Reference-free reconstruction and quantification of transcriptomes from Nanopore long-read sequencing

Ivan de la Rubia[1,2], Joel A. Indi[1,3], Silvia Carbonell-Sala[2,4], Julien Lagarde[2,4], M Mar Albà[2,5,6,*], Eduardo Eyras[1,5,6,7,*]

[1]EMBL Australia Partner Laboratory Network at the Australian National University, Acton ACT 2601, Canberra, Australia

[2]Pompeu Fabra University, E08003 Barcelona, Spain.

[3]Universidade de Lisboa, Lisboa, Portugal

[4]CRG, E08001 Barcelona, Spain

[5]ICREA, E08010 Barcelona, Spain

[6]IMIM, E08001 Barcelona, Spain

[7]Australian National University, Acton ACT 2601, Canberra, Australia

* co-corresponding authors: malba@imim.es, eduardo.eyras@anu.edu.au

Abstract

Single-molecule long-read sequencing with Nanopore provides an unprecedented opportunity to measure transcriptomes from any sample[1–3]. However, current analysis methods rely on the comparison with a reference genome or transcriptome[2,4,5], or the use of multiple sequencing technologies[6,7], thereby precluding cost-effective studies in species with no genome assembly available, in individuals underrepresented in the existing reference, and for the discovery of disease-specific transcripts not directly identifiable from a reference genome. Methods for DNA assembly[8–10] cannot be directly transferred to transcriptomes since their consensus sequences lack the required interpretability for genes with multiple transcript isoforms. To address these challenges, we have developed RATTLE, the first tool to perform reference-free reconstruction and quantification of transcripts from Nanopore long reads. Using simulated data, isoform spike-ins, and sequencing data from tissues and cell lines, we demonstrate that RATTLE accurately determines transcript sequence and abundance, is comparable to reference-based methods, and shows saturation in the number of predicted transcripts with increasing number of input reads.

Results

RATTLE starts by building read clusters that represent potential genes. To circumvent the quadratic complexity of an all-vs-all comparison of reads, RATTLE performs a greedy deterministic clustering using a two-step k-mer based similarity measure (Fig. 1a). The first step consists of a fast comparison of the k-mers (k=6) shared between two reads (Supp. Fig. 1a), whereas the second step is based on the Longest Increasing Subsequence (LIS) problem to find the largest list of co-linear matching k-mers between a pair of reads, which defines the RATTLE similarity score (Supp. Fig. 1b). Clusters are greedily generated by comparing reads to a representative of each existing cluster at every step of the iteration. This process generates clusters that represent potential genes with reads originating from all possible transcript isoforms.

Gene-clusters are subsequently split into sub-clusters representing candidate transcripts. These transcript-clusters are built by determining whether every pair of reads in a gene-cluster is more likely to originate from different transcript isoforms rather than from the same isoform. This is estimated from the distribution of gap-lengths between co-linear matching k-mers (Supp. Fig. 1c). RATTLE performs error correction within each one of these transcript-clusters by generating a multiple sequence alignment (MSA) (Fig. 1a). Each read is assessed for error correction taking into account the error probability for each base and the average error probability of the consensus at each MSA-column. RATTLE then builds the final transcripts after a polishing step to refine the cluster definitions. The transcript sequence is obtained from the consensus of the final transcript-cluster and the abundance is calculated as the total read count in that cluster (Fig. 1a). More details are provided in the Methods section.

To evaluate the strength of RATTLE similarity score to perform read clustering, we simulated reads from different transcripts with DeepSimulator[11], taking into account the read length distribution observed in a Nanopore cDNA sequencing run (Supp. Fig. 1d). RATTLE similarity score separates reads originating from different transcripts better than a minimizer-based score (Fig. 1b). To test the ability of RATTLE to separate reads from two transcript isoforms in a gene-cluster, we considered reads simulated from two transcripts that differ from each other only by an internal exon of 154nt, i.e. one transcript is a subsequence of the other (Supp. Fig. 1e). RATTLE approach was able to separate reads originating from the two different transcript isoforms better than using the number of common bases between reads (Fig. 1c).

To test the accuracy in the identification of gene-clusters, we compared RATTLE with two other methods to cluster long reads, CARNAC[12] and isONclust[13]. We built several reference datasets of simulated reads from multiple genes with one or more transcripts per gene and with a different number of reads per transcript. RATTLE showed higher accuracy at recovering gene-clusters in most of the comparisons and using different metrics (Fig. 1d) (Supp. Table S1). At transcript level, we only tested RATTLE, as it is the only method that predicts transcripts, and it also showed high accuracy (Supp. Table S1). Moreover, RATTLE was faster than CARNAC and isONclust in all datasets (Supp. Table S1).

