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

Kristoffer Sahlin[4,5]*, Botond Sipos[6]*, Phillip L James[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
†Corresponding author, pzm11@psu.edu

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 its 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 a median accuracy of 98.9-99.6%, 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 (Bayega et al. 2018; Byrne, Cole, et al. 2019; 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 typing (Cole et al. 2019), RNA modifications (Leger et al. 2019; Sessegolo et al. 2019; Jenjaroenpun et al. 2020), 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) or genome (Workman et al. 2019; Wyman and Mortazavi 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. 2017),(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, a human PCS108 cDNA dataset, and *in silico* data. Our method opens the door for much broader application of ONT transcriptome sequencing.

Results

We will present an overview of our algorithm followed by its evaluation. For evaluation, we used one biological, one synthetic, and two simulated datasets (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.

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, with the goal of correcting 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 between similar reads, such as exon differences and variable transcription start and stop sites.

IsONcorrect works in two stages: first, partition each read into intervals and, second, error correct each interval separately. For the first stage, we start by identifying *anchor k*-mers in each read; we use anchors as a way to identify similar sequences across reads without doing alignment of whole reads. We found alignment of whole reads unreliable due to the compounded difficulties of having to span splice junctions, noisy reads, and variability in coverage. Technically, we use *minimizers* (Roberts et al. 2004) as the anchors; intuitively, it means that the chosen anchors are not too many but they are also not too far apart, guaranteeing that reads that share homologous sequence will likely also share the anchors in it.

After identifying anchors, we partition each read into a set of intervals with the following constraints: 1) each interval must begin and end at an anchor, 2) each interval's length must be above some predefined minimum and below some predefined maximum, 3) the intervals must be non-overlapping, 4) the intervals should cover as much of the read as possible, 5) the substring of each interval is found in as many other reads in the cluster as possible. This strategy is designed to make the computational problem tractable (conditions 1-3), to maximize the number of errors we correct during the second stage (condition 4), and to maximize our power during the second stage (condition 5). We note that condition 5 makes it unlikely that an interval crosses an exon/intron boundary, since such an interval would be found in less reads than either of the split intervals.

In the second stage of isONcorrect, we take each read and correct each interval in its partition one at a time. We identify the interval's flanking anchors and pull out all other read substrings that are flanked by the same anchors. All these substrings are then aligned to each other to generate an alignment matrix; for each column in the matrix, we identify values that have high enough support and flag them as trusted variants. We correct the read by choosing the trusted variant whose surrounding sequence (i.e. the *k* nucleotides around it) has the smallest edit distance to the equivalent region in the read.

Please see the Methods section for a more detailed description of our algorithm, including some heuristic modifications not discussed here.

Error rate analysis

We sequenced the transcriptome of a Drosophila sample using ONT, with a total of 4,350,977 reads (Table 1). From these, we ran pychopper and identified 3,646,342 reads as being end-to-end (which we call full length) and at least 50nt long. These reads had a median length of 559nt, and we processed the reads further by and error correcting 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 1.1% (Table 2).

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 3.5 million full length reads from transcripts at controlled relative abundances (in the range of 1 to 100) from transcripts (Table 1) (for details of the simulations, see Supplementary Note A). We denote this dataset SIM-full. 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 7.0% to 0.4% (Table 1). Full error rate distributions for before and after error correction with isONcorrect are shown in Fig. S1. We also studied how the post-correction error rate was affected by repetitive regions within the transcripts in the SIM-full dataset (Supplementary Note B). We did not observe clear dependence of the average repetitiveness of the transcript sequence and the post-correction error rate (Fig. S2).

Unfortunately, while eliminating the effect of biological variability on error rate measurement, simulated data does not always 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 0.4% after (Table 2), a significant reduction.

Error profiles

We also investigated the error profiles of the current ONT cDNA 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 2) was lower than previously published cDNA ONT datasets (about 13-14%, Table 2 (Workman et al. 2019)), likely due to improvements in experimental protocol and the base calling software. The substitution, insertion, and deletion error rate was 2.5%, 2.2%, and 3.0%, respectively (Fig. 1C). We observed a similar distribution for Drosophila (2.6%, 2.1%, and 3.2%), with the caveat that it also includes true biological variation (Fig. 1C). Error correction substantially reduced the error rate in each category. The substitution, insertion, and deletion rates of SIRV reads fell to 0.1%, 0.4%, and 0.3%, respectively, after correction (Fig. 1C).

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 simulated a dataset with a controlled read depth for each of the 10,367 distinct chr6 transcripts (Table 1). For details of the simulations, see Supplementary Note A. We call this dataset SIM-ca. As expected, the post-correction error rate decreased as a function of depth (Fig. 2A). Compared to the median pre-correction error rate of about 6.95%, the median post-correction error 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 much 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 5% in SIRV and to 3% in simulated data. This is due to isONcorrect's ability to jointly use all isoforms from a gene or gene-family 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. Transcripts occurring at very low read depths that have exons with unique mutations or small variations in splice junctions may however be overcorrected. We investigate such artifacts in the following sections.

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 12,982, respectively, with 178 lost and 99 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 178 transcripts that were not captured by a FSM read after correction, 107 and 37 of them had only one and two FSM original reads, respectively, and all but four of them occurred in less than 10 original reads. Therefore, a consequence of our correction algorithm is that the lowest abundant transcripts may be mis-corrected. However, we also observed 99 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 in a way that the read more closely aligns to the reference genome. For example, such overcorrection can happen both because of allele specific or gene-copy specific variation that produce similar transcripts differing in only SNPs. More formally, we classified a simulated 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. 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 SIM-ca simulated dataset, where the true transcript is known.

The overcorrected reads made up less than 0.6% of all reads (359 out of 59,440; Fig. 4). 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. We observed this in 76 original reads (21% of overcorrected reads), i.e. reads that aligned better to a transcript different to the true transcript even before error correction.

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 isONcorrect's overcorrected reads, i.e. 5 or less positions in >80% of the overcorrected reads. In addition, the overcorrection was mostly limited to reads at low abundances, with 47% of overcorrected reads coming from transcripts with an abundance of ≤ 5. This indicates that overcorrection with isONcorrect is mostly limited to SNPs or short indels, as opposed to larger exon-level miscorrections.

Effect of pre-correction error rate

Since the error rate of the sequencing technology can change over time, we investigated how it would affect the performance of isONcorrect. Our SIM-ca dataset had a median pre-correction error rate of 7% and we will refer to it as SIM-ca-7 in this section. We also generate datasets SIM-ca-4 and SIM-ca-11 which had median pre-correction error rates of ~3.94% and ~11.40%, respectively (Supplementary Note A).

The post-correction error rates, as a function of depth, are shown in Figure 2A for SIM-ca7, Figure S3A for SIM-ca-4, and Figure S3B for SIM-ca-11. The overall post-correction median error rate was 0.4%, 0.6%, and 1.7% for SIM-ca-4, SIM-ca-7, and SIM-ca-11, respectively. We observe in these figures that the tails are longer for higher pre-correction error rates, i.e. for higher pre-correction error rates, more read depth is needed to achieve a stable post-correction error rate.

We also measured how overcorrection was affected by the pre-correction error rate (Fig. S3C-D). The percentage of overcorrected reads were 0.3% in SIM-ca-4 (Fig. S3C), 0.6% in SIM-ca-7 (Fig. 4), and 0.8% in SIM-ca-11 (Fig. S3D). Similar to what we previously observed in the SIM-ca-7 dataset, overcorrection in SIM-ca-4 and SIM-ca-11 was mostly limited to reads that came from transcripts with multiple similar copies and that occurred at relatively low abundances (Fig. S3C-D).

Overall, we conclude that isONcorrect behaves as expected; i.e. the post-correction error rate and the number of overcorrected reads are lower when the pre-correction error rate is lower.

isONcorrect on an older ONT protocol

We also investigated isONcorrect's performance on a dataset from an older cDNA protocol (denoted ONT-old; Table 1) of human ONT cDNA sequencing with a minION (Workman et al. 2019). This dataset has a pre-correction error rate of 13.4%. On this dataset, isONcorrect is able to reduce the error rate to 3.4%, increase the number of aligned reads (Table 2, Fig. S4A), and increase the number of FSM reads (Fig. S4B). The 3.4% post-correction error rate is due to the higher pre-correction error rate and is consistent with our analysis on simulated datasets. For example, in the SIM-ca-7 and SIM-ca-11 datasets, a 63% (11.4/7.0) increase in pre-correction error rate leads to a 183% (1.7/0.6) increase in post-correction error rate. In comparison, in the two biological datasets Drosophila and ONT-old, we have a 91% (13.4/7.0) increase in pre-correction error rate, which leads to a 209% (3.4/1.1) increase in post-correction error rate. However, we do not exclude the possibility that the complexity of the transcriptome can be a contributing factor to this increased post-correction error rate..

On this dataset, in contrast to the *Drosophila* dataset, we observed more reference transcripts that have at least one FSM read in the corrected reads compared to the original reads (7,788 and 7,710, respectively, with 377 lost and 462 gained). We investigated the lost transcripts after correction as a function of how abundant they were in the original reads (Fig. S4C). Out of the 377 transcripts that were not captured by a FSM read after correction, 299 and 44 of them had only one and two FSM original reads, respectively, and all but six of them occurred in less than 10 original reads. Similarly to what we observed in the *Drosophila* dataset we observe that the lowest abundant transcripts may be mis-corrected.

Effect of number of isoforms

Using the SIRV data we also investigated how the post-correction error rate changed with different numbers of isoforms per gene and with read depth. We performed subsampling of reads from three gene loci in the SIRV data that had at least 8 isoforms each (SIRV3, SIRV5 and SIRV6). Our subsampling was performed as in the read depth experiments (details in Data Analysis in methods section under SIRV subsampling experiments).

We found that the same post-correction error rate was achieved regardless of the number of isoforms (Fig. S5). However, less reads per isoform were needed to achieve equivalent error rates for experiments with more isoforms. This may seem counterintuitive but is due to isONcorrect's ability to combine reads from different isoforms for the purposes of error correction. For example, a median error rate of 2% is reached after 3, 4, and 5 reads per isoform in the datasets with 8, 4, and 1 isoforms, respectively.

We also observe that while higher numbers of isoforms do not affect the post-correction error rate, they do result in a higher fraction of overcorrected reads (Fig. S6). When we aggregate the overcorrection percentages for experiments that had the same number of isoforms, they are 2.7%, 1.6%, and 0.3% for the experiments with eight, four, and one isoforms, respectively. Increased read depth does not help to alleviate the problem, because the relative depth of the confounding isoforms does not change (Fig. S6). However, we note that the overcorrection is again mostly due to small variations rather than exon-level changes; for example, we observed that 57% of the overcorrected reads (across all experiments) switched isoforms between SIRV506 and SIRV511, which are identical except for a splice junction difference of 6 nt.

