correcting reads in intervals is that it gives a natural way of correcting exon regions, where multiple alignment methods would fail to align noisy reads over exon boundaries.

The steps of the algorithm are illustrated in Fig. 5. We now give the relevant definitions, then go in detail through the steps of the algorithm, and finally describe heuristic modifications to improve run-time.

Definitions

Let r be a string of nucleotides that we refer to as a read. Given two integers k and w such that $1 \leq k \leq w \leq |r|$, the minimizer of r at position p is the lexicographically smallest substring m of length k that starts at a position in the interval of $[p,\ p+w)$. We then say that r has a minimizer m, or, alternatively, has a *positional minimizer* (m,p). For example, for $r = AGACCAT,\ k=2,\ w=3$ we have that the ordered set $M = \{(m_i,p_i)\}$ of positional minimizers are $M = \{(AG,\ 0),\ (AC,\ 2),\ (CA,\ 4),\ (AT,\ 5)\}$. Let x_{min} and x_{max} be two positive integer parameters, where we call x_{max} the maximum paired anchor distance . Then we let $W_r = \{((m_i,p_i),\ (m_j,p_j)) \in M \times M \mid x_{min} \leq p_j - p_i \leq x_{max}\}$ be the ordered set (according to increasing p_i then p_j) of paired positional minimizers separated by at least x_{min} and at most x_{max} nucleotides in r. Similarly, we let $StrW_r = \{(m_i,m_j) \mid ((m_i,p_i),\ (m_j,p_j)) \in W_r\}$ be the sequence of paired minimizers, i.e. W_r with the positions omitted but duplicates retained. For example, the above set of minimizers with $x_{min} = 2,\ x_{max} = 3$ gives $W_r = \{((AG,\ 0),\ (AC,\ 2)),\ ((AC,\ 2),\ (CA,\ 4)),\ ((AC,\ 2),\ (AT,\ 5)),\ ((CA,\ 4),\ (AT,\ 5))\}$ and $strW_r = (AG,AC),\ (AC,CA),\ (AC,AT),\ (CA,AT)$. Given a set of reads R, we let W be the union of all W_r for the reads in R and we let $StrW$ be the union of all $StrW_r$.

The weighted interval scheduling problem takes as input a set of intervals $I = \{i_1, ...i_n\}$, where $i_j \in [a_j, b_j]$, $a_j, b_j \in R$ and $a_j < b_j$, and a weight w_j associated with each i_j. The output is a subset $I' \subseteq I$ of non-overlapping intervals whose sum of weights is maximized. The weighted interval scheduling problem can be solved exactly using a dynamic programming algorithm that runs in $O(n \log n)$ time, where n is the number of intervals (Kleinberg and Tardos 2013)

Algorithm details

For a given cluster, we first generate all the paired positional minimizers $strW$ of the reads. Then for each read r we will construct a weighted interval scheduling instance (Step 1). Each positional minimizer pair $((m_1,p),(m_2,q)) \in W_r$ defines an *interval* on r that is spanned by, but does not contain the minimizers, i.e $[p+k,q)$. The interval is given the weight $a(q_1 - p_1 - k)$, where a denotes the *support* of the interval. We compute the support as the number of occurrences of (m_1,m_2) in $strW_r$ whose intervals have a similar sequence to the one spanned by (m_1,m_2) in r, and this is computed as follows. Let $r' \neq r$ be a read containing (m_1,m_2) with coordinates (p',q') in r' and let $s = r[p : q+k]$ and $s' = r'[p' : q'+k]$ be substrings of r and r' spanned by, and including, the minimizer windows. We consider that s' has similar sequence to s if $\frac{ed(s,s')}{|s|} < Q$, where Q is the allowed relative distance decided from the quality values in

s and s'. Here, $ed(\cdot)$ refers to the edit distance between the two segments. We use edlib (Šošić and Šikić 2017) to calculate the edit distance. If a read has multiple (possibly disjoint) intervals matching a single interval of r, only the most similar one according to edit distance is considered.

Next, for the read r we send the instance of all intervals and their weights to a weighted interval scheduler (Step 2) and obtain an optimal solution by solving the weighted interval scheduling problem in $O(n \log n)$ time using classic dynamic programming algorithm. Intuitively, this gives us a set of disjoint intervals in r, with a preference of a combination of intervals that are highly supported and covering as much of the read as possible.

For each interval obtained in the WIS solution, we send for correction the substring of the read and all the supporting substrings. The correction is performed as follows. We create a consensus substring c of all the m substrings included in the instance by forming a partial order alignment graph (Lee, Grasso, and Sharlow 2002) using SPOA (Vaser et al. 2017), then choosing the consensus based on the heaviest bundle algorithm (Lee 2003) (Step 3). We then perform error correction of the substring with respect to the consensus (Step 4), as follows. First, we create a multi-alignment matrix A from pairwise alignment of all the substrings to the consensus (we use the method described in (Sahlin et al. 2018)). We then identify all sufficiently covered variations with respect to the consensus and include them as *alternative references*. These alternative references together with the spoa consensus forms the set of sequences that we will correct the read subsequence to.

Let c_j denote the nucleotide at position j in spoa consensus c and $c_{j:j'}$ denote the substring of nucleotides between position j and j'. For each position j in c we construct *alternative references* as follows. We identify the subset of columns $A_{\cdot,j-k:j+k}$ in A to where the *kmer context* $c_{j-k:j+k}$ is aligned. We denote $c_{j-k:j+k}$, $A_{\cdot,j-k:j+k}$, and any row $a_{i,j-k:j+k}$ (i.e., aligned read) in A as $c^k{}_j$, $a^k{}_j$, and $A^k{}_j$ for simplicity. Now, as alternative references at position i we store all rows in $A^k{}_j$ that occur more than $\frac{mT}{ed(H(c^k{}_j), H(a^k{}_j))}$ times as alternative references, together with the variant over position j that the alternative reference support. Here $H(\cdot)$ is the homopolymer compression function. We compress contexts to reduce alternative references based on homopolymer length differences solely, as these regions are more error prone to deletions. This means that, at each position j in A we have one primary variant and reference (based on the sequence of c), and zero or more alternative references and variants that passed the abundance threshold.

We now correct the substring a_i of a read i at position j (denoted a_{ij}) as follows. If position j has alternative references, we let a_{ij} equal the nucleotide to the primary or alternative reference with lowest edit distance to $a^k{}_j$. In case of only the primary reference, we set $a_{ij} = c_j$.

Heuristic modifications

We refer to the algorithm we have described up to this point as *exact*. We find that it works fast in practice for clusters of small and medium size clusters (i.e. for clusters with tens or hundreds of reads). However, for large clusters with thousands of reads this algorithm can be slow, and in this section we describe how we modify the exact algorithm to make it faster. We refer to the modified algorithm as "approximate." The time bottleneck is in steps 1, 3, and 4. Firstly, we repeatedly call edlib to calculate edit distance for all reads, regions and identical minimizer combinations. Secondly, we repeatedly do error correction by using spoa and creating the multi-alignment matrix. We take the following action to reduce the running time.