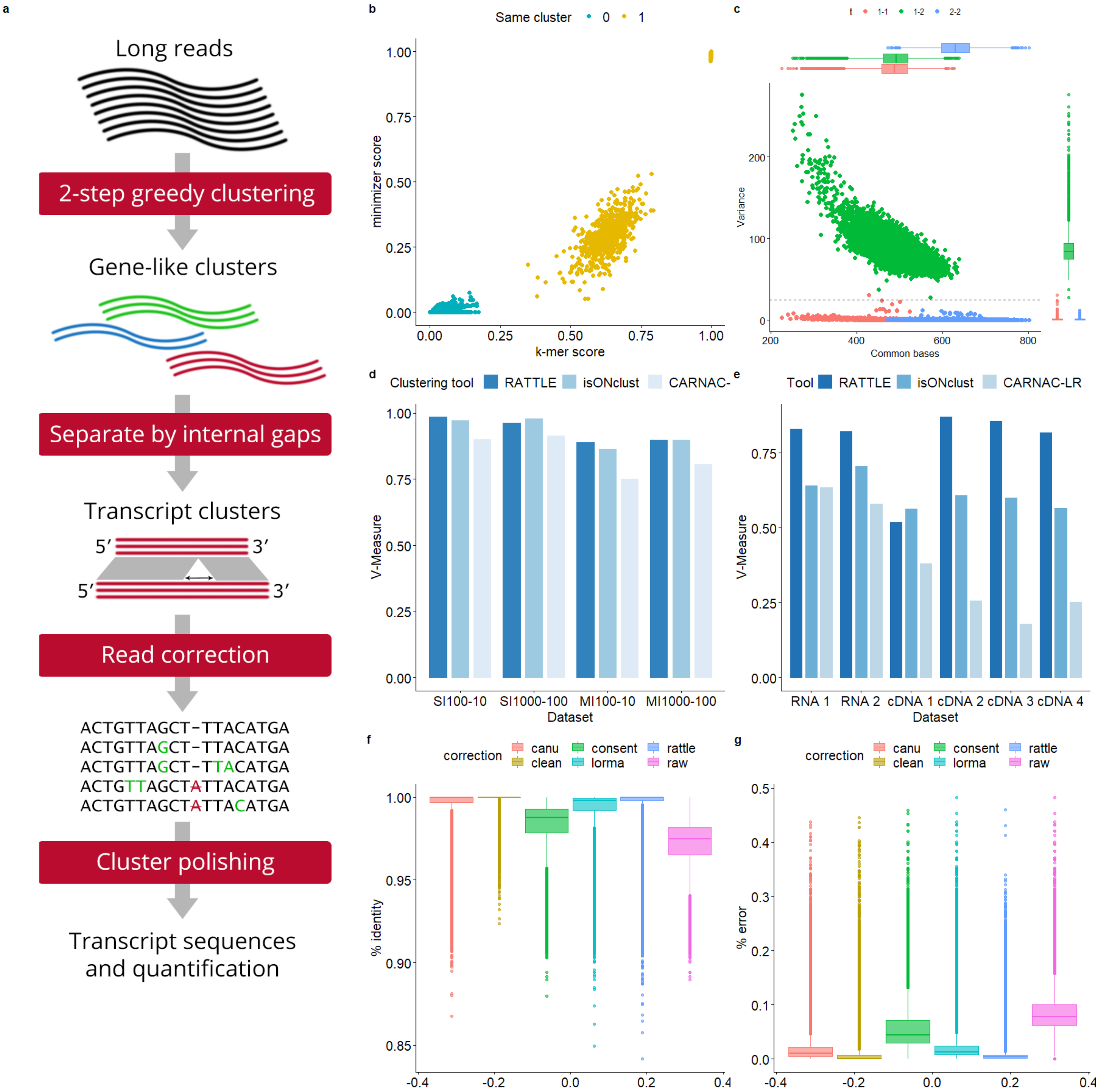

Figure 1. (a) Illustration of RATTLE workflow. **(b)** Comparison of the RATTLE similarity (k-mer) score (x axis), based on the longest increasing subsequence, and a similarity score based on minimizers (y axis), using k=6. Each dot represents a comparison between two simulated reads originating from the same (orange) or different (blue) transcripts from two different genes. **(c)** Number of common bases (x-axis) and variance in the distribution of gap-length differences between adjacent matching k-mers between two reads (y axis) from the comparison of reads simulated from two transcript isoforms from the same gene (Supp. Fig. 1e). Each dot is colored according to whether the reads originated from the same transcript (1-1, 2-2) or not (1-2). **(d)** Clustering accuracy of RATTLE, CARNAC and isONclust in terms of the V-measure (y axis), using simulated reads. Simulations (x-axis) were performed with single (SI) or multiple (MI) isoforms per gene, and using different number of transcripts per gene (*t*) and different number of reads (*r*), indicated as SI*t*-*r* or MI*t*-*r*. Other accuracy metrics are provided in Supp. Table S1. **(e)** Clustering accuracy using Spike-

in RNA Variant (SIRV) genes as reference. The plot shows the V-measure (y axis) for RATTLE, CARNAC and isONclust using SIRV reads from of the tested samples (x axis), using the direct RNA-seq protocol (RNA 1, 2) and the cDNA-seq protocol (cDNA 1, 2, 3, 4) (Methods). **(f)** Percentage identity distributions of SIRV reads before (raw) and after correcting with RATTLE, CONSENT, Lorma, Canu and TranscriptClean (clean) for Nanopore cDNA-seq data (sample cDNA 3). Percentage identity was calculated as the number of nucleotide matches divided by the total length of the aligned region. Other samples are shown in Supp. Fig. 2. **(g)** Error rate distribution of SIRV reads before (raw) and after correction with RATTLE, CONSENT, Lorma, Canu and TranscriptClean (clean) for the same sample as in (f). Error rate was calculated as the sum of insertions, deletions and substitutions divided by the length of the read. Other samples are shown in Supp. Fig. 3.