Overcorrection of SNPs with allele-specific expression

To study how well isONcorrect can distinguish errors from allele-specific SNPs, especially when the alleles have different expression levels, we simulated data reads from two transcripts in an isolated setting. We started with two identical copies of a transcript from the RTN4IP1 gene of length 208nt. In each experiment, we inserted a SNP in one of the copies at a randomly chosen position (between 50 and 150nt). We then simulated a given number of reads (*n*) (at a 7% error rate as in our other simulations), with a given fraction (*f*) coming from the mutated transcript. We varied the number of reads *n* from 10 to 100 and the relative frequency of the SNP (*f*) between 10% and 50%. For each *n* and *f*, we ran 10 replicate experiments. We then measured the fraction of replicates which had at least one SNP-containing read after error correction. Our experimental design is intended to vary the sequence context of the SNP and the locations of the errors while controlling for read depth and SNP frequency.

Figure S7 shows the result. At least 20 reads are needed to retain SNPs at 20% or 30% frequency with a probability of 80%. For SNPs at 10% frequency, at least 50 reads are needed. This corresponds to about 4-6 reads needed to cover the minor SNP allele in order to avoid overcorrecting it.

IsONcorrect's behavior is guided by a formula which is a function of the number of reads coming from the minor allele and the sequence context around the SNP (see Algorithmic Details in Methods). Intuitively, in the simple case of an isolated SNP with no sequencing errors within *k/2* nucleotides, the formula will retain the SNP if its frequency is at least 10% and there are at least 3 reads covering it. When there are multiple SNPs nearby, this threshold decreases, while when there are sequencing errors nearby, this threshold increases.

Overcorrection of exons

We study how error correction with isONcorrect preserves the exon structure for transcripts at different abundances, using both simulated and SIRV data. For the simulated data, we used a similar design to the allele-specific expression experiment. We started with two identical copies of a transcript from the RTN4IP1 gene. In each experiment, we removed a substring (of length 5, 10, or 20nt) in one of the copies at a randomly chosen position (between 50 and 150nt). We chose short length substrings because shorter exons pose a bigger challenge to isONcorrect than longer ones. We then simulated a given number of reads (*n*) (at a 7% error rate as in our other simulations), with a given fraction (*f*) coming from one of the two transcripts (chosen at random). We varied the number of reads *n* from 10 to 100 and the relative frequency of the minor isoform (*f*) between 10% and 50%. For each *n* and *f*, we ran 10 replicate experiments. We then report the fraction reads with best alignment to the minor isoform before and after correction (Figure S8). We observe that substantial overcorrection occurs when the exon is very small (5nt), at all read depths and minor isoform fractions. However, no noticeable overcorrection occurs at 20bp exons, and the overcorrection for 10bp exons is negligible for most downstream analyses.

For the SIRV data, we took the only two available isoform-pairs, SIRV506 and SIRV511, and SIRV606 and SIRV616 that are identical besides differing in an internal splice site. This internal splice difference gives rise to a 6nt and 14nt deletion, respectively, between two otherwise identical transcripts. We then used the subset of the SIRV reads that had primary alignments (using minimap2) to each of the four transcripts, and furthermore, an edit distance strictly smaller to one of the transcripts. The edit distance constraint was invoked because some reads were ambiguous to which transcript they belonged, particularly for the transcript pair with 6nt distance. We then subsampled reads from the subset of reads from each isoform at different controlled depths and fractions and computed the overcorrection in the same manner as for the simulated dataset. As for the simulated dataset we report the fraction reads with best alignment to the minor isoform before and after correction (Fig. S9). In the case of the 6nt deletion, we observe substantial overcorrection at low (≤ 20 reads) read depths. For the 14nt deletion, we observe overcorrection similar to the case of 5nt in the simulated experiment. The change in isoform fraction is up to 10-20 percentage points. The reason for the difference between SIRV data and our simulated results may be due to the sequence of the transcript (and, in particular, the deleted exon is fixed compared to simulated data, where it is randomly varied across experiments), the ONT specific error profile, selection of reads in the experiment, or to other unknown causes. In summary, we observe that while exon overcorrection is in general more frequent in our SIRV data than on the simulated data, substantial overcorrection occurs when the exon is small and read depth is low for the minor isoform (Fig. S9A).

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. S10). 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. S10). 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. S11), 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 3). 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.

The current memory usage requires a large memory server to run. We note that in our simulated data, some transcripts were very long (>20,000 nucleotides). This resulted in a 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).

Comparison against other tools

As (Lima et al. 2019) found that applying genomic error correctors to ONT transcriptome data had undesirable downstream effects, we only evaluated the genomic error corrector CONSENT (Morisse et al. 2019) that was not evaluated in (Lima et al. 2019) as well as canu (Koren et al. 2017) and concluded similarly to (Lima et al. 2019) that they are not suitable for long transcriptomic reads (see Supplementary note C). We instead focused on comparing our tool to a recent long transcriptomic read error correcting tool RATTLE (de la Rubia et al., 2020).

We ran RATTLE on our five datasets (Table 1) and results are shown in Table 2. RATTLE showed comparable median error rates to isONcorrect on the *Drosophila,* SIRV,SIM-ca*,* and ONT-old datasets (0.9%, 1.1%, 0.4%, and 3.1% for RATTLE and 1.1%, 0.4%, 0.6%, and 3.4% for isONcorrect, respectively). However, we observed that a significant fraction of reads were miscorrected by RATTLE, which we define as reads that have more errors after correction than before. Specifically, the percentage of reads that were miscorrected by RATTLE were 1.5%, 32%, and 11.3% for the *Drosophila*, SIRV, and SIM-ca datasets, compared to 0.3%, 0.2%, and 0.4% by isONcorrect (Table 2). For Drosophila and SIRV, these numbers may be influenced by mapping ambiguities, but for the simulated dataset they are exact. For the ONT-old dataset, we observed less reads classified as miscorrected for RATTLE (1.4%) compared to isONcorrect (2.4%).

We further studied the effect of error correction to alterations in splice structure, a negative side effect of error correction that our mis-correction measure does not explicitly capture. We used the *Drosophila* and the ONT-old data. Similarly to isONcorrect, RATTLE increased the number of reads fully matching an annotated transcript (FSM) on both Drosophila (Fig. S12A) and ONT-old (Fig. S12B). However, for both Drosophila and ONT-old, we observed significantly less reference transcripts having at least one RATTLE corrected FSM read (9,893 and 5,597, respectively) than having at least one original FSM read (13,062 and 7,710, respectively). This corresponds to a 24.3% and 17.4% reduction, respectively, compared to only a 0.6% reduction and 0.1% increase for isONcorrect. Specifically, for Drosophila RATTLE lost 3,248 and gained

79 transcripts with at least one FSM, while isONcorrect lost 180 and gained 100 transcripts. In some cases, transcripts that had as many as 756 original FSM reads were lost (Fig. S12C). For ONT-old, RATTLE lost 2,519 and gained 406 transcripts with at least one FSM, while isONcorrect lost 377 and gained 462 transcripts. Similarly to the Drosophila dataset, we observe a substantial alteration of splice sites in the FSM reads for the ONT-old dataset after correction with RATTLE (Fig. S12D) compared to isONcorrect (Fig S4C). Our splice site analysis on the two biological datasets showed that RATTLE correction can miscorrect substantially more correct splice sites than isONcorrect.

We further investigated RATTLE's overcorrection using the SIM-ca dataset (Fig. S12E), in the same way we did for isONcorrect (Fig. 4). We observed that 16.7% (9,947 out of 59,440) of RATTLE corrected reads aligned with better identity to a transcript other than the true transcript (compared to just 0.6% for isONcorrect). These overcorrected reads were found throughout different abundances and different magnitudes of overcorrection.

We also used the SIM-ca dataset to analyze the effect of read depth on correction accuracy of RATTLE (Fig. S12F). We observed that RATTLE needs a depth of at least six reads to decrease the median error rate (Fig. S12F), but the standard deviation of the median remains large even at larger depths because of the large fraction of miscorrected reads (11.3%; Table 2). IsONcorrect on the other hand decreases the median error rate for reads already at depth 1, with a narrower standard deviation due to significantly fewer miscorrected reads (Fig. 2A). Reducing error rate for reads at low depths is expected with isONcorrect, as it is designed to leverage shared exons, while RATTLE is not designed to make use of shared exons between different isoforms in the error correction.

As for resource utilization, RATTLE’s clustering and correction pipeline is faster than that of isONclust and isONcorrect (Table 3). However, RATTLE uses more memory (3 times of isONcorrect for Drosophila) and exceeded 500Gb on the human SIM-full dataset. It was not able to complete on SIM-full because it exceeded the available memory on the server.

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 1.1%. 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 2), with the resulting mismatch rates between 0.4 - 1.1%. We also observe that the correction is not always correct, with 0.6% of the reads being overcorrected, and that SNPs or short indels occuring at low relative abundance are not always retained.

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 3-5%) for transcripts with just one read. However, there is a disadvantage that one gene's transcript that shares an exon with another gene's transcript may get miscorrected if it occurs at a much lower frequency.

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.

Our strategy for partitioning of the read into separate intervals is based on a related idea used in the context of genomic error correction in the algorithm CONSENT (Morisse et al. 2019). CONSENT identifies alignment piles within reads, i.e., a set of reads aligning to the same region, where the alignments are produced by a third party genomic read aligner. Within an alignment pile, CONSENT finds the longest collinear chain of anchor k-mers shared by at least a threshold of the reads in the pile. The collinear regions are then used to split the pile into segments which are error corrected separately. This approach was shown to reduce run-time (Morisse et al. 2019) while solving a simpler local and hopefully less error-prone problem. However, it does not guarantee that the k-mer anchors are chosen so that the segmentation solution is optimal with respect to read coverage across the segments in the read. We make use of the idea of interval segmentation and anchor k-mers, but adapt the approach to a 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 the anchor k-mer depth. In addition, we cannot rely on third party tools to find self-to-self transcriptome read alignments, as this is a challenging problem due to the variable abundance.

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.

Methods

isONcorrect algorithm

The Results section contains a high-level overview of our algorithm. Here, we describe isONcorrect in detail, by giving the relevant definitions, the steps of the algorithm, and finally the heuristic modifications to improve run-time. The steps of the algorithm are illustrated in Fig. 5.

Definitions

Let r be a string of nucleotides that we refer to as a read. We use $r[i]$ to refer to the i-th position of r. 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 $[p, p+w-k+1)$. 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 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)

We will use $ed(\cdot)$ to refer to the edit distance between the two strings, and we use edlib (Šošić and Šikić 2017) to calculate the edit distance. We will use $HC(\cdot)$ to denote the homopolymer compression function. Given a string s, $HC(s)$ removes from s all characters identical to the preceding one, i.e. $HC(ATTTCAA) = ATCA$.