Recall that when error correcting a given read segment s, we identify all other read segments s' that support s and build an alignment matrix A. In the approximate version, we use the opportunity to also error correct all other segments s', using the same alignment matrix A. For each s', we store the corrected substring, the support of the instance, as well as the start and end position within the given read as information in a hash table, indexed by the read id. At the time of correcting a read, this hash table will be queried to identify the previously processed regions in this read. The processed regions may overlap. We do not compute the support for these processed regions (Step 1), and instead use the support stored in the hash table. If the processed region is then selected in the WIS solution, error correction is not done as per steps 3-4; instead, the corrected substring stored in the hash table is used directly. The approximate algorithm greatly reduces the runtime, as many segments are already computed and corrected in previous iterations.

We also make other heuristic modifications, in addition to the approximate algorithm. We introduce a parameter max_seq_to_spoa to limit the amount of sequences that goes into forming the consensus for very large clusters with spoa (default 200). This reduces runtime without noticeable effect in accuracy. We also mask positional minimizer pairs that contain only A's in both anchors. This is because many transcripts have a poly A tail, leading the minimizer database to be redundant and repetitive in these regions. Finally, we limit to process max_seq reads at a time within a cluster (default 1000).

As w will affect runtime and memory, we set appropriate w based on the number of reads in the batch to correct, where w is chosen as follows: $w = k + floor\{|C|/500\}$ where $|C|$ is the size of the cluster.

Experimental

D. melanogaster total RNA was isolated from adult W1118 flies according to the protocol outlined in Supplementary Note A and sequenced according to the PCS-109 protocol (https://community.nanoporetech.com/protocols/cdna-pcr-sequencing_sqk-pcs109/v/PCS_9085

_v109_revJ_14Aug2019). Primers were modified so that only the forward primer contained rapid attachment chemistry, resulting in single end adaption of the cDNA representing the 5' end of the RNA molecule (stranded sequencing). For amplification of the first strand cDNA, 12 cycles were used and 100 fM of library was loaded onto a FLO-MIN106 flowcell and sequenced for 48 hrs on the GridION system. Basecalling was performed in real time using guppy 3.4.8.

Synthetic spike-in transcripts made by Lexogen (SIRV E0): https://www.lexogen.com/store/sirvs SIRV E0 polyA RNA (Lexogen) (1ng) was used as a template for reverse transcription for use in the PCS-109 cDNA by PCR sequencing kit (Oxford Nanopore) Following the manufacturer's instructions (see link above). For amplification of the first strand cDNA, 12 cycles were used and 100 fM of library was loaded onto a FLO-MIN106 flowcell and sequenced for 48 hrs on the GridION system. Basecalling was performed in real time using guppy 3.4.8. Only a subset of pass reads with mean base quality larger than 7 were uploaded.

The SIRV and Drosophila data has been deposited into the ENA under project accession number PRJEB34849, to be released prior to publication.

Data Analysis

Computational processing of the read data

To identify full length reads among the reads sequenced with ONT we ran pychopper (https://github.com/nanoporetech/pychopper, commit 6dca13d) on Drosophila and SIRV datasets that identifies and removes forward and reverse primers, and splits eventual chimeric reads containing more than one transcript (barcodes in the middle). Only reads deemed to have both a forward and reverse primer are used for downstream analysis. Pychopper2 was run with default parameters and 50 cores.

To process the full length reads into gene-clusters, we ran isONclust with default ONT parameter settings using the flag '--ont' that sets (-w 20, -k 13). We ran isONcorrect with parameters (for all datasets) of "k= 9, --xmin 2k, --xmax 80", and w is chosen adaptively.

Inferring read error rates from alignments

For drosophila data, where it is unknown which transcripts are sequenced, and novel transcripts compared to annotated transcriptome may be present, we infer read error rates by doing a spliced alignment of reads to the Drosophila reference genome (assembly BDGP6.22) using minimap2 with parameters: -w4 -k 14 -ax splice --eqx. The -w 4 is supposed to be more sensitive but higher runtime than the recommended parameters for ONT transcript reads. We then infer insertions, deletions, substitutions from extended cigar strings of the primary alignments (with reads that are unaligned omitted from the analysis). However, we make the following modification not to count small introns as deletions. For a deletion in the cigar string of

the genomic alignment, we check whether the coordinates for the deletion matches a previously annotated intron from database Ensembl release 97 annotated on assembly BDGP6.22. If the deletion start and stop coordinates matches the intron annotation, we do not count it towards a deletion. We then say that for a read, the "% difference to the genome" is the total number of insertions, deletions, and substitutions divided by the alignment length, which is the total number of insertions, deletions, substitutions and matches.

For SIRV data, where we have the true transcripts present in the sequencing material, we infer read error rates by aligning reads to the transcriptome consisting of 68 synthetic transcripts using minimap2 with parameters -w4 -k 14 -a --eqx. We infer insertions, deletions, substitutions from the extended cigar strings of the alignments, but do not make the modification for intron deletions as we did for genomic alignments. The mismatch rate is computed as the sum of insertions, deletions, and substitutions divided by the alignment length.

SIRV subsampling experiments

The 68 SIRV transcripts contain five transcripts that are perfect substrings of other larger transcripts. These substring transcripts confound the alignments of the reads and the error rate calculations, so we filtered them out for this analysis, We aligned the 1,287,612 full-length SIRV reads to the remaining 63 of SIRV transcripts. We then ran 100 experiments, with 10 replicates in each. For each value of y between 1 and 100, we subsampled y aligned reads from each transcript. This resulted in a dataset of $63 \cdot y$ reads with an expected read depth of y. For each y, we did 10 replicates, to alleviate sampling variation. This gave a total of 1000 experiments. For each experiment, we clustered the reads with isONclust (git commit 8ba49e) with default parameters for ONT data. Then, we ran isONcorrect on the clusters, using the default parameters k=9, xmin=2*k, xmax=80. We also set the parameter --exact_instance_limit 50, that computes exact mode for clusters smaller than 50 reads.

Splice sites

To classify Drosophila reads, we use minimap2 to align reads to the Drosophila reference genome. We classify as a splice site everything that minimap2 flags as an intron location or any deletions (relative to the reference) whose start and stop sites match a true intron annotation in the ENSEMBL annotations. The second condition is necessary not to count small introns that are preserved in the reads but flagged as deletions in the alignment due to their small size (we observed introns as small as only two bases). We then match the splice sites of the alignments to existing Drosophila annotations and classify the transcripts according to the four categories defined by (Tardaguila et al. 2018) as follows. A transcript is a full splice match (FSM) if all its start and stop splice sites are in the database annotation and the particular combination of start and stop splice sites matches that of a known transcript; incomplete splice match (ISM) if all its start and stop splice sites are in the database annotation and they match match a consecutive subset of start and stop splice sites of an annotated transcript; novel-in-catalogue (NIC) if all the individual start and stop splice sites are in the database annotation but they create a new

combination of start and stop splice sites, or; novel-not-in-catalogue (NNC) if the transcript has at least one splice site that is not in the database.

Effect of parameters and heuristics experiments

First we aligned all SIRV reads to the 68 distinct transcripts (we observed the coverage shown in plot Figure S2). We then subsampled, without replacement, 3, 5, 10, and 20 reads that had unambiguous primary alignments from 4 randomly selected transcripts, with the requirement that the transcript had more unambiguous primary alignments than the required subsample size. We run isONcorrect on these datasets and measure the error rate of the corrected reads using both exact and approximate correction. We repeat the above experiment 10 times to alleviate variation from picking specific transcripts and reads.