We further assessed the accuracy of RATTLE at recovering gene-clusters using Lexogen Spike-in RNA Variant Control Mixes (SIRVs). The SIRV genome (SIRVome) is organized into 7 different gene loci, each containing several transcript isoforms, conforming a total set of 69 transcripts with known coordinates, sequence and abundances. We used available MinION RNA-seq (RNA 1) and cDNA-seq (cDNA 1) samples from mouse brain that included SIRVs[14]. Additionally, we performed 4 different sequencing experiments with added SIRVs with MinION: cDNA sequencing (cDNA-seq) in human brain, two independent replicates (cDNA 2, cDNA 3), and in human heart (cDNA 4), as well as direct RNA (RNA-seq) in human heart (RNA 2). We first used the SIRV transcripts aggregated per gene to evaluate the clustering at gene-level. RATTLE showed higher accuracy in the identification of SIRV genes compared with the other two methods using various metrics (Fig. 1e) (Supp. Table S2). Furthermore, RATTLE achieved a high accuracy at transcript level using multiple metrics (Supp. Table S2).

To test RATTLE accuracy to correct errors in Nanopore reads without using a reference, we used the same SIRV reads and compared RATTLE results with CONSENT[15], Canu[8], and Lorma[9], which are self-correction methods developed for DNA long reads. Reads were mapped to the SIRVome with Minimap2[16] before and after correction by each method. As a benchmark, we included TranscriptClean[4] to correct the mapped raw reads with the SIRVome sequence but without using the SIRV annotations. After correction, all methods improved the percentage identity with the SIRV isoform sequences (Fig. 1f) (Supp. Fig. 2) and showed a decrease in the error rates (Fig. 1g) (Supp. Fig. 3). Compared with the other self-correction methods, RATTLE showed on average higher percentage identity and lower error rates, with values similar to TranscriptClean, despite not using the SIRVome sequence for correction. Notably, RATTLE was faster than the other methods in all tested samples, with runtimes 1.64-123.9 minutes (mins) (Supp. Table S3). From the other tools, TranscriptClean was the fastest for most samples with 13.06-223.00 mins, not considering the mapping of reads to the SIRVome (0.85-3.92 mins). Furthermore, RATTLE corrected approximately as many reads as Canu and TranscriptClean, and more than CONSENT and Lorma (Supp. Table S3).

We next evaluated the ability to accurately recover exon-intron structures after mapping the corrected reads to the SIRVome. We measured the exact coordinate match of the mapped reads with the SIRV annotation features: introns, intron-chains, as well as internal and external exons. RATTLE recovered similar proportion of introns as the other tools, but slightly fewer intron-chains (Fig. 2a). In contrast, RATTLE displayed higher precision values, especially for introns (Fig. 2b), suggesting that other tools produce more false positives. To further investigate this, we calculated a read-precision metric, defined as the proportion of correctly identified SIRV features over the total number of features predicted in all corrected reads, i.e. considering the number of reads supporting each predicted feature on the SIRVome. All tools, including RATTLE, showed an increase in read-precision with respect to precision (Fig. 2c), especially for intron-chains, where RATTLE had values higher than the other tools for most of the datasets tested. Similar trends were observed for internal exons (Supp. Fig. 4) (Supp. Table S4). In contrast, external exons showed lower values of recall, precision, and read-precision for all tools, indicating a general limitation to correctly recover terminal exons (Supp. Fig. 4) (Supp. Table S4). As before, RATTLE showed higher values of precision and read-precision compared with the other tools. Overall, as suggested by our analyses, the majority of false positives had low read support (Supp. Fig. 5). We decided to exploit this observation to establish the accuracy in terms of the read support. For each method, we calculated the recall, precision, and read-precision for SIRV introns at different thresholds of read support, and estimated the minimal read support needed to achieve a precision of 0.95. All correction methods showed an improvement over the raw reads (Fig. 2d) (Supp. Fig. 6). Moreover, RATTLE was the method that required the least read support and had the highest recall at 0.95 precision (Fig. 2d) (Supp. Fig. 6).