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 $a(q-p-k)$, where a denotes the *support* of the interval. We compute the support as the number of occurrences of (m_1, m_2) in $StrW$ 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 the edit distance is less than $|s|(\varepsilon_s + \varepsilon_{s'})$, where ε_s (respectively, $\varepsilon_{s'}$) is the average per-base error rate of s (respectively, s'), inferred from the quality values. If a read has multiple (possibly disjoint) intervals matching a single interval of r, only the one with the smallest edit distance is considered.

Next, for the read r, we send the instance of all intervals and their weights to a weighted interval scheduling algorithm (Step 2). This gives us a set of disjoint intervals in r, with, intuitively, a preference of a combination of intervals that are highly supported and covering as much of the read as possible. For each interval, we then send for correction the corresponding segment of the read and all the supporting segments.

The segment correction is performed as follows. We build a partial order alignment graph (Lee, Grasso, and Sharlow 2002) from the segments using SPOA (Vaser et al. 2017) c using the heaviest bundle algorithm (Lee 2003) (Step 3). Next, we create a multi-alignment matrix A from the pairwise alignments of all the segments to the consensus (we use the method described in (Sahlin et al. 2018)). We generally find this matrix reliable because we expect our segments to be easy to align, since they do not cross exon/intron boundaries.

Given A and c, we create for every column j of A a set of *trusted contexts and variants*, as follows. Let A' correspond to the submatrix of A from column $j - floor(k/2)$ to $j + floor(k/2)$ (inclusive). A row of A' is classified as a *trusted* context if it occurs more than a certain number of times in A' (to be made more precise below). The value of this row at the column corresponding to column j in A is called the trusted variant. This can be a nucleotide or a deletion symbol. We also add as a trusted context the segment of the consensus $c' = c[j - floor(k/2) : j + floor(k/2)]$ and its corresponding variant $c[j]$.

We decide that a subsegment b is a trusted context if it occurs at least $max(3,\ mT/min(ed(c', b),\ ed(H(c'),\ H(b))))$ times as a row in A'. Here, m is the number of segments sent for correction, and T is a parameter with default value of 0.1. The edit distance term lowers the required threshold for more dissimilar segments to the consensus. This means that lower read depth is required to preserve co-occurring variants (relative to the consensus), and is beneficial under the assumption that co-occurring variants are more common across different reads than co-occurring errors. We use the edit distance between both the original and the homopolymer compressed segments in the denominator to account for the higher number of errors associated with homopolymer lengths. If the strings are identical under homopolymer compression (denominator of 0) the variant (which is a homopolymer length variation) is not considered. Furthermore, we require a variation to be present in at least three reads to be considered.

Once the trusted contexts and variants are established, we error correct the segment of the read r, as follows. Let a be the row of A corresponding to r. For every position j in A, we find the trusted variant whose context has the smallest edit distance to a. We replace $a[j]$ with this variant if it is different than $a[j]$. These updates are then projected back onto the sequence of the read. Optionally, we correct the segments in A' belonging to other reads as well (Step 7), described below in heuristic modifications.

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 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 it to make it faster. We refer to the modified algorithm as "approximate." The time bottleneck of the exact algorithm 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 scheduling 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 D 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 are available on ENA under project accession number PRJEB34849.

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. We kept all reads classified as full length and longer than 50 nucleotides.

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 of k=9 and xmax=80, while xmin is set to 2k and w is chosen adaptively. These settings were used for all datasets except SIM-ca-11, where we adjusted the value of k to account for the high error rate (we used k=11 for isONclust and k=7 for isONcorrect). We also used the parameter '--split_wrt_batches' for the SIRV dataset, which parallelizes the correction within clusters and is more effective for datasets with a small number of large clusters. To the RATTLE analysis, we ran RATTLE (git commit 3c1d21bb, Feb 21st, 2020) in clustering and correction modes with default parameters, as was done in (de la Rubia et al., 2020) for ONT cDNA reads. For clustering, the following default parameters are set: k=10, s=0.20, v=1000000, iso-score-threshold=0.30, iso-kmer-size=11, iso-maxvariance=25, and p=0.15. For correction, the following default parameters are set: g=0.3, m=0.3, and s=200.

Inferring read error rates from alignments

For Drosophila and ONT-old 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) and human reference genome (hg38) using minimap2 with parameters: -w4 -k 13 -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 an annotated transcriptome database. We use Ensembl release 97 annotated on assembly BDGP6.22 for the Drosophila data, and Ensembl release 101 for the ONT-old assembly. 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 and simulated 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 -w1 -k 8 -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 the analyses we did with subsampled SIRV data. We aligned the 1,514,274 full-length SIRV reads to the remaining 63 SIRV transcripts.

For the experiment where we investigated error rates as a function of read depth, we 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 simulations.

For the experiment where we investigated error rates as a function of both read depth and number of isoforms, we ran 60 experiments with varying numbers of isoforms and read depth, with 10 replicates in each experiment. Concretely, for each of the three SIRV genes, for each $y \in \{1, 4, 8\}$, and for each value of x between 1 and 20, an experiment was performed. In the experiment, we randomly picked y isoforms of the given gene and, for each isoform, randomly sampled x reads from the reads aligning to that isoform. Each experiment was repeated 10 times to obtain replicates. Each set of replicates resulted in a controlled dataset with 10 replicates and each replicate with $3 \cdot x \cdot y$ reads and a depth of x reads per transcript.

For each simulation in above described experiments, we clustered the reads with isONclust (git commit 5b969b6d) with default parameters for ONT data. Then, we ran isONcorrect on the clusters, using the default parameters k=9, xmin=2k, xmax=80, and --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 Fig. S11). 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.

Data availability

The SIRV and Drosophila data are available on ENA under project accession number PRJEB34849.

Code availability

All scripts used for simulating datasets as well as to run the evaluation are found at https://github.com/ksahlin/isONcorrect. Specifically, the evaluation pipeline for simulated data is found under https://github.com/ksahlin/isONcorrect/tree/master/evaluation_sim, for SIRV data under https://github.com/ksahlin/isONcorrect/tree/master/evaluation_sirv, and for biological data under https://github.com/ksahlin/isONcorrect/tree/master/evaluation. The source code of isONcorrect is available at https://github.com/ksahlin/isONcorrect.

Competing interests

Data generation was funded by Oxford Nanopore Technologies (ONT). BS and PJ are ONT employees and stock option holders.

Author contributions

KS and PM conceived the project. K.S. designed the algorithm with input from PM and BS. KS implemented the algorithm. K.S. and PM designed the evaluation and experiments. KS implemented the evaluation. KS and PM analyzed the results with input from BS. BS implemented the preprocessing tool pychopper with input from KS. PLJ performed the wet-lab experiments. K.S. and PM drafted the paper with contributions from BS. All authors critically revised the text.

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).

Main Tables and Figures

Dataset	Sequencing chemistry/kit)	# unique transcripts	# reads	# inferred full length eads (over 50nt)	Median length of full length reads (nt)
SIM-full (chr6)	simulated	10,367	3,500,000	3,500,000	1035
SIM-ca (chr6)	simulated	10,367	57,804	57,804	1042
SIRV	ONT R9/PCS109	68	1,680,000	1,514,274	540
Drosophila	ONT R10/PCS109	NA	4,350,977	3,646,342	559
ONT-old	ONT R9/PCS108	NA	890,503	890,503*	695

Table 1. Datasets used in evaluation of the transcriptomic Oxford Nanopore Sequencing datasets. *Dataset has been preprocessed in (Workman et al. 2019).

Dataset	Uncorrected full length reads		isONcorrect corrected full length reads			RATTLE corrected full length reads		
	#aligned	Median diff to ref(%)	#aligned	Median diff to ref(%)	# reads with more errors after correction	#aligned	Median diff to ref(%)	# reads with more errors after correction
SIM-full	3,500,000	7.0	3,500,000	0.4	1,578	-	-	-
SIM-ca	57,804	7.0	57,804	0.6	214	57,804	0.4	6,541
SIRV	1,512,325	6.9	1,512,450	0.4	2,295	1,512,929	1.1	488,593
Drosophila	3,327,355	7.0	3,369,876	1.1	11,369	3,395,958	0.9	52,125
ONT-old	874,531	13.4	879,188	3.4	21,441	875,346	3.1	13,233

Table 2. Read statistics before and after error correction.

Dataset	pychopper		isONclust		isONcorrect		RATTLE cluster		RATTLE correct	
	Peak memory	Runtime	Peak memory	Runtime	Peak memory	runtime	Peak memory	runtime	Peak memory	runtime
Drosophila	5Gb	32m	34 Gb	2h 05m	102 Gb	43h 54m	48 Gb	16 h 52m	314Gb	1h 34m
SIM-ca	-	-	8 Gb	0h 08m	99 Gb	1h 40m	2Gb	14m	83Gb	9m
SIRV	3Gb	14m	10 Gb	0h 9m	127 Gb	6h 13m	19 Gb	28m	18Gb	49m
SIM-full	-	-	49Gb	2h 13m	385 Gb	54h 49m	108 Gb	1h 29m	-	-
ONT-old	-	-	24Gb	4h 43m	89Gb	11h 11m	14 Gb	11 h 51m	14Gb	23m

Table 3. Runtime and memory usage of error correction pipelines. RATTLE correct on SIM-full did not complete because it exceeded the 500Gb memory capacity of our server (we tried using 1,2,4,8,16,32, and 60 cores).