Discussion

We presented a novel computational tool isONcorrect to error correct cDNA reads from Oxford Nanopore Technologies. On a Drosophila dataset, the raw data had an initial mismatch rate of 7.0%, which isONcorrect further decreased to 1.3%. This is a drastic improvement over previously published ONT transcriptome mismatch rates of about 14% (Workman et al. 2019). Compared to the R2C2 (Rolling Circle Amplification to Concatemeric Consensus) method, which modifies the experimental protocol, our approach does not decrease the throughput and achieves a significantly better mismatch rate (2.5% for R2C2) (Byrne, Supple, et al. 2019; Volden et al. 2018; Byrne, Cole, et al. 2019).

Evaluating the error rate of a transcriptome read error-correction tool is a challenge due to, on the one hand, the presence of biological variation and alignment ambiguity in real data, and, on the other hand, the limitations of simulated and synthetic data. In this paper, we took the kitchen sink approach and evaluated isONcorrect's performance on all these datasets. Our results showed consistent performance (Table 1), with the resulting mismatch rates between 0.5 - 1.3%

One of the underlying strengths of the isONcorrect algorithm is its ability to error correct reads even if there are as little as one read per transcript. The idea is to leverage exons that are shared between different splice isoforms. To achieve this, we pre-process the reads using our isONclust clustering algorithm, which clusters reads according to the gene family of origin. This strategy is in sharp contrast to approaches which cluster based on the isoform of origin. Such clustering results in low read coverage per transcript (Sahlin and Medvedev 2019), particularly for genes expressing multiple isoforms with variable start and stop sites and makes error correction unable to utilize full coverage over shared exons. By using isONclust to cluster at the gene family level, each read retains more complete exon coverage and helps the correction process preserve allele- or copy-specific small variant differences between transcripts that otherwise share the same structure. This effect is shown in our experiments, where there is already a significant reduction in the error rate (down to 3-4%) for transcripts with just one read.

IsONcorrect relies on two additional key algorithmic components to achieve scalability and high accuracy. First, we are able to partition the reads within a cluster into exon-like segments in a way that maximizes the read depth of each segment by formulating the problem as an instance of the classical weighted interval scheduling problem. This scheduling problem can then be solved optimally using an efficient and exact dynamic programming algorithm (Kleinberg and Tardos 2013). IsONcorrect is then able to separately correct the regions produced from the scheduling solution, where each region can have highly variable coverage but the coverage within a region is roughly equal. Second, we identify heuristic optimizations that drastically speed up our algorithm and adaptively apply them when the expected run-time is expected to be slow. We show empirically that these heuristics do not significantly reduce the accuracy.

There exist other algorithms for reference-free error correction of long transcriptomic reads that are specific to the Pacific Biosciences Iso-Seq platform. These include ToFU/isoseq3 (Gordon et al. 2015) and IsoCon (Sahlin et al. 2018), which perform both clustering and error correction and the final result is predicted unique transcripts. Isoseq3 is inherently limited to Iso-Seq data, while IsoCon, which is intended for targeted sequencing data, assumes high exon coverage and is not designed to handle variable start/end sites, which are ubiquitous in non-targeted datasets. Other approaches use short read data for error correction of long IsoSeq reads (Fu et al. 2018; Hackl et al. 2014).

There also exist several methods for error correction of ONT genomic data, both long-read-only and hybrid (short+long reads). We do not compare against these because a recent comprehensive benchmark showed that applying these to transcriptome data is problematic (Lima et al. 2019). While these tools reduced the error rate from about 13% down to 4%, all the tools also reduced the number of detected genes, gene family sizes, and the number of isoforms; they also reduced the number of detected splice sites and split reads up in low coverage regions. Similar findings were also observed in (Kuo et al. 2019) for genomic error correctors applied to PacBio's IsoSeq transcriptome reads. Given that genomic error correction tools alter the structural landscape of these reads, we do not consider them useful for most transcriptome applications.

The protocol used in this paper is based on the sequencing of cDNA, but there also exists a ONT protocol to sequence RNA directly (Jenjaroenpun et al. 2018; Smith et al. 2019; Depledge et al. 2019; Garalde et al. 2018; Workman et al. 2019). Direct RNA sequencing with ONT is a promising alternative to cDNA sequencing, but its potential has not yet been realized because of higher error rates (14%), low throughput, and the inability to guarantee reads spanning the full transcript (Workman et al. 2019). Because of high error rates, some of the analysis in (Workman et al. 2019), e.g. splice site analysis or allele-specific expression, was done using a combination of the GENCODE reference and the sequencing of cDNA from the same sample. On the other hand, cDNA sequencing produces high throughput and can, through experimental and computational methods, produce reads that are guaranteed to span the full molecule. With the method in this paper, the cDNA approach can now achieve error rates of about 1%, making it

applicable to reference-free analysis. However, applying isONcorrect to direct RNA reads is a direction for future work that should enable the reference-free use of direct RNA reads.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grants No. 1453527 and 1439057. Research reported in this publication was supported by the National Institute Of General Medical Sciences of the National Institutes of Health under Award Number R01GM130691 and by the National Human Genome Research Institute under Award Number U24HG010263. Data generation was funded by Oxford Nanopore Technologies (ONT). BS, DT, and PJ are ONT employees and stock option holders.

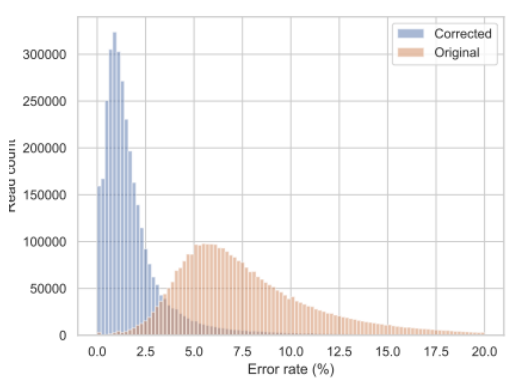
Tables and Figures

Dataset	Sequencing chemistry	# unique transcripts	# reads	Median read length	# full length reads	Uncorrected full length reads		Corrected full length reads	
						#aligned	Median diff to ref(%)	#aligned	Median diff to ref(%)
chr6	simulated	10,367	59,440	1017	59,440	59,440	7.0	59,440	0.6
SIRV	ONT R9	68	1,680,000	553	1,529,921	1,486,836	6.9	1,501,570	0.5
Drosophila	ONT R10	NA	4,350,977	538	3,747,729	3,327,355	7.0	3,368,963	1.3

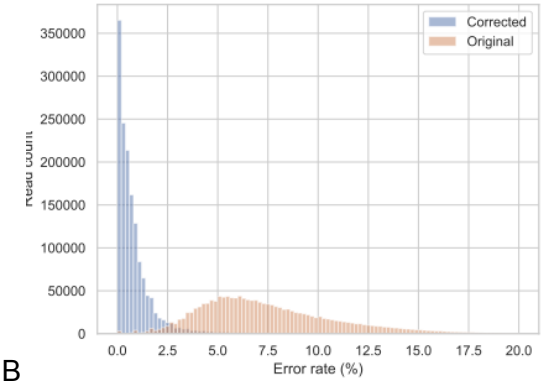
Table 1. Datasets used in evaluation of the transcriptomic Oxford Nanopore Sequencing datasets, before and after error correction.