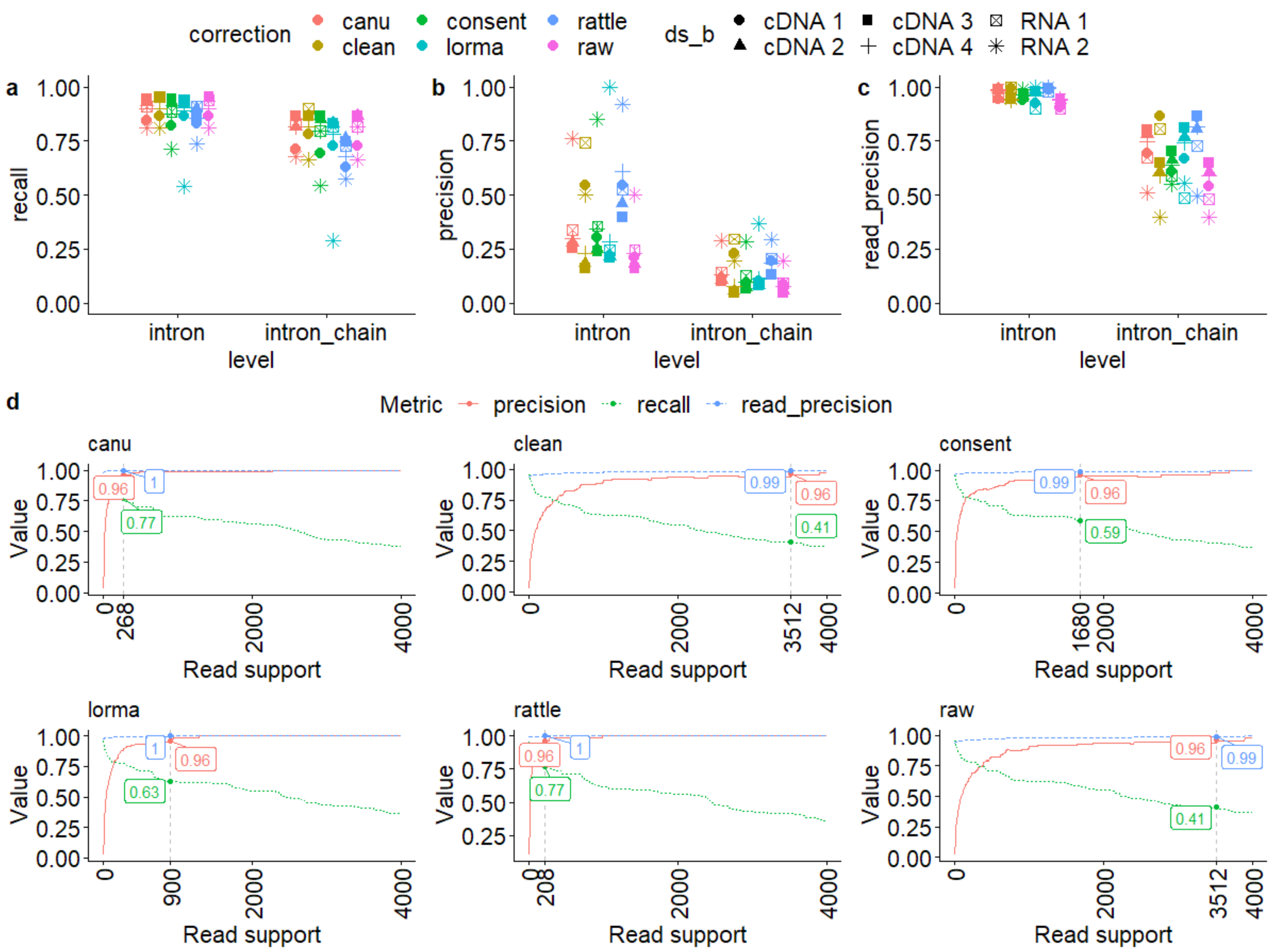

Figure 2. (a) Recall of unique SIRV introns and intron-chains obtained by mapping reads to the SIRV genome before (raw) and after correction with RATTLE, CONSENT, Canu, LORMA and TranscriptClean (clean) for the direct RNA-seq (RNA 1, 2) and the cDNA (cDNA 1, 2, 3, 4) samples. Recall was calculated as the fraction of unique annotated introns or intron-chains correctly (and exactly) found by each method with 5 or more supporting reads. **(b)** Precision values for introns and intron-chains for the same methods and datasets as (a). Precision was calculated as the fraction of unique introns or intron-chains predicted by reads that matched correctly (and exactly) the SIRV annotation with support of 5 or more reads. **(c)** Read-precision for introns and intron-chains for the same methods and datasets as in (a). Read-precision was calculated as the fraction of all introns or intron-chains predicted in reads that corresponded to SIRV introns or intron chains and had support of 5 of more reads. **(d)** We plot the recall (green), precision (red) and read-precision (blue) of the SIRV introns (y axis), as a function of the number of minimum reads supporting the predictions (x axis). We indicate for each case the threshold at which a precision (red) of approximately 0.95 is achieved. For that threshold we indicate the corresponding recall (green) and read-precision (blue). The plot corresponds to the dataset cDNA 3. Results for other samples are available in Supp. Fig. 6.