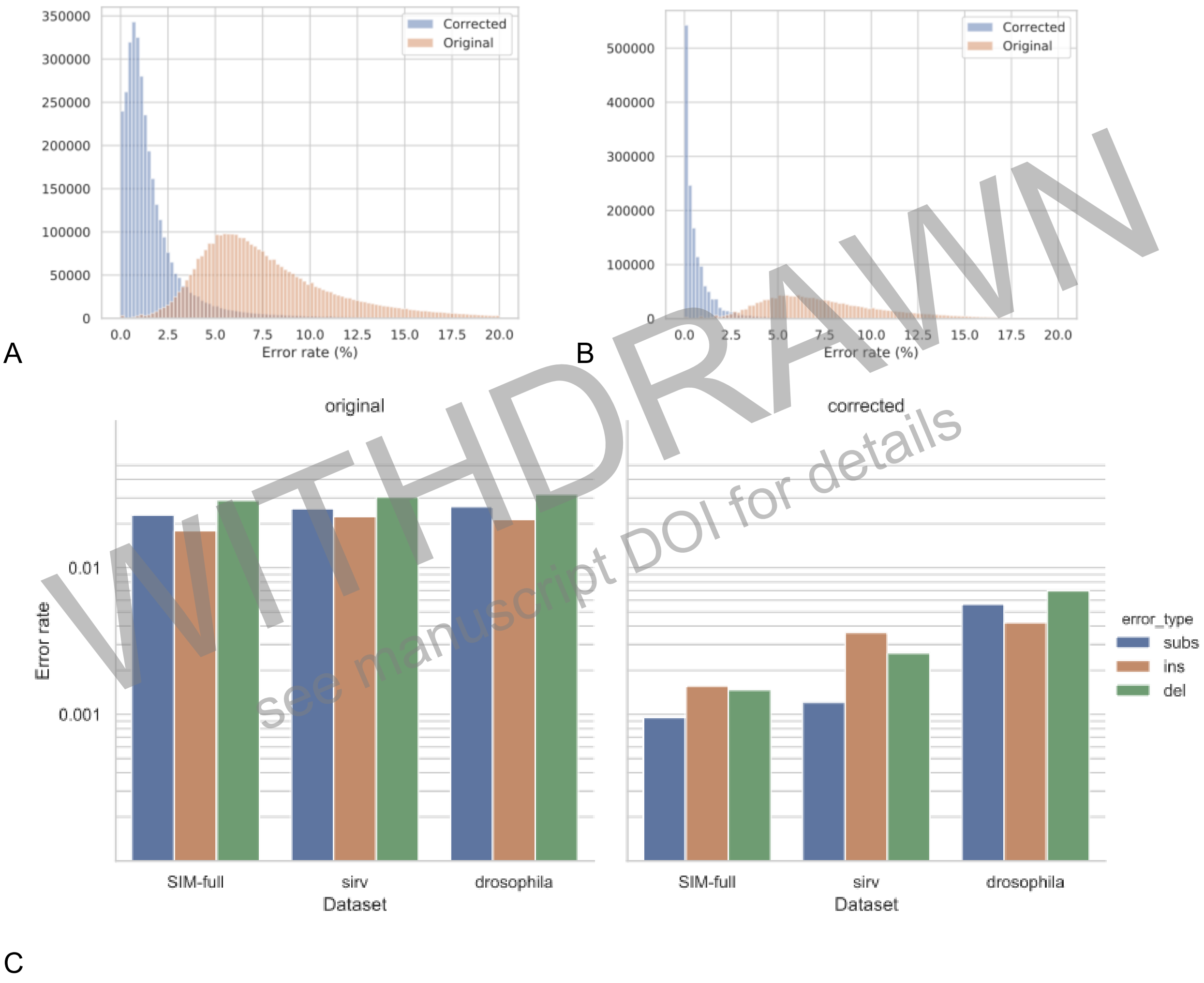

Figure 1: Error rates of ONT reads before and after error correction. (A) Alignment difference distribution of corrected and original Drosophila reads. Differences can arise both from sequencing errors and variation to the reference genome. (B) Error rate distribution of corrected and original SIRV reads, 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.

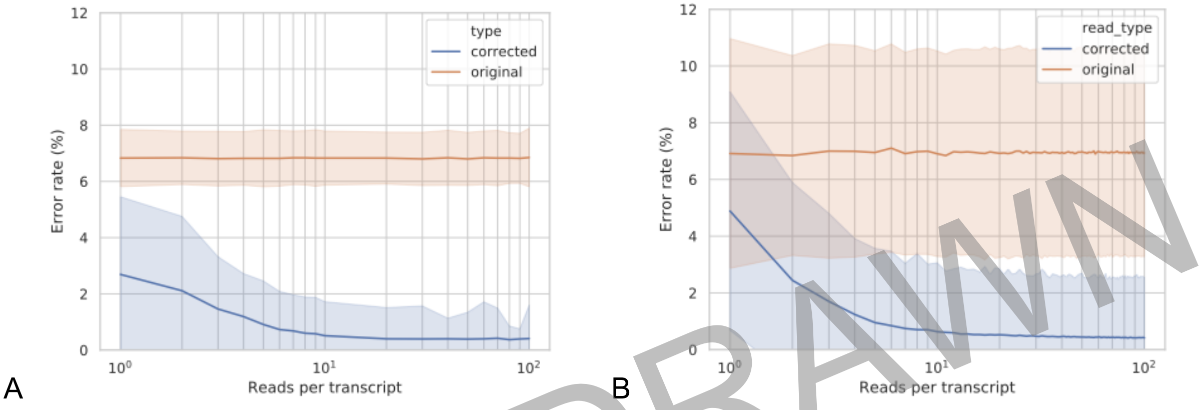

Figure 2: Effect of read depth on error rate. Panel (A) shows the median error rate of the SIM-ca read experiment based on true read depth of the transcript (i.e. number of reads sequenced from it). Panel (B) shows the median error rate of the SIRV data as a function of read depth, obtained via subsampling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

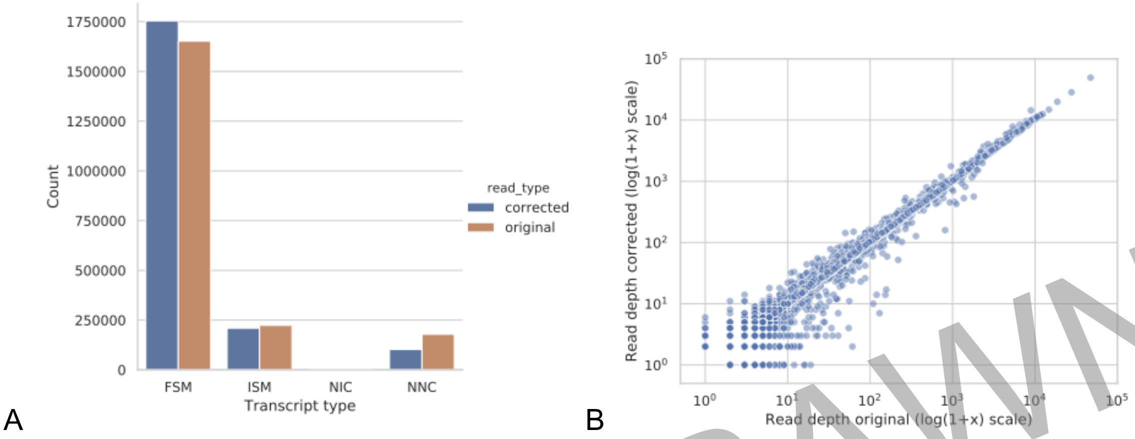

Figure 3. Splice site accuracy before and after error correction in the Drosophila data. (A) Total number of reads classified per splice site category, using the terminology of (Tardaguila et al. 2018). FSM stands for full splice match, ISM stands for incomplete splice match, NIC stands for novel-in-catalogue, 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.

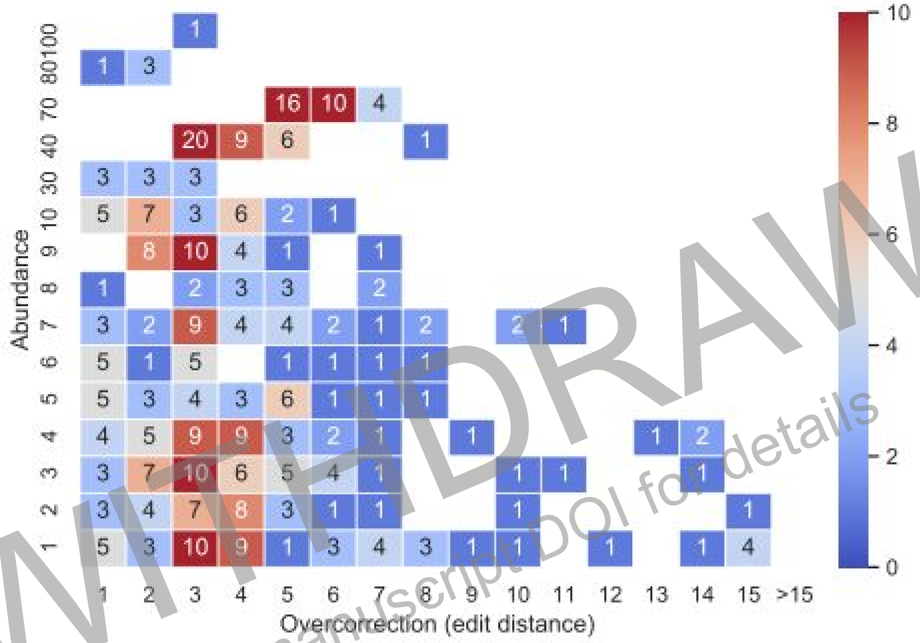

Figure 4. The effect of overcorrection in SIM-ca. We bin each overcorrected read according to the abundance of its true transcript (y-axis) and its overcorrection distance (x-axis). Each cell shows the number of reads in the bin.

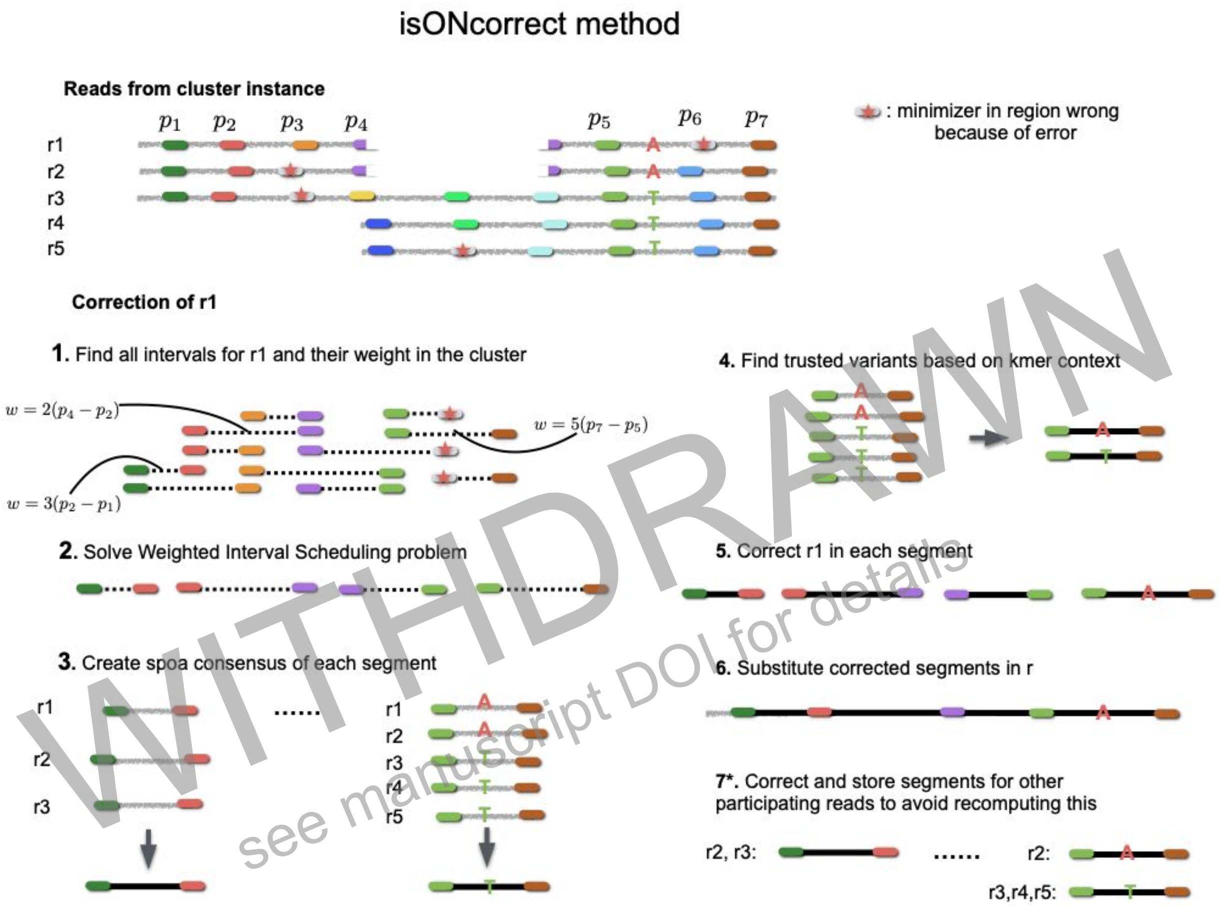
isONcorrect method
Reads from cluster instance
p_1 p_2 p_3 p_4 p_5 p_6 p_7
r1
r2
r3
r4
r5
A
T
: minimizer in region wrong because of error
Correction of r1
1. Find all intervals for r1 and their weight in the cluster
$w = 2(p_4 - p_2)$
$w = 5(p_7 - p_5)$
$w = 3(p_2 - p_1)$
2. Solve Weighted Interval Scheduling problem
3. Create spoa consensus of each segment
r1
r2
r3
r1
r2
r3
r4
r5
4. Find trusted variants based on kmer context
5. Correct r1 in each segment
6. Substitute corrected segments in r
7*. Correct and store segments for other participating reads to avoid recomputing this
r2, r3:
r2:
r3,r4,r5:

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 anchor minimizers (shown as colored blocks) and adds them to a hash table. To correct a single read (e.g. r1), all the anchor minimizer pairs found in r1 are queried in the hash table, and all reads containing a given anchor minimizer pair are retrieved. In this example, r1 has 11 such anchor pairs (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 anchor pair (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 anchor pairs with the largest weight (step 2). In this case, four anchor pairs are selected. All segments between the chosen anchor pairs are sent for correction. A consensus is created (step 3) using spoa, and one or more trusted variants are identified, based on their frequencies and sequence contexts (step 4). Each read segment in r1 is corrected to the closest trusted context (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 anchor pair (p1,p2) may be again encountered in the optimal scheduling solution, allowing steps 3-5 to be skipped at that point.

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.

Jenjaroenpun, Piroon, Thidathip Wongsurawat, Taylor D. Wadley, Trudy M. Wassenaar, Jun Liu, Qing Dai, Visanu Wanchai, et al. 2020. “Decoding the Epitranscriptional Landscape from Native RNA Sequences.” *Nucleic Acids Research*, July. https://doi.org/10.1093/nar/gkaa620.

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. 2017. “Canu: Scalable and Accurate Long-Read Assembly via Adaptive K-Mer Weighting and Repeat Separation.” *Genome Research* 27 (5): 722–36.

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.

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.

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.

Roberts, M., W. Hayes, B. R. Hunt, S. M. Mount, and J. A. Yorke. 2004. “Reducing Storage Requirements for Biological Sequence Comparison.” *Bioinformatics*. https://doi.org/10.1093/bioinformatics/bth408.

De la Rubia, Ivan, Joel A. Indi, Silvia Carbonell, Julien Lagarde, M. Mar Albà, and Eduardo Eyras. n.d. “Reference-Free Reconstruction and Quantification of Transcriptomes from Long-Read Sequencing.” https://doi.org/10.1101/2020.02.08.939942.

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.

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.

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 E. W. Myers. n.d. "Non Hybrid Long Read Consensus Using Local de Bruijn Graph Assembly. bioRxiv. 2017." *DOI* 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.

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, and Ali Mortazavi. 2019. "TranscriptClean: Variant-Aware Correction of Indels, Mismatches and Splice Junctions in Long-Read Transcripts." *Bioinformatics* 35 (2): 340–42.

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.

Supplementary Figures and Notes

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

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

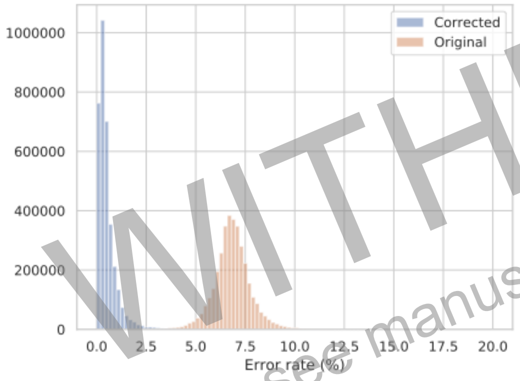

Figure S1: Error rates of the SIM-full reads before and after error correction by isONcorrect.

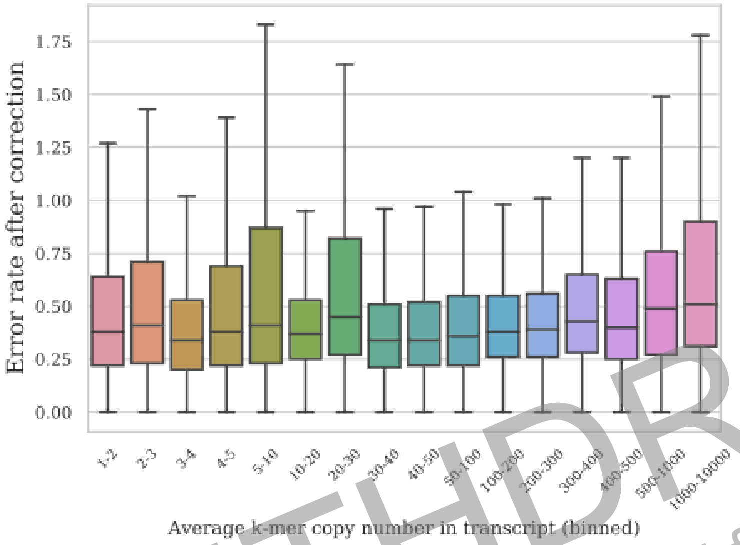

Figure S2. Error rates of reads after correction broken down per average mappability of transcripts that the read originates from. The average mappability is derived from a kmer size of 31.

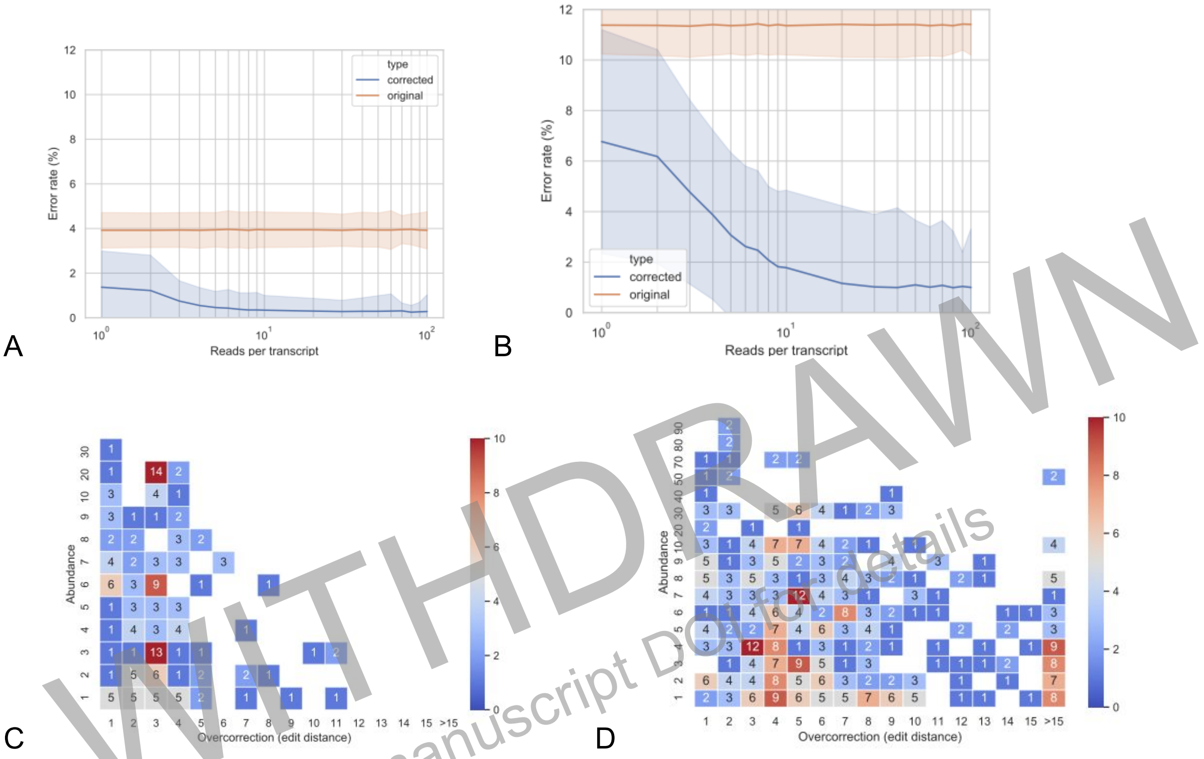

Figure S3. The effect of varying pre-correction error rates on isONcorrect. Effect of read depth on error rate for the (A) SIM-ca-4 dataset and (B) SIM-ca-11 dataset. Overcorrection in the (C) SIM-ca-4 dataset and (D) SIM-ca-11 dataset.