Dataset	isONclust		isONcorrect	
	Peak memory	Runtime	Peak memory	runtime
Drosophila	34 Gb	2h 05m	256 Gb	56h 20m
chr6	7 Gb	0h 09m	261 Gb	3h 13m
SIRV	5 Gb	0h 10m	36 Gb	52h 58m

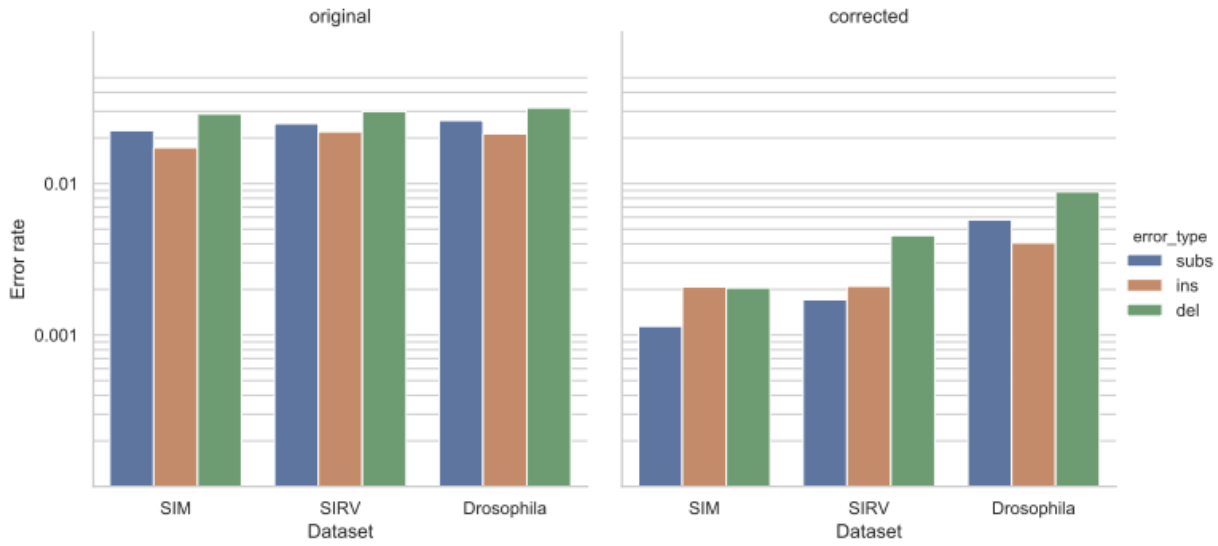
Table 2. Runtime and memory usage of the error correction pipeline.



A



B



C

Figure 1: Error rates of ONT reads before and after error correction. (A) Alignment difference distribution of corrected and original Drosophila reads. Differences can arise both from sequencing errors and variation to the reference genome. (B) Error rate distribution of corrected and original SIRV reads, for the whole SIRV dataset. (C) Error profiles of the datasets before and after correction, shown on a log scale. For Drosophila, the difference to genome is treated as an error rate in this panel.

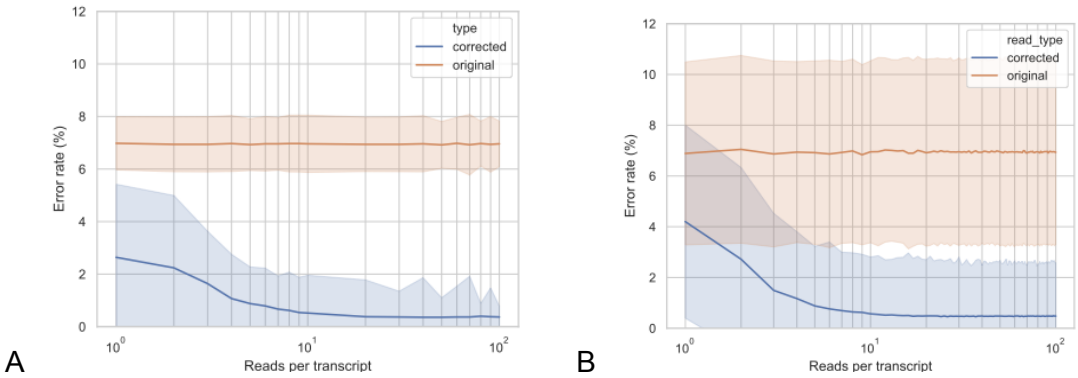


Figure 2: Effect of read depth on error rate. Panel (A) shows the median error rate of simulated read experiment based on true read depth of the transcript (i.e. number of reads sequenced from it). Panel (B) shows the median error rate of the SIRV data as a function of read depth, obtained via subsampling (see Data Analysis in Methods). The shaded areas show the standard deviation of the error rates.

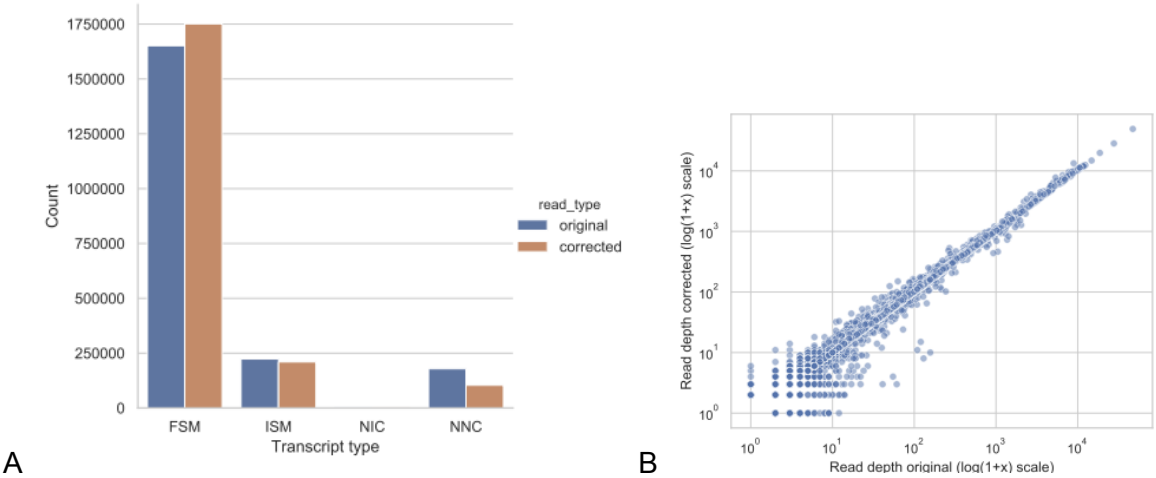


Figure 3. Splice site accuracy before and after error correction in the Drosophila data. (A) Total number of reads classified per splice site category, using the terminology of (Tardaguila et al. 2018). FSM stands for full splice match, ISM stands for incomplete splice match, NIC stands for novel-in-catalogue, and NNC stands for novel-not-in-catalogue. (B) For each transcript in the reference, we measure the number of reads aligning to it as a FSM, before and after error correction. Each dot represents a distinct transcript with at least one FSM in either the original or corrected reads.