We next tested the capacity of RATTLE to estimate the abundance of SIRV transcript isoforms. Using the same datasets, we compared RATTLE with StringTie2[5], FLAIR[17], and TALON[18], which use the genome and annotation references to delineate transcript isoforms and their abundances. Additionally, we considered the approach of mapping raw reads to the SIRV isoforms with Minimap2 and then either assigned reads to SIRVs according to the best match or used NanoCount (https://github.com/a-slide/NanoCount), which assigns reads to isoforms using an expectation-maximization (EM) algorithm. For RATTLE, we assigned each SIRV to the most abundant matching predicted transcript. Despite not using any information from the SIRV genome or annotation, the correlation with SIRV isoform abundances achieved by RATTLE (Pearson R = 0.76) was comparable to those obtained with StringTie2 (R=0.73), FLAIR (R=0.79), and NanoCount (0.73), and superior to TALON (R=0.62), for RNA-seq data (RNA 1) (Fig. 3a) (Supp. Table S5). The correlation using cDNA reads was generally lower than using RNA reads for all methods and the correlation for RATTLE was generally comparable or higher than the correlation for the reference-based methods (Supp. Fig. 7) (Supp. Table S5). We next mapped the final set of transcripts predicted by RATTLE to the SIRVome and calculated the precision and recall for the recovery of annotated introns, at different thresholds of transcript abundance. For all datasets tested, RATTLE maintained a high precision (>0.75) at varying recall values (Fig. 3b) and achieved a maximum recall of over 0.75, confirming that RATTLE predictions attain high precision at different expression levels.

To establish the robustness of the transcriptome predicted by RATTLE in a cellular system, we analyzed two replicates of cDNA and RNA sequencing with MinION from a lymphoblastoid cell line (LCL)[2]. The cDNA-seq sequencing experiments contained 962,197 and 1,048,328 reads, and the RNA-seq experiments 181,465 and 139,224 reads. Due to the correction step, RATTLE only produced transcripts with a minimum read support, which we took to be 5 for the analyses presented here. RATTLE identified 8,951 and 11,468 transcripts from the cDNA-seq replicates, and 3,370 and 2,795 transcripts from the RNA-seq replicates, with >5 reads support (Supp. Table S6). The 5-mer content of the transcripts predicted by RATTLE showed a significant correlation with that of the human transcriptome annotation, for both RNA (Pearson R=0.92, p-value<2e-16) (Fig. 3c) and cDNA (Pearson R=0.95, p-value<2e-16) (Supp. Fig. 8a) reads.

We then mapped the transcripts predicted by RATTLE to the human transcriptome annotation with Minimap2 and recovered 7,309 annotated transcripts in at least one of the two cDNA-seq replicates (4,005 in common), and 3,651 in at least one of the RNA-seq replicates (1,878 in common) (Supp. Table S6). Although RNA-seq produced fewer transcripts, a higher proportion of them (>97%) mapped to the annotated transcriptome compared to cDNA-seq (89-96%) (Supp. Table S6). The transcripts predicted from both replicates showed a high correlation of the abundances estimated by RATTLE (for RNA-seq: N=1,878, Pearson R=0.91, p-value<2e-16; for cDNA-seq: N=4,005, Pearson R=0.87, p-value<2e-16) (Fig. 3d) (Supp. Fig. 8b). As a comparison, we mapped the raw reads from the same replicates to the transcript annotation and considered the transcripts with more than 5 best-matching reads. This recovered a similar number of transcripts, with 3,534 in at least one of the RNA replicates (2,197 in common), and 10,086 transcripts in at least one of the cDNA replicates (7,568 in common).