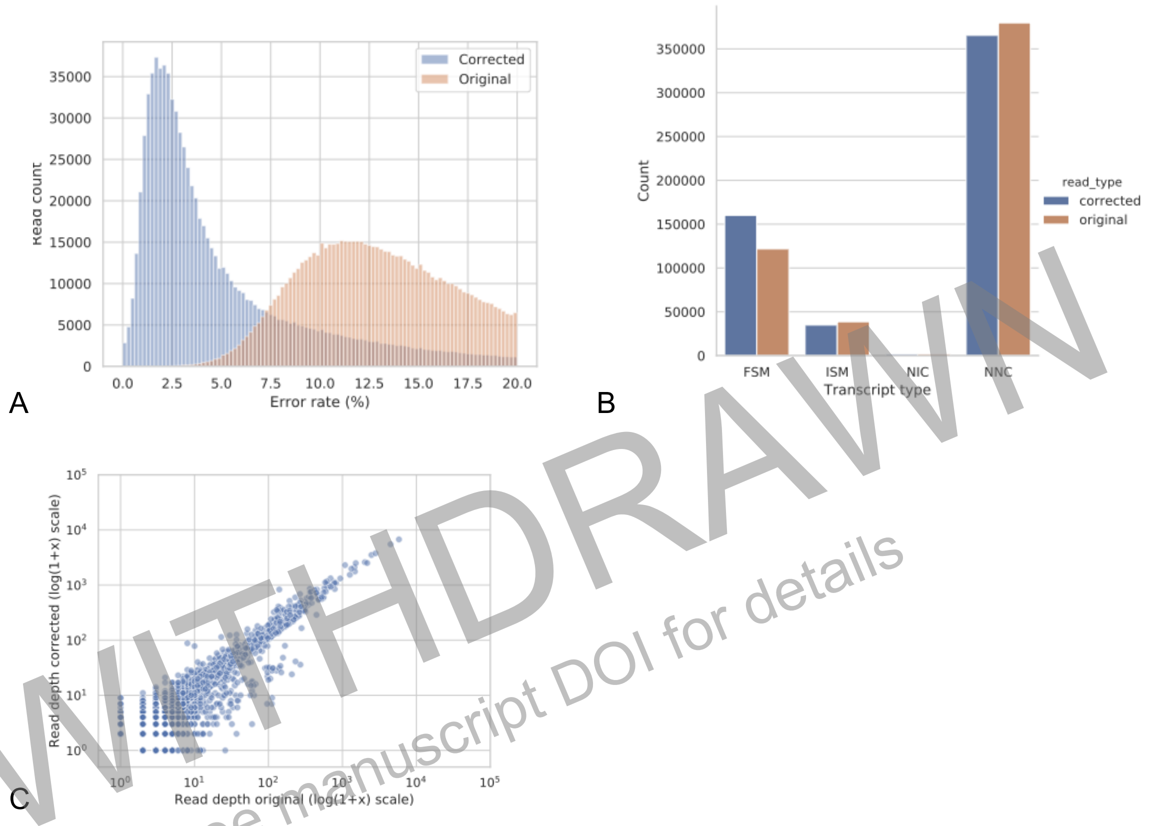

Figure S4. isONcorrect performance on the ONT-old dataset. (A) Error rates of the ONT-old reads before and after error correction by isONcorrect. (B) Total number of reads classified per splice site category. (C) For each transcript in the reference, we measure the number of reads aligning to it as a FSM, before and after error correction. Each dot represents a distinct transcript with at least one FSM in either the original or corrected reads.

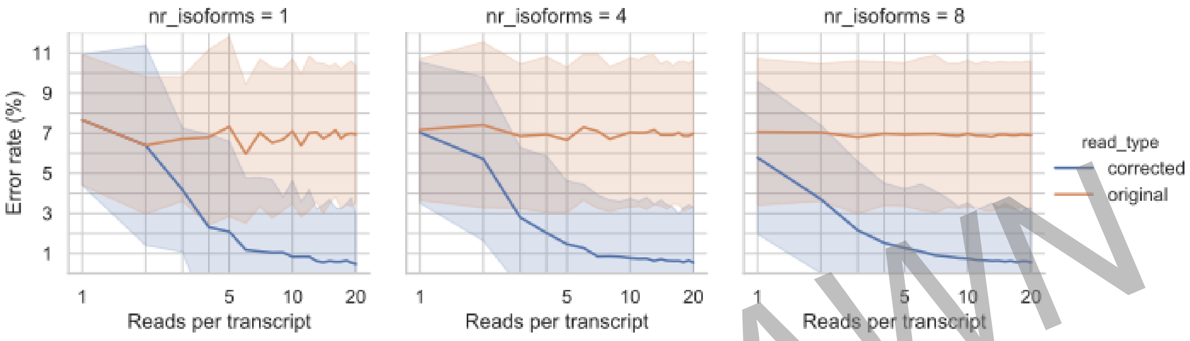

Figure S5. Effect of the number of isoforms on error rates, using subsampled SIRV datasets with varying numbers of isoforms and read depths. The shaded areas show the standard deviation of the data, aggregated over each x-value.

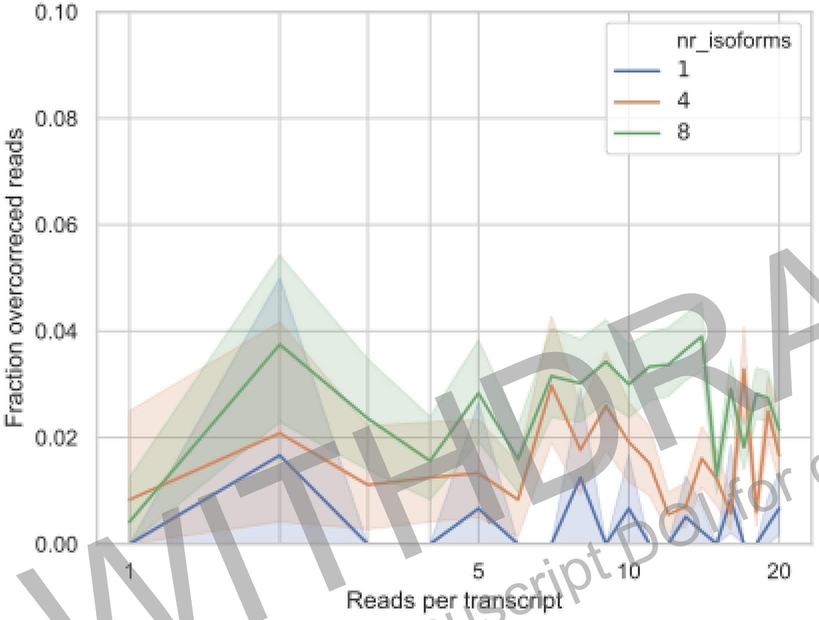

Figure S6. Effect of the number of isoforms on overcorrection, using the same subsampled SIRV datasets as in Fig. S4. The y-axis shows the percentage of overcorrected reads. The shaded areas show the standard deviation of the data, aggregated over each x-value.

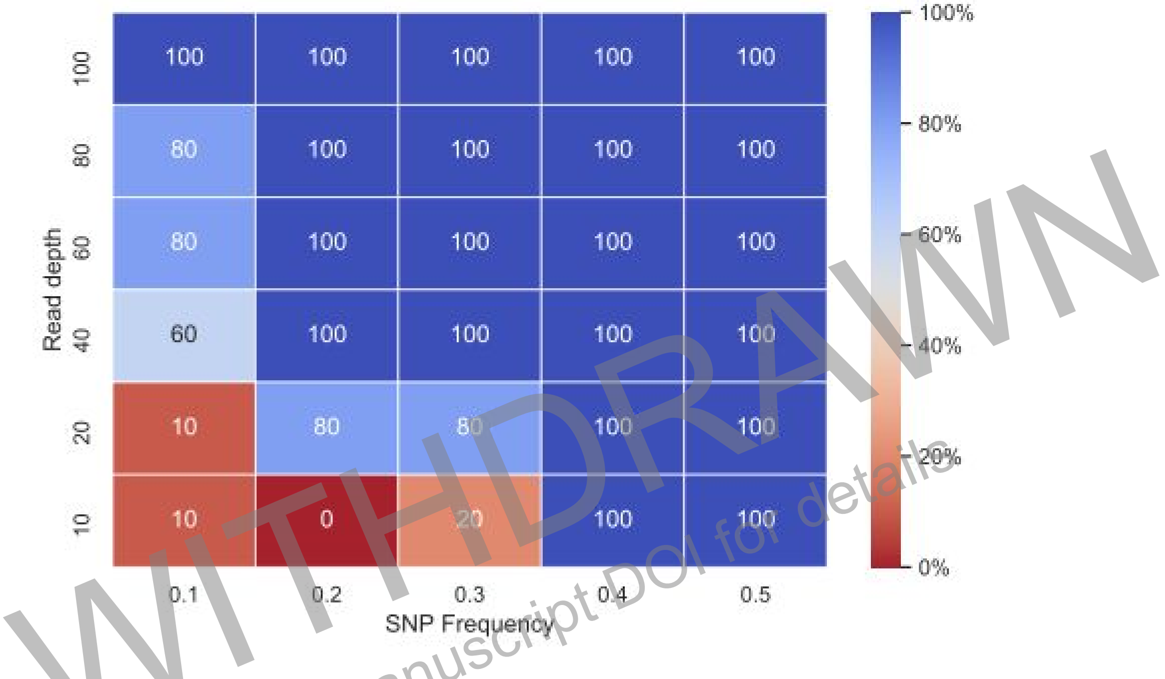

Figure S7: IsONcorrect's ability to retain a minor-allele SNP in the case of allele-specific expression. The percentage of replicate experiments where the minor allele SNP is preserved in the dataset (value in each cell) as a function of the minor allele frequency (x-axis) and the total number of reads (y-axis).

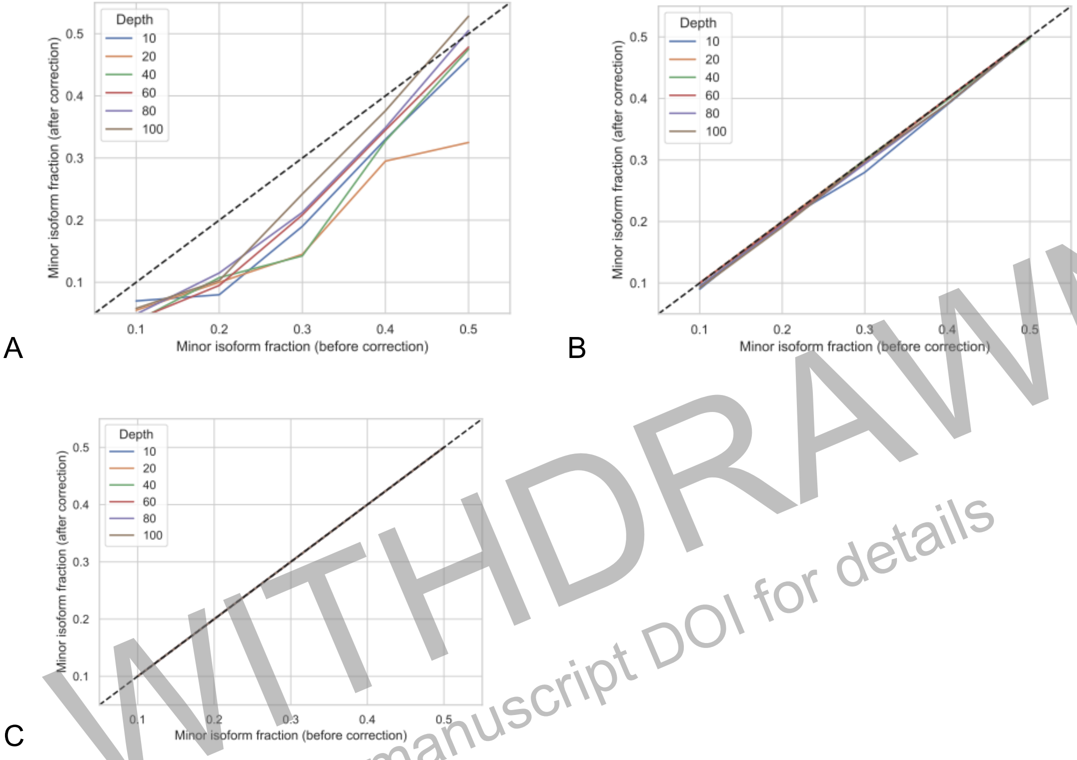

Figure S8. Perturbance of isoform abundance after error correction with isONcorrect using simulated data for a (A) 5bp exon, (B) 10bp exon, (C) 20bp exon.