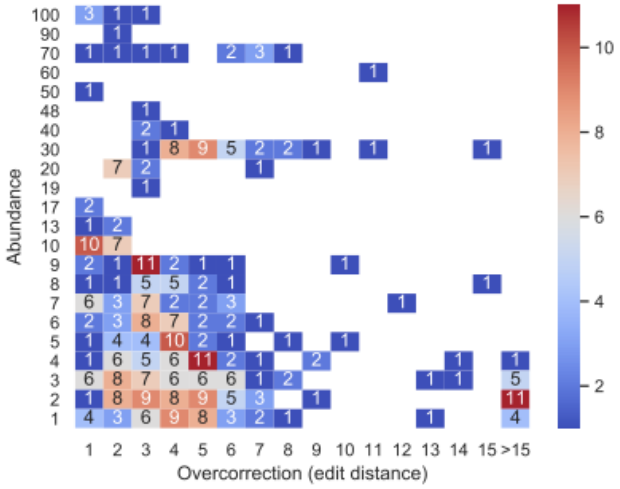


Figure 4. The effect of overcorrection in the simulated data. We bin each overcorrected read according to the abundance of its true transcript (y-axis) and its overcorrection distance (x-axis). Each cell shows the number of reads in the bin.

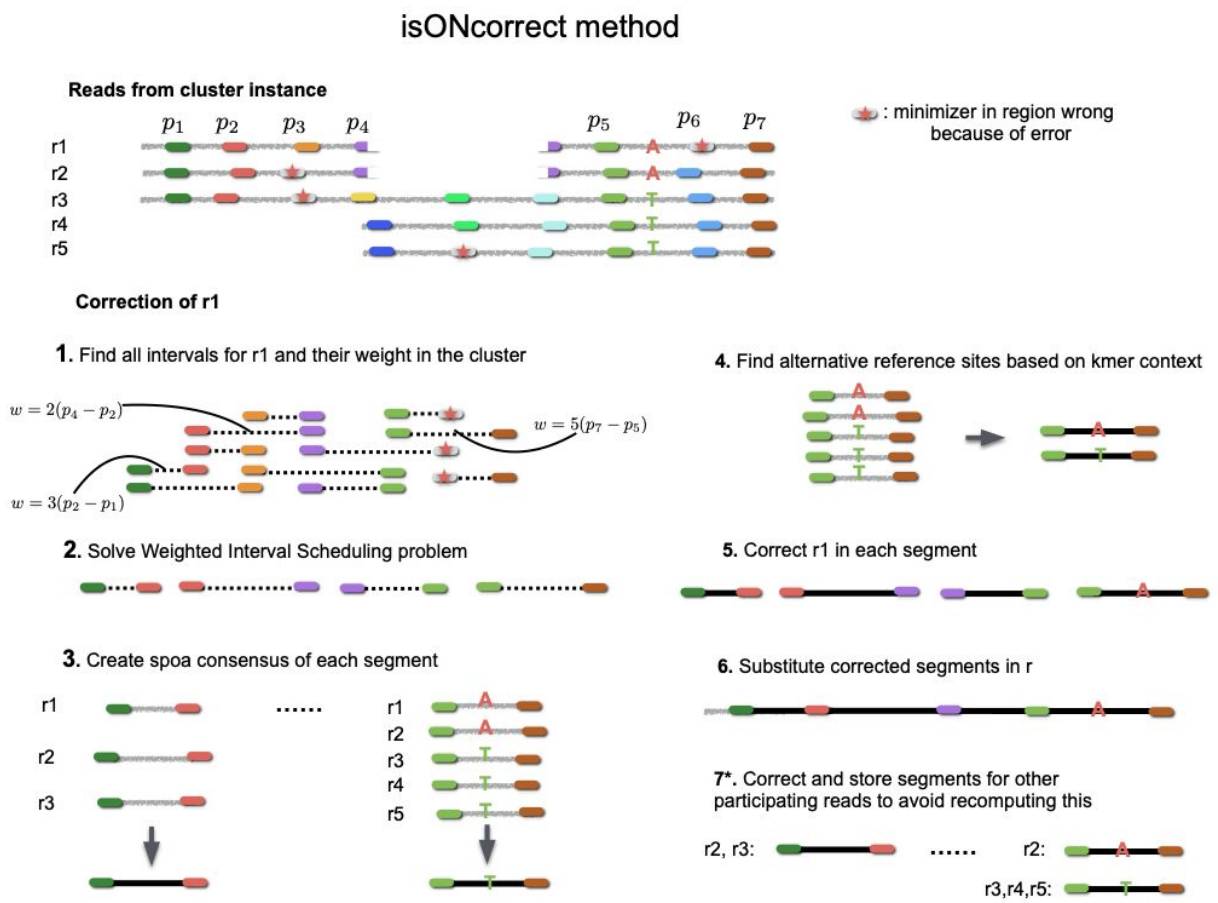


Figure 5. Overview of isONcorrect. The input to isONcorrect are reads from a single cluster produced by isONclust (or any other software that group reads into gene families of origin). This figure illustrates a cluster with five reads (r1 - r5) from three isoforms. isONcorrect finds all intervals with distance between x_{min} to x_{max} using paired minimizer anchors (shown as colored blocks) and adds them to a hash table. To correct a single read (e.g. r1), all the paired minimizer anchors found in r1 are queried in the hash table, and all reads containing this anchor pair are retrieved. In this example, r1 has 11 such paired anchors (shown in step 1). Each anchor pair is assigned a weight that is the product of its span and the number of reads containing this anchor pair (with the exception of filtering out anchor pairs of dissimilar regions; details in methods; step 1). For example, the paired anchor (p1,p2) occurs in three reads (r1, r2, and r3). The instance is sent to a weighted interval scheduler that finds the set of non-overlapping paired anchors with the biggest weight (step 2). In this case, four paired anchors are selected. All segments between the chosen anchor pairs are sent for correction. A reference is created (step 3) using spoa, and eventual alternative references are created as well (step 4). Each read segment in r1 is corrected to the closest (alternative) reference (step 5). The segments are inserted back into the original read r1 in what becomes the corrected read of r1 (step 6). An optional step 7 corrects the segments of the other reads in the same manner and stores them in a hash table to be retrieved whenever it is their turn to be corrected. For example, when it is r2's and r3's turn to be corrected, the interval spanned by the paired anchor (p1,p2) may be again encountered in the optimal WIS solution, allowing steps 3-5 to be skipped at that point.