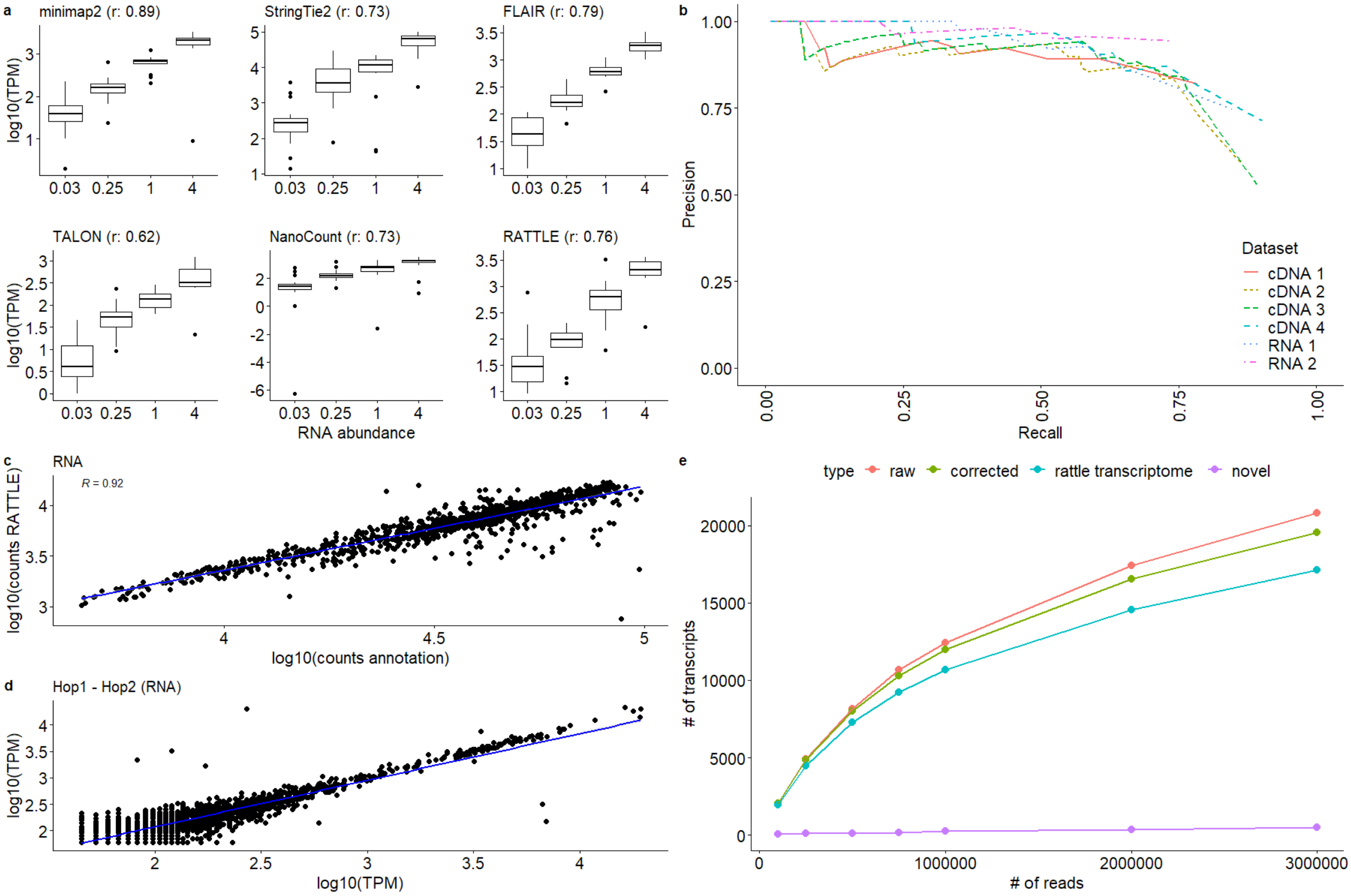

Figure 3. (a) Comparison of the transcript abundances (y axis) predicted by RATTLE, FLAIR, StringTie2, TALON, NanoCount, and selecting the best match from Minimap2 (minimap2), with abundances of the SIRV transcript isoforms (x axis). In each panel we show the Pearson correlation *r* for the comparison of the abundance values. Units on the y-axis vary according to the method: RATTLE provides abundances in terms of read counts per million, similar to TALON and FLAIR. StringTie2 produces a TPM value. For NanoCount and minimap2 we give read counts per million. SIRV data corresponds to the RNA 1 sample (Methods). Correlations for other datasets are provided in Supp. Fig. 7. **(b)** Precision-recall curve for the prediction of SIRV introns by RATTLE transcripts for all datasets tested. **(c)** Correlation of the 5-mer counts (in log10 scale) in the annotation (x axis) and in the transcript sequences predicted by RATTLE (y axis) for transcripts with read-support >5, for a direct RNA-seq run in a human lymphoblastoid cell line (LCL) from the Nanopore sequencing consortium[2] (Supp. Table S6). **(d)** Correlation of the transcript abundances calculated by RATTLE in two direct RNA-seq replicates of a human LCL from the Nanopore sequencing consortium[2] (Supp. Table S6). **(e)** Saturation plot of transcripts using direct RNA-seq reads from the 30 runs of the Nanopore sequencing consortium[2]. For each number of input reads (x axis) we give the number of transcripts identified (y axis) using the raw reads mapped to the transcriptome annotation (raw, in red), the RATTLE-corrected reads mapped to the transcriptome annotation (corrected, in green), the final RATTLE transcripts with a match to the transcript annotation (rattle transcriptome, in blue), and the

final RATTLE transcripts without any matches to the transcript annotation (novel, in purple). The number of transcripts and genes obtained are given in Supp. Table S9.

We further measured the running times and memory footprint for RATTLE with increasing number of input reads. For different configurations of parameters (Supp. Fig. 9a) (Supp. Table S7), RATTLE showed slightly above linear running time (Supp. Fig. 9b) and its memory footprint grew linearly in the clustering step and was sublinear in the correction step (Supp. Fig. 9c) (Supp. Table S8). Finally, we used data from 30 RNA-seq runs from the same LCL samples as above[2] to investigate the transcriptome generated by RATTLE with increasing number of reads. We observed that the number of transcripts and genes predicted by RATTLE tended to saturate with increasing number of input reads (Fig. 3e) (Supp. Table S9), reaching a total of 18,806 transcripts from 3M reads. The majority of those transcripts (>97%) had a match to the annotated transcriptome, and RATTLE transcripts as well as RATTLE corrected transcripts that matched the annotation followed a similar saturation trend as reads directly mapped to the annotation (Fig. 3e) (Supp. Fig. 9d) (Supp. Table S9). Moreover, the number of novel transcripts predicted by RATTLE, i.e. those without any match to the annotation, remained below 3% of the total predicted, reaching 480 for 3M input reads (Fig. 3e) (Supp. Table S9). In conclusion, RATTLE was able to effectively recover the known part of the transcriptome that is being expressed in a sample, while also being capable of identifying additional transcripts that might be cell type-specific and not be present in the annotation.

Our analyses provide compelling evidence that RATTLE accurately builds transcripts and estimate their abundances from Nanopore reads without the use of a reference genome or annotation, and without the use of additional technologies. Importantly, RATTLE achieved in general higher precision compared to other methods, which will be fundamental to reliably identify new genes and transcript isoforms in unannotated samples. Our analyses also showed that despite increasing the number of input reads, the majority of the predicted transcripts may be already known. Although our analyses indicate that error-correction impacts the ability of identifying lowly expressed molecules, it remains an essential step in long-read processing. RATTLE modularity, with the ability to parameterize its different steps, means it can be easily applicable to any sample. Additionally, RATTLE's rich output including information about the predicted transcripts and genes, and the reads used to build each transcript, will prove valuable in downstream applications, including the study of differential transcript usage[19], the identification of transcript sequence polymorphisms between individuals[20], and the analysis of single-cell long-read sequencing[21]. We anticipate that RATTLE will enable cost-effective long-read based transcriptomics in any sample and any species using only Nanopore sequencing, and in particular, in species without an available genome reference and in populations underrepresented in the current human reference.

Data availability

Nanopore sequencing data generated in this study has been deposited in the European Nucleotide Archive (ENA) under study accession PRJEB39835 (http://www.ebi.ac.uk/ena/data/view/PRJEB39835). Further datasets used in this study is available in ENA under study accession PRJEB27590 (http://www.ebi.ac.uk/ena/data/view/PRJEB27590) (runs ERR2680375 and ERR2680377)[10], and from the Nanopore sequencing consortium from https://github.com/nanopore-wgs-consortium/NA12878 (under nanopore-human-transcriptome)[2].

Software availability

RATTLE is written in C++ and is available at https://github.com/comprna/RATTLE under the GPL-3.0 license.

Methods

RATTLE clustering algorithm

Before running RATTLE, reads are pre-processed with porechop (https://github.com/rrwick/Porechop) and those of length 150nt or shorter are filtered out. RATTLE then sorts the reads in descending order by their length and processes one at a time in that order. In the first iteration, RATTLE creates a new cluster with the first unclustered read. All subsequent unclustered reads are then compared against each existing cluster and assigned greedily if the scores resulting from the comparison are above certain thresholds. Otherwise, a new cluster is created. In subsequent iterations, thresholds are decreased, and clusters are created or merged as initially. The first cluster is selected, and all other clusters are compared against that cluster, including single-read clusters, i.e. singletons. If they are similar above the set thresholds, a new cluster is formed with the reads from the selected cluster and all the similar clusters. To ensure fast computation, cluster comparisons are performed using a representative read from each cluster, which is defined by the position in the ranking of read lengths within the cluster and can be set as a parameter by the user. In our analyses, we used the read at the position 0.15 x (number of reads in the cluster).

Reads are compared using a two-step similarity calculation. This circumvents the quadratic time complexity of an all-vs-all comparison and achieves both fast and sensitive comparisons. To reduce memory usage and for efficient calculation, sequence k-mers in reads are hashed to 32-bit integers with the hashing function H(A)=0, H(C)=1, H(G)=2, H(T)=3, such that for any k-mer $s=b_1…b_k$, $H(s) = 4^{k-1}H(b_1) + 4^{k-2}H(b_2) + … + H(b_k)$. First, a similarity score is calculated as the number of common 6-mers shared between two reads divided by the maximum number of 6-mers in either read. All 6-mers are extracted for each pair of reads and a 46-bit vector is created and the positions in the vector of the hashed 6-mers in each read are set to 1. An AND operation is then performed between the two vectors to obtain the number of common 6-mers. Extraction and hashing of 6-mers is performed only once per read in linear time, and the vector operations are performed in constant time. If this first score is above a previously set threshold, a second similarity calculation is performed. For the second metric, all k-mers both reads are extracted. Now k can be chosen on the command line, and k-mers are hashed as before. The intersection of the k-mers from both reads and their positions in each read are extracted to generate a list of triplets (s, p_1, p_2), where s is the hashed k-mer, p_1 is the position of this k-mer in the first read, and p_2 is the position of the same k-mer in the second read. These triplets are then sorted by p_1 and the Longest Increasing Subsequence (LIS) problem is solved with dynamic programming for p_2. This produces the longest set of common co-linear k-mers between a pair of reads. The similarity value is defined as the number of bases covered by these co-linear common k-mers over the length of the shortest read in the pair. If the orientation for cDNA reads is unknown[22], RATTLE tests both relative orientations for each pair of reads. As a consequence, all reads within a cluster are oriented the same way.

The number of iterations to be performed for clustering is specified in the command line by setting the initial and final thresholds for the first bitvector-based score (default 0.4 to 0.2) and a decreasing step (default 0.05), i.e. default iterations are performed for thresholds 0.4, 0.35, 0.3, 0.25, and 0.2. A final comparison is done using a threshold of 0.0, i.e. all remaining singletons and all cluster representatives are compared to each other using the LIS-based score. The LIS-based score threshold remains fixed over the entire clustering process. In our analyses, it was required to be 0.2 or larger. Analyses shown here were carried out for k=10. The initial and final thresholds in the bitvector-based comparison and the decreasing step, as well as the k-mer length and similarity thresholds in the LIS comparison (bases and variance) can be set up as input parameters.

RATTLE transcript-cluster identification and error correction

Initial read clusters produced by the algorithm described above are considered to approximately correspond to genes, i.e. gene-clusters. Reads within each cluster are then separated into subclusters according to whether they are likely to originate from different transcript isoforms to form transcript-clusters. RATTLE takes into account the relative distances between co-linear k-mers calculated from the LIS-based score. Two reads in the same gene-cluster are separated into different transcript-clusters if the distribution of the relative distances between co-linear matching k-mers has a variance greater than a given threshold. That is, if co-linear matching k-mers calculated from the LIS algorithm show relative distances that would be compatible with a difference in exon content. Different thresholds were tested and the value 25 was used for the analyses. This value can also be modified as input parameter.

RATTLE performs read correction within each transcript-cluster in two steps. First, each cluster with N reads is separated into blocks; each with a number of reads R. In our analyses, R was set to 200. If R≤N<2R, the cluster is split in half, and if N<R, we take a single block. To avoid length bias, blocks are built in parallel from the reads in the cluster sorted by length: to build K blocks, block 1 will be made from reads 1, K+1, 2K+1, …, block 2 will be made from reads 2, K+2, 2K+2, … , and block K is made from reads K, 2K, 3K, … . A multiple sequence alignment (MSA) is obtained from each block using SIMD partial order alignment (SPOA) (https://github.com/rvaser/spoa)[23]. A consensus from each column in the MSA is then extracted in the following way: for each read and each base of the read, the base is changed to the consensus if the consensus base occurs with at least 60% frequency, but not if the base being assessed has an error probability (obtained from the FASTQ file) lesser than or equal to 1/3 times the average for the consensus base in that position of the alignment. Indels are treated similarly, but without the error constraint. This is only performed using aligned positions. That is, for a given base in a given read, other reads of the MSA contribute to the correction if they have in that position a base or an internal gap, i.e. we do not consider terminal gaps. The consensi from each block are then realigned with SPOA to obtain a final MSA for the transcript-cluster and an associated consensus is obtained as before. Only transcript-clusters with a minimum number of reads are corrected and considered for further analysis. Here we used transcript-clusters with more than 5 reads. The frequency of the consensus, error-probability cutoff, and minimum number of reads for a transcript cluster can be set up as input parameters. We observed that in MSAs many reads had a few bases wrongly aligned at the terminal regions. To try to fix these cases so that they do not affect the correction step, RATTLE identifies small blocks (less than 10nt) followed by large gaps (larger than or equal to 20 positions) at both ends of each aligned read and removes them. A block is defined as a subsequence that might have gaps shorter than or

equal to 3nt. RATTLE keeps removing blocks that satisfy these conditions at both ends of every aligned read until there no more such blocks are found.

RATTLE transcript polishing and quantification

To define the final list of transcripts, RATTLE performs a final polishing step of the transcript-cluster definitions. This is done to correct possible splitting into different transcript-clusters of reads originating from the same transcript (over-clustering). For this, RATTLE uses the same 2-step greedy clustering described above on the transcript-clusters using the representative read from each cluster in the comparison. In each of the resulting clusters, an MSA column consensus is calculated, with abundance given by all the reads contained in the final cluster. Additionally, the transcripts are given a gene ID that corresponds to the gene-clusters they belong to. When two transcript clusters are merged, if they were part of the same original gene-cluster, the resulting transcript stays in the same gene. If they were part of different genes, the gene with more transcripts absorbs the transcripts from the other gene to become one single gene.

RATTLE outputs different files at different stages of its execution. In the clustering step, it can either output gene-clusters or transcript-clusters in binary files. These files can be then used to extract a CSV file containing each read ID and the cluster it belongs to. These same binary files are also used for the correction step. This step outputs three files, one file with the corrected reads, one with those that are left uncorrected, and one containing the consensus sequence for each cluster from the input (in FASTQ format). Finally, the transcript-cluster polishing step receives as input the consensus sequences from the correction step and outputs a new file in FASTQ format with the final transcriptome. The quantification of each transcript is included in the header line. The quality per base in each FASTQ entry is calculated as the average of the PHRED quality scores from the bases in each column of the MSA in the transcript-cluster.

Simulated reads and clusters

We developed a wrapper script (available at https://github.com/comprna/RATTLE) for DeepSimulator[11] to simulate a specific number of reads per transcript considering a read length distribution. The length distribution was calculated from a human cDNA sequencing sample from the Nanopore consortium[2]. To simulate the read sequences, we used the Gencode transcript annotation (v29), filtering out pseudogenes and genes from non-standard chromosomes, and removing transcripts that showed > 95% percentage identity with other transcripts using CDHIT[24]. We then randomly selected different number of genes and transcripts to simulate reads. We considered genes with one single transcript isoform (SI), or genes with multiple isoforms (MI). For each case, various datasets were simulated using different number of reads per transcript and different number of transcripts per gene. To determine the accuracy of the clustering we used the adjusted rand index, which is a measure of the similarity between two cluster sets corrected for chance[25]. Additionally, we used homogeneity, completeness and the V-measure[26]. Homogeneity is maximal when each cluster contains only elements of the same class. Completeness is maximal when all the elements of a single class are in the same cluster. The V-measure is the harmonic mean of the completeness and homogeneity. We compared the clusters predicted by each method with the simulated clusters as reference set. We run isONclust[13] (options: --ont --t 24), CARNAC-LR[12] with Minimap