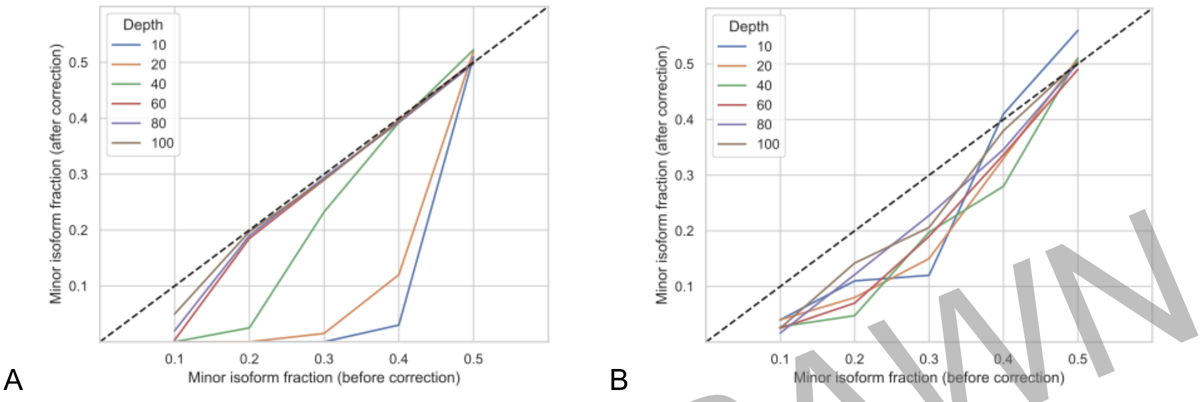

Figure S9. Perturbance of isoform abundance after error correction with isONcorrect using SIRV data for a (A) 6bp exon and (B) 14bp exon.

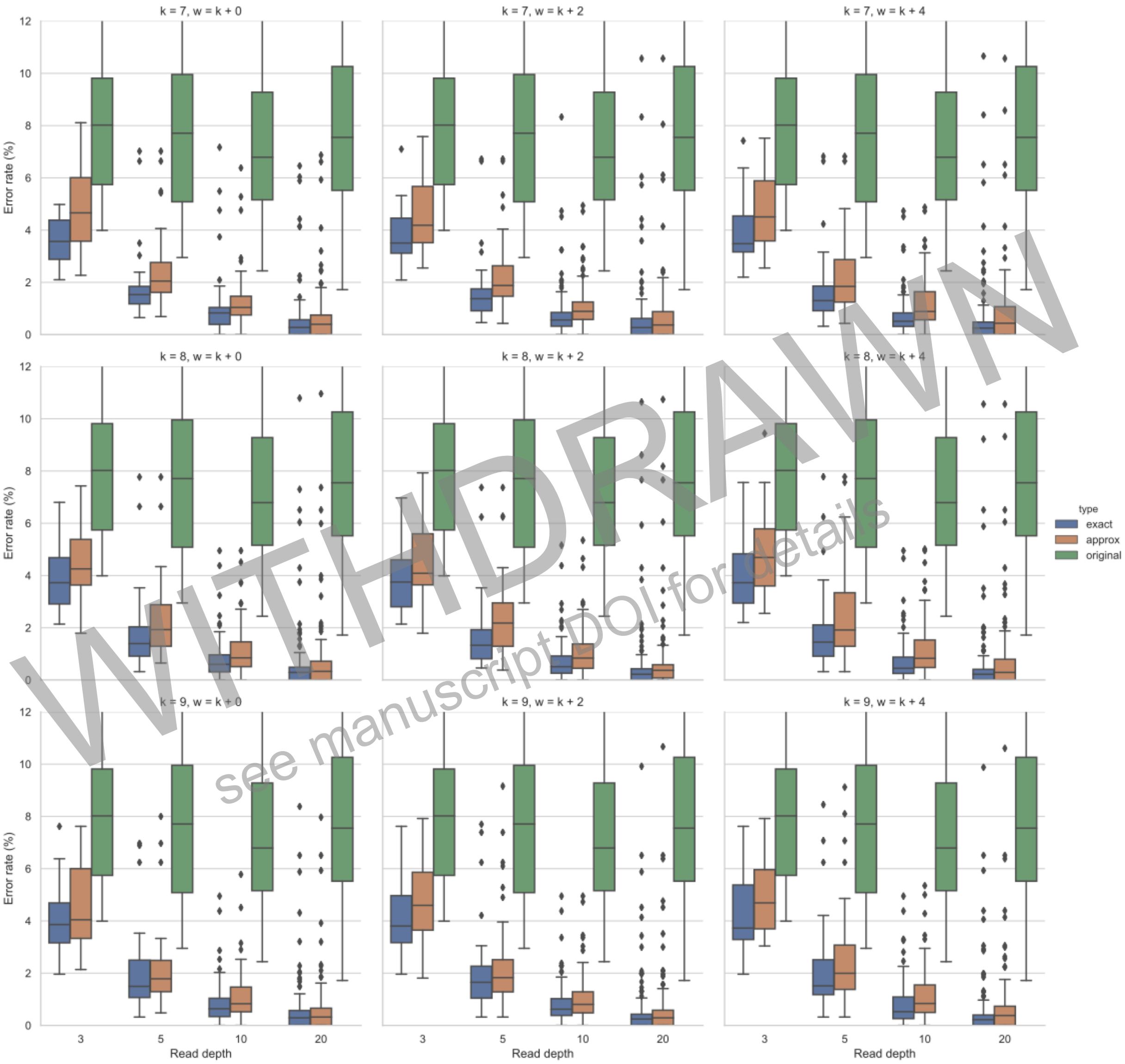

Figure S10. The effect on the error rate of parameters k and w, read depth, and the heuristic approximation algorithm. Each panel is labeled with a fixed value of k and w.

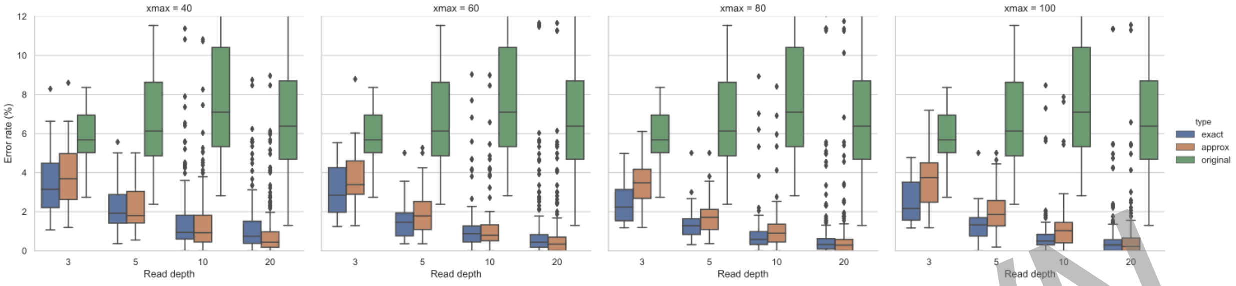

Figure S11. The effect on the error rate of the maximum anchor distance x_{max} , read depth, and the heuristic approximation algorithm. Each panel is labeled with a fixed value of x_{max} . The value of k and w is fixed to 9 in these experiments.