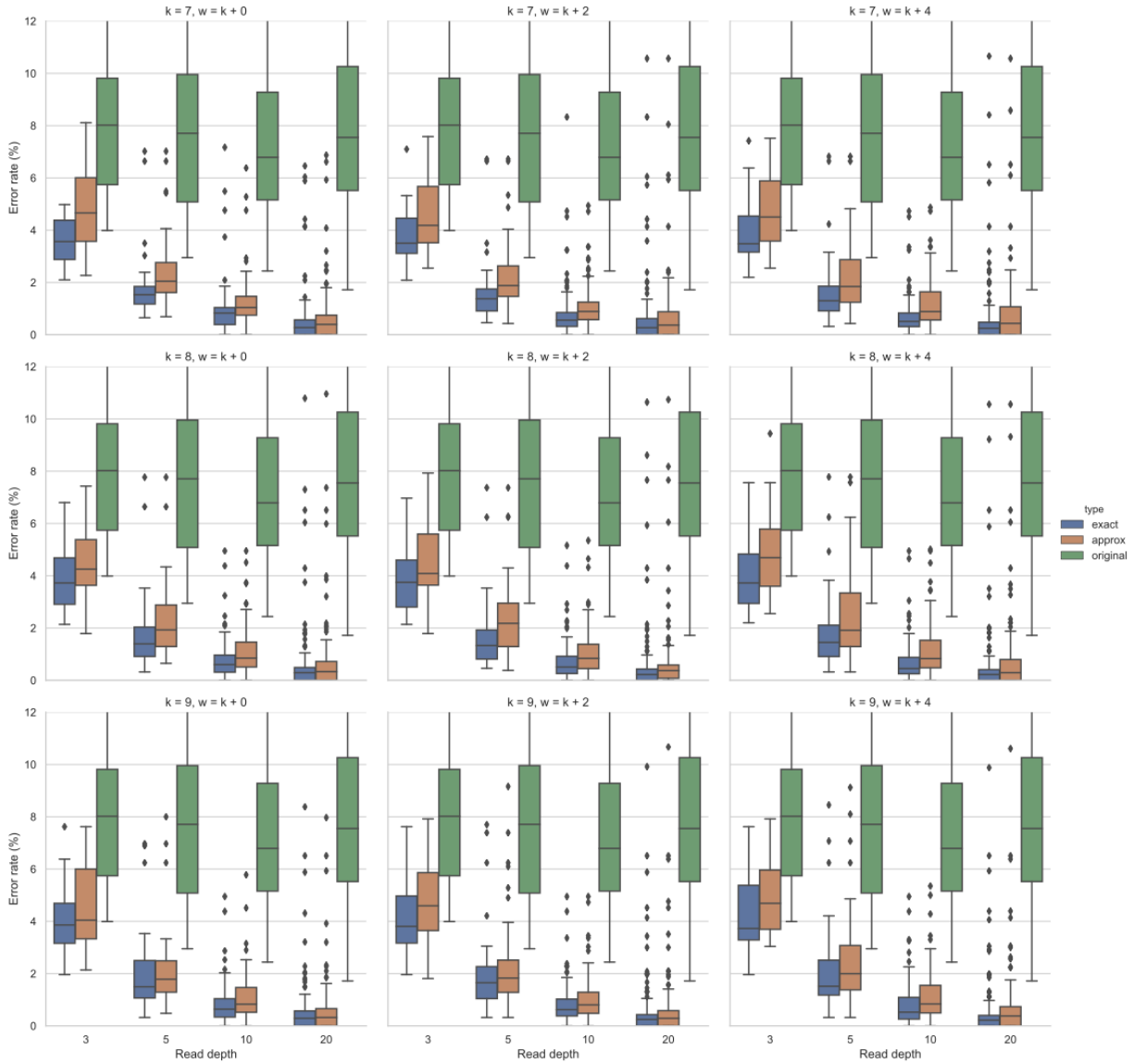


Figure S1. The effect on the error rate of parameters k and w, read depth, and the heuristic approximation algorithm. Each panel is labeled with a fixed value of k and w.

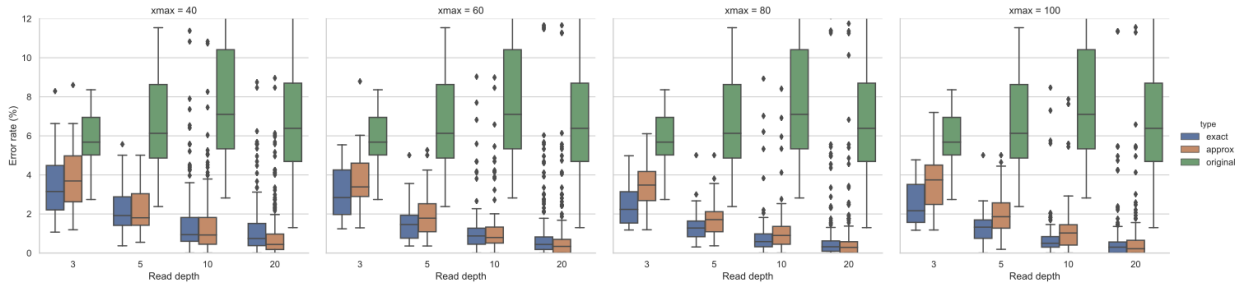


Figure S2. The effect on the error rate of the maximum anchor distance x_{max} , read depth, and the heuristic approximation algorithm. Each panel is labeled with a fixed value of x_{max} . The value of k and w is fixed to 9 in these experiments.

References

Bayega, Anthony, Yu Chang Wang, Spyros Oikonomopoulos, Haig Djambazian, Somayyeh Fahiminiya, and Jiannis Ragoussis. 2018. “Transcript Profiling Using Long-Read Sequencing Technologies.” *Methods in Molecular Biology* 1783: 121–47.

Byrne, Ashley, Anna E. Beaudin, Hugh E. Olsen, Miten Jain, Charles Cole, Theron Palmer, Rebecca M. DuBois, E. Camilla Forsberg, Mark Akeson, and Christopher Vollmers. 2017. “Nanopore Long-Read RNAseq Reveals Widespread Transcriptional Variation among the Surface Receptors of Individual B Cells.” *Nature Communications* 8 (July): 16027.

Byrne, Ashley, Charles Cole, Roger Volden, and Christopher Vollmers. 2019. “Realizing the Potential of Full-Length Transcriptome Sequencing.” *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences* 374 (1786): 20190097.

Byrne, Ashley, Megan A. Supple, Roger Volden, Kristin L. Laidre, Beth Shapiro, and Christopher Vollmers. 2019. “Depletion of Hemoglobin Transcripts and Long Read Sequencing Improves the Transcriptome Annotation of the Polar Bear (Ursus Maritimus).” https://doi.org/10.1101/527978.

Chikhi, Rayan, Jan Holub, and Paul Medvedev. 2019. “Data Structures to Represent Sets of K-Long DNA Sequences.” *arXiv*.

Chin, Chen-Shan, David H. Alexander, Patrick Marks, Aaron A. Klammer, James Drake, Cheryl Heiner, Alicia Clum, et al. 2013. “Nonhybrid, Finished Microbial Genome Assemblies from Long-Read SMRT Sequencing Data.” *Nature Methods*. https://doi.org/10.1038/nmeth.2474.

Clark, Michael B., Tomasz Wrzesinski, Aintzane B. Garcia, Nicola A. L. Hall, Joel E. Kleinman, Thomas Hyde, Daniel R. Weinberger, Paul J. Harrison, Wilfried Haerty, and Elizabeth M. Tunbridge. 2019. “Long-Read Sequencing Reveals the Complex Splicing Profile of the Psychiatric Risk Gene CACNA1C in Human Brain.” *Molecular Psychiatry*, November. https://doi.org/10.1038/s41380-019-0583-1.

Cole, Charles, Ashley Byrne, Matthew Adams, Roger Volden, and Christopher Vollmers. 2019. “Complete Characterization of the Human Immune Cell Transcriptome Using Accurate Full-Length cDNA Sequencing.” https://doi.org/10.1101/761437.

Depledge, Daniel P., Kalanghad Puthankalam Srinivas, Tomohiko Sadaoka, Devin Bready, Yasuko Mori, Dimitris G. Placantonakis, Ian Mohr, and Angus C. Wilson. 2019. “Direct RNA Sequencing on Nanopore Arrays Redefines the Transcriptional Complexity of a Viral Pathogen.” *Nature Communications* 10 (1): 754.

Fu, Shuhua, Yingke Ma, Hui Yao, Zhichao Xu, Shilin Chen, Jingyuan Song, and Kin Fai Au. 2018. “IDP-Denovo: De Novo Transcriptome Assembly and Isoform Annotation by Hybrid Sequencing.” *Bioinformatics* 34 (13): 2168–76.

Garalde, Daniel R., Elizabeth A. Snell, Daniel Jachimowicz, Botond Sipos, Joseph H. Lloyd, Mark Bruce, Nadia Pantic, et al. 2018. “Highly Parallel Direct RNA Sequencing on an Array of Nanopores.” *Nature Methods* 15 (3): 201–6.

Gordon, Sean P., Elizabeth Tseng, Asaf Salamov, Jiwei Zhang, Xiandong Meng, Zhiying Zhao, Dongwan Kang, et al. 2015. “Widespread Polycistronic Transcripts in Fungi Revealed by Single-Molecule mRNA Sequencing.” *PloS One* 10 (7): e0132628.

Hackl, Thomas, Rainer Hedrich, Jörg Schultz, and Frank Förster. 2014. “Proovread: Large-Scale High-Accuracy PacBio Correction through Iterative Short Read Consensus.”

Bioinformatics 30 (21): 3004–11.
Jenjaroenpun, Piroon, Thidathip Wongsurawat, Rui Pereira, Preecha Patumcharoenpol, David W. Ussery, Jens Nielsen, and Intawat Nookaew. 2018. "Complete Genomic and Transcriptional Landscape Analysis Using Third-Generation Sequencing: A Case Study of Saccharomyces Cerevisiae CEN.PK113-7D." *Nucleic Acids Research* 46 (7): e38.
Kleinberg, Jon, and Eva Tardos. 2013. *Algorithm Design: Pearson New International Edition*. Pearson Higher Ed.
Koren, Sergey, Brian P. Walenz, Konstantin Berlin, Jason R. Miller, Nicholas H. Bergman, and Adam M. Phillippy. n.d. "Canu: Scalable and Accurate Long-Read Assembly via Adaptive K-Mer Weighting and Repeat Separation." https://doi.org/10.1101/071282.
Kuo, Richard Izen, Yuanyuan Cheng, Jacqueline Smith, Alan L. Archibald, and Dave W. Burt. 2019. "Illuminating the Dark Side of the Human Transcriptome with TAMA Iso-Seq Analysis." https://doi.org/10.1101/780015.
Lebrigand, Kevin, Virginie Magnone, Pascal Barbry, and Rainer Waldmann. 2019. "High Throughput, Error Corrected Nanopore Single Cell Transcriptome Sequencing." https://doi.org/10.1101/831495.
Lee, Christopher. 2003. "Generating Consensus Sequences from Partial Order Multiple Sequence Alignment Graphs." *Bioinformatics* 19 (8): 999–1008.
Lee, Christopher, Catherine Grasso, and Mark F. Sharlow. 2002. "Multiple Sequence Alignment Using Partial Order Graphs." *Bioinformatics* 18 (3): 452–64.
Leger, Adrien, Paulo P. Amaral, Luca Pandolfini, Charlotte Capitanchik, Federica Capraro, Isaia Barbieri, Valentina Migliori, et al. 2019. "RNA Modifications Detection by Comparative Nanopore Direct RNA Sequencing." https://doi.org/10.1101/843136.
Li, Heng. 2018. "Minimap2: Pairwise Alignment for Nucleotide Sequences." *Bioinformatics* 34 (18): 3094–3100.
Lima, Leandro, Camille Marchet, Ségolène Caboche, Corinne Da Silva, Benjamin Istace, Jean-Marc Aury, Hélène Touzet, and Rayan Chikhi. 2019. "Comparative Assessment of Long-Read Error Correction Software Applied to Nanopore RNA-Sequencing Data." *Briefings in Bioinformatics*, June. https://doi.org/10.1093/bib/bbz058.
Liu, Xiaoxian, Wenbin Mei, Pamela S. Soltis, Douglas E. Soltis, and W. Brad Barbazuk. 2017. "Detecting Alternatively Spliced Transcript Isoforms from Single-Molecule Long-Read Sequences without a Reference Genome." *Molecular Ecology Resources* 17 (6): 1243–56.
Li, Yu, Renmin Han, Chongwei Bi, Mo Li, Sheng Wang, and Xin Gao. 2018. "DeepSimulator: A Deep Simulator for Nanopore Sequencing." *Bioinformatics* 34 (17): 2899–2908.
Marchet, Camille, Lolita Lecompte, Corinne Da Silva, Corinne Cruaud, Jean-Marc Aury, Jacques Nicolas, and Pierre Peterlongo. 2019. "De Novo Clustering of Long Reads by Gene from Transcriptomics Data." *Nucleic Acids Research* 47 (1): e2.
Morisse, Pierre, Camille Marchet, Antoine Limasset, Thierry Lecroq, and Arnaud Lefebvre. 2019. "CONSENT: Scalable Self-Correction of Long Reads with Multiple Sequence Alignment." https://doi.org/10.1101/546630.
Ruiz-Reche, Angel, Akanksha Srivastava, Joel A. Indi, Ivan de la Rubia, and Eduardo Eyras. 2019. "ReorientExpress: Reference-Free Orientation of Nanopore cDNA Reads with Deep Learning." *Genome Biology* 20 (1): 260.
Sahlin, Kristoffer, and Paul Medvedev. 2019. "De Novo Clustering of Long-Read Transcriptome Data Using a Greedy, Quality-Value Based Algorithm." In *Research in Computational Molecular Biology*, 227–42. Springer, Cham.
Sahlin, Kristoffer, Marta Tomaszkiewicz, Kateryna D. Makova, and Paul Medvedev. 2018. "Deciphering Highly Similar Multigene Family Transcripts from Iso-Seq Data with IsoCon."

Nature Communications 9 (1): 4601.
Salmela, Leena, Riku Walve, Eric Rivals, and Esko Ukkonen. 2016. “Accurate Self-Correction of Errors in Long Reads Using de Bruijn Graphs.” *Bioinformatics*. https://doi.org/10.1093/bioinformatics/btw321.
Semmouri, Ilias, Karel A. C. De Schamphelaere, Jan Mees, Colin R. Janssen, and Jana Asselman. 2019. “Evaluating the Potential of Direct RNA Nanopore Sequencing: Metatranscriptomics Highlights Possible Seasonal Differences in a Marine Pelagic Crustacean Zooplankton Community.” *Marine Environmental Research*. https://doi.org/10.1016/j.marenvres.2019.104836.
Sessegolo, Camille, Corinne Cruaud, Corinne Da Silva, Audric Cologne, Marion Dubarry, Thomas Derrien, Vincent Lacroix, and Jean-Marc Aury. 2019. “Transcriptome Profiling of Mouse Samples Using Nanopore Sequencing of cDNA and RNA Molecules.” *Scientific Reports* 9 (1): 14908.
Smith, Andrew M., Miten Jain, Logan Mulroney, Daniel R. Garalde, and Mark Akeson. 2019. “Reading Canonical and Modified Nucleobases in 16S Ribosomal RNA Using Nanopore Native RNA Sequencing.” *PloS One* 14 (5): e0216709.
“SNaReSim: Synthetic Nanopore Read Simulator - IEEE Conference Publication.” n.d. Accessed June 21, 2019. https://ieeexplore.ieee.org/document/8031171.
Šošić, Martin, and Mile Šikić. 2017. “Edlib: A C/C Library for Fast, Exact Sequence Alignment Using Edit Distance.” *Bioinformatics*. https://doi.org/10.1093/bioinformatics/btw753.
Stöcker, Bianca K., Johannes Köster, and Sven Rahmann. 2016. “SimLoRD: Simulation of Long Read Data.” *Bioinformatics*. https://doi.org/10.1093/bioinformatics/btw286.
Tardaguila, Manuel, Lorena de la Fuente, Cristina Marti, Cécile Pereira, Francisco Jose Pardo-Palacios, Hector del Risco, Marc Ferrell, et al. 2018. “SQANTI: Extensive Characterization of Long-Read Transcript Sequences for Quality Control in Full-Length Transcriptome Identification and Quantification.” *Genome Research*. https://doi.org/10.1101/gr.222976.117.
Tischler, German, and Eugene W. Myers. n.d. “Non Hybrid Long Read Consensus Using Local De Bruijn Graph Assembly.” https://doi.org/10.1101/106252.
Vaser, Robert, Ivan Sović, Niranjan Nagarajan, and Mile Šikić. 2017. “Fast and Accurate de Novo Genome Assembly from Long Uncorrected Reads.” *Genome Research* 27 (5): 737.
Volden, Roger, Theron Palmer, Ashley Byrne, Charles Cole, Robert J. Schmitz, Richard E. Green, and Christopher Vollmers. 2018. “Improving Nanopore Read Accuracy with the R2C2 Method Enables the Sequencing of Highly Multiplexed Full-Length Single-Cell cDNA.” *Proceedings of the National Academy of Sciences of the United States of America* 115 (39): 9726–31.
Wongsurawat, Thidathip, Piroon Jenjaroenpun, Trudy M. Wassenaar, Taylor D. Wadley, Visanu Wanchai, Nisreen S. Akel, Aime T. Franco, Michael L. Jennings, David W. Ussery, and Intawat Nookaew. n.d. “Decoding the Epitranscriptional Landscape from Native RNA Sequences.” https://doi.org/10.1101/487819.
Workman, Rachael E., Alison D. Tang, Paul S. Tang, Miten Jain, John R. Tyson, Roham Razaghi, Philip C. Zuzarte, et al. 2019. “Nanopore Native RNA Sequencing of a Human poly(A) Transcriptome.” *Nature Methods* 16 (12): 1297–1305.
Wyman, Dana, Gabriela Balderrama-Gutierrez, Fairlie Reese, Shan Jiang, Sorena Rahmanian, Weihua Zeng, Brian Williams, et al. n.d. “A Technology-Agnostic Long-Read Analysis Pipeline for Transcriptome Discovery and Quantification.” https://doi.org/10.1101/672931.
Xiao, Chuan-Le, Ying Chen, Shang-Qian Xie, Kai-Ning Chen, Yan Wang, Yue Han, Feng Luo, and Zhi Xie. 2017. “MECAT: Fast Mapping, Error Correction, and de Novo Assembly for

Single-Molecule Sequencing Reads." *Nature Methods* 14 (11): 1072–74.
Yang, Chen, Justin Chu, René L. Warren, and Inanç Birol. 2017. "NanoSim: Nanopore Sequence Read Simulator Based on Statistical Characterization." *GigaScience* 6 (4): 1–6.

Sup Note A: Protocol for generating total RNA

1) Add 500 µl of TRIzol to a 1.5 ml Eppendorf with 50 mg of flies and homogenise with a disposable cell pestle.
2) Add 500 µl of TRIzol and invert multiple times to mix.
3) Incubate at room temperature for 15 minutes.
4) Centrifuge at 4oC for 10 minutes at 12000 x g.
5) Transfer supernatant to a new 1.5 ml Eppendorf.
6) Add 200 µl of chloroform and invert to mix.
7) Incubate at room-temperature for 10 minutes.
8) Centrifuge at 4oC during 15 minutes at 10000 x g.
9) Transfer supernatant to a new 1.5 ml Eppendorf tube and add 500 µl of ice-cold isopropanol.
10) Invert several times to mix and incubate 15 minutes at room temperature.
11) Centrifuge at 4oC during 10 minutes at 10000 x g.
12) Discard supernatant and add 1000 ul of 70% ice-cold ethanol.
13) Invert several times to wash the pellet.
14) Centrifuge at 4oC during 5 minutes at 10000 x g.
15) Discard the ethanol and use a sterile wipe to absorb the remaining ethanol from the tube walls.
16) Elute in 200 µl of TE. 150 fM of total RNA was used as a template for reverse transcription for use in the PCS-109 cDNA by PCR sequencing kit (Oxford Nanopore) Following the manufacturer's instructions (https://community.nanoporetech.com/protocols/cdna-pcr-sequencing_sqk-pcs109/v/PCS_9085_v109_revJ_14Aug2019).

Sup Note B: Simulation design

Before implementing our own transcriptomic read simulator, we explored the possibility of using already existing read simulators such as NanoSim (Yang et al. 2017), DeepSimulator (Y. Li et al. 2018), SimLord (Stöcker, Köster, and Rahmann 2016), and SNaReSim ("SNaReSim: Synthetic Nanopore Read Simulator - IEEE Conference Publication" n.d.). However, they are all genomic read simulators and cannot easily be modified to simulate full-length transcript reads and to output quality values, as well as to sample reads from transcripts at controlled abundances.

We downloaded 10,384 ENSEMBL transcripts from human chromosome 6 and filtered them down to 10,367 distinct transcripts (distinct meaning they are not identical sequences). We chose chromosome 6 because it harbors transcripts from difficult instances such as the highly polymorphic HLA loci. For each transcript, we assign an abundance from the set $A = \{1, 2, ..., 10, 20, 30, .., 100\}$, with the probability of an abundance a being $p_a = \frac{1}{a \sum_{b \in A} 1/b} \approx .32a$.

The effect of this simulation design is that most transcripts (32%) are expected to have an abundance of 1, and fewest transcripts (0.32%) are expected to have an abundance of 100.

Once an abundance a is chosen for a transcript, we generate a full length reads from that transcript. Over each base pair in the read we pick a quality value q uniformly at random from the set $\{0.8, 0.9, 0.92, 0.96, 0.98, 0.99, 0.995\}$. The base is assigned the Phred score corresponding to q. Then, with probability q, we make the base erroneous. We simulate the error type as either deletion, substitution, or insertion with probabilities of 0.45, 0.35, and 0.2, respectively. These values were chosen to reflect the error profile that we observed in our real data.

If the error type is a substitution, we uniformly replace the sequenced base with one of the three alternate bases and set the base's Phred quality score based on *q*. If the error is an insertion, we enter an insertion state where we generate an inserted base with probability 0.3, or exit this state otherwise. Thus, multiple bases are more likely to be inserted consecutively. Inserted bases are drawn uniformly at random over A, C, G, and T. As for the quality values in an insertion, the current base and the first insertion will have quality value corresponding to q and the rest of the inserted bases, if any, will have a Phred quality score corresponding to 0.7, so that the error rate roughly matches the error rate in the Phred quality values. If the error is a deletion, the base is simply omitted, however, the quality score will be propagated to the next base, if the next base is simulated to be correct.