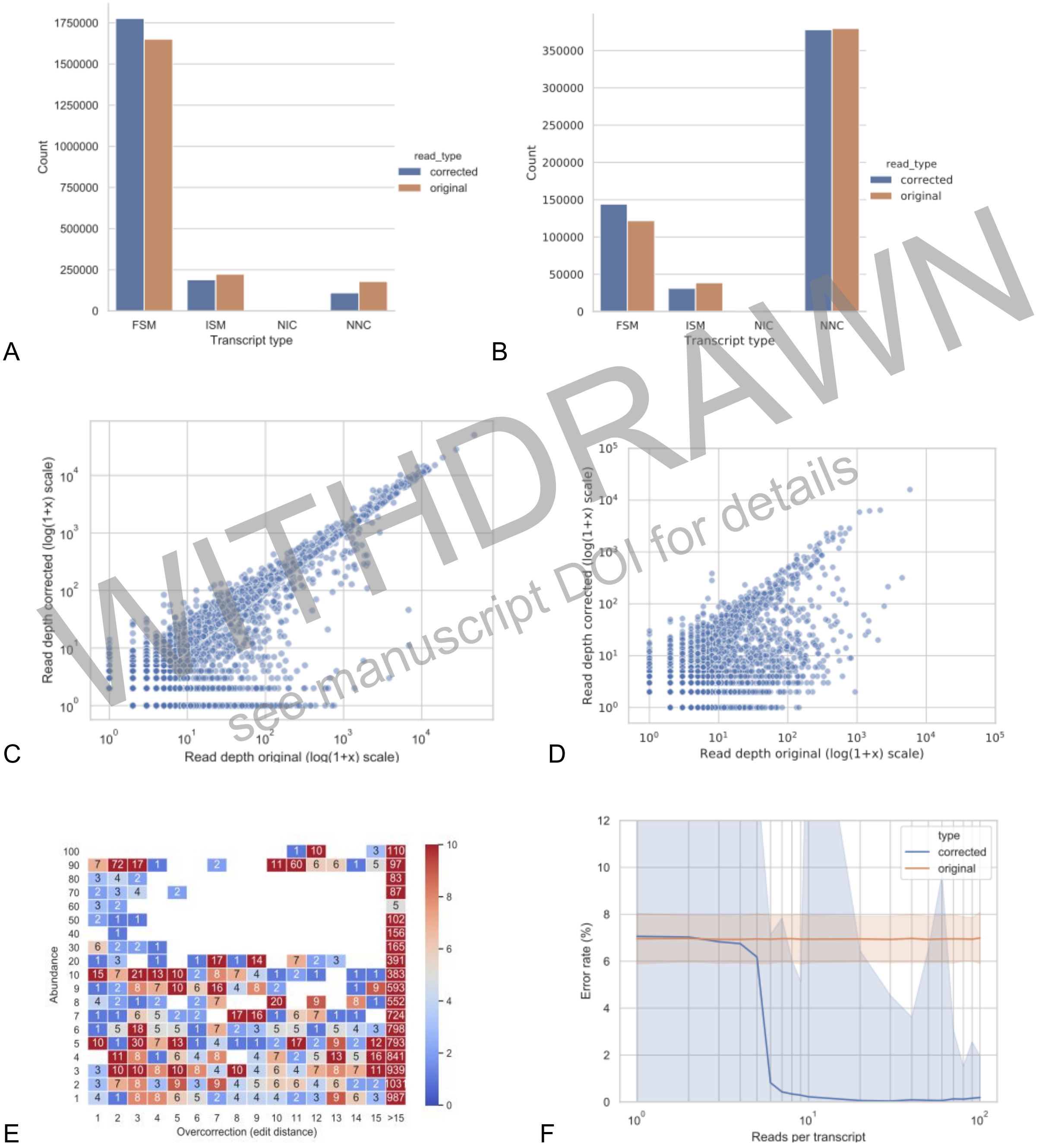

Figure S12. Splice site accuracy and overcorrection with RATTLE. Total number of reads classified per splice site category, using the terminology of (Tardaguila et al. 2018) in the Drosophila data (A) and ONT-old data (B). In (C) and (D) We measure the number of reads aligning as an FSM to a transcript in the Drosophila and human reference, respectively, before and after error correction. Each dot represents a distinct transcript with at least one FSM in either the original or corrected reads. (F) The effect of overcorrection in the SIM-ca data using RATTLE. 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. (E) The median error rate of the SIM-ca dataset based on controlled read depth of the transcript.

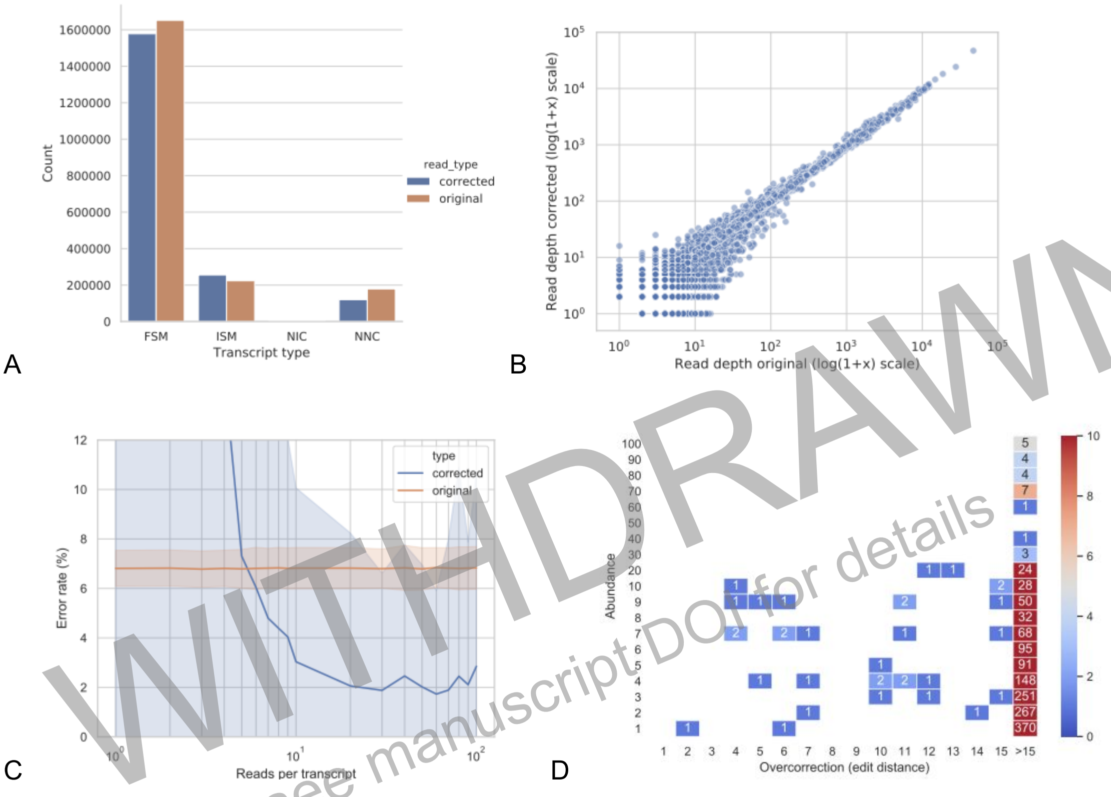

Figure S13. Canu error correction performance. (A) Total number of reads classified per splice site category on the *Drosophila* dataset. (B) For each transcript in the *Drosophila* reference, we measure the number of reads aligning to it as a FSM, before and after error correction. Each dot represents a distinct transcript with at least one FSM in either the original or corrected reads. (C) The median error rate of the SIM-ca dataset based on controlled read depth of the transcript. (D) The effect of overcorrection in the SIM-ca data using canu. 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.

Supplementary Note A: 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 $p_a = \frac{1}{a \sum_{b \in A} 1/b} \approx .32a^{-1}$.

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. We use this weighted abundance in both our simulated experiments. In the full chr6 experiment we generate 3.5M reads randomly from this relative abundance distribution. For the controlled subsampling experiments, we generate precisely a full length reads from a transcript with abundance a.

To simulate a read we pick a quality value q uniformly at random over each base pair in the transcript from a set Q of quality values. 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. In the SIM-full and SIM-ca experiments we set $Q = \{\ 0.85,\ 0.875,\ 0.9,\ 0.92,\ 0.96,\ 0.98,\ 0.99,\ 0.995\}$ to obtain a median error rate of ~7% as is seen in biological data. When we investigated error rates of ~3.94% (SIM-ca-4) and ~11.40% (SIM-ca-11), we used the sets $\{0.9,\ 0.95,\ 0.96,\ 0.98,\ 0.99,\ 0.995\}$ and $\{0.75,\ 0.85,\ 0.875,\ 0.91,\ 0.95,\ 0.98\}$, respectively. These contained 59,572 and 60,336 simulated reads, respectively.

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.

Supplementary Note B: Error rate in repetitive regions

A fraction of transcripts may come from multi-gene copies, or contain regions that are highly repetitive with respect to the transcriptome or genome because of satellite DNA, or LINE repeats. We used our human simulated dataset (SIM-full) to investigate whether correction performance was affected by repetitive regions in the transcripts. To study this we computed the average copy number of each kmer in the reference genome (Hg38) for each transcript (using k-mer size of 31) for the SIM-full dataset where we have the true annotations. We then plotted the error rates of reads after correction for the reads in SIM-full binned according to the average k-mer copy number (Figure SX). We do not observe any clear dependence of the post-correction error rate on the repetitiveness of the gene sequence.

While it is expected that difficulties in multi-mapping reads would make reference-based correction difficult, a reference-agnostic approach does not suffer from trusting genome mis-alignments. In fact, any gene with multiple isoforms will behave like a repetitive region to a transcriptomic error correction algorithm, and hence such algorithms are by necessity designed to avoid being misguided by such similarities.

Supplementary Note C: Evaluation of CONSENT and canu

We used the same machine to evaluate CONSENT and canu as we evaluated isONcorrect on. The machine was an x86_64 system running Linux v3.2.0-4-amd64 and equipped with 32 2-threaded cores and 512 GB RAM. We ran CONSENT using parameters --type ONT and --nproc 60. CONSENT returned an error on the SIRV dataset after approximately two days and on the Drosophila dataset after approximately 31 days. On the SIM-full dataset, we cancelled the job after 216 hours (9 days) without any output. These results are not surprising since CONSENT was designed for genomic, rather than transcriptomic, data.

We ran canu using parameters, -correct, useGrid=false corOutCoverage=all minReadLength=100 minOverlapLength=100 -maxThreads=60 -maxMemory=480g corOverlapper=minimap -nanopore. We also set the genomeSize parameter (which is supposed to be the estimated genome size) to the size of the transcriptome as performed in (Lima et al. 2019) and inspired by discussions at https://github.com/marbl/canu/issues/641. Specifically, we set it to 30m, 18m, and 75k for Drosophila, SIM-full, and SIRV, respectively. On the SIM-ca dataset, the average coverage per transcript is lower than the canu minimum suggested coverage (10x). As expected, canu returned an error on this dataset related to coverage and we therefore (as suggested in the output error message) reduced genomeSize to 1m.

Canu produced a post-correction error rate of 2.0%, 2.7%, and 1.9% for the SIRV, Drosophila, and the SIM-full dataset, respectively. This is significantly higher than both isONcorrect and RATTLE (Table 2). Canu also produced 266,659 (8%) reads with a higher number of errors after correction, on the SIM-full dataset, compared to only 1,578 reads for isONcorrect (Table 2). The number of reads with more errors after correction was 4,622 and 8,292 for the SIRV and Drosophila datasets, respectively, which is comparable to isONcorrect (Table 2). However, as we do not have the true annotations for the non-simulated datasets, this analysis does not include reads that have been overcorrected into another isoform.

We also studied the effect of error correction to alterations in splice structure using the *Drosophila* data. As opposed to isONcorrect and RATTLE, canu decreased the number of reads fully matching an annotated transcript (FSM) on Drosophila (Fig. S13A), and we also observed significantly less reference transcripts having at least one canu corrected FSM read (11,127) than having at least one original FSM read (13,062). This corresponds to a 14.8% reduction for Drosophila compared to only a 0.6% reduction for isONcorrect. Specifically, for Drosophila canu lost 2,023 and gained 80 transcripts with at least one FSM, while isONcorrect lost 180 and gained 100 transcripts. As for isONcorrect, mostly transcripts with few FSM reads disappeared after correction with canu (Fig. S13B).

We further investigated canu's performance on the SIM-ca dataset with variable abundances (Fig. S13C-D). The results on this dataset shows that variable coverage is a challenge for genomic error correctors that assume uniform coverage. Canu's post-correction error rate was 3.8%, which is substantially higher than isONcorrect and RATTLE (0.6% and 0.4%, respectively). The median error rate only improved for a coverage above 10 (Fig. S13C). Furthermore, canu overcorrected 1,482 reads in this dataset compared to 214 (isONcorrect) and 6,541 (RATTLE) on this dataset and these overcorrected reads were mostly substantial overcorrections (Fig. S13D) and were found throughout different abundances.

The relatively high post-correction error rates, together with substantial overcorrection and the inability to correct reads at various coverages, we conclude that canu is not suitable for transcriptomic error correction. As with CONSENT, this is not surprising since canu was not designed with this type of data in mind.

Supplementary Note D: 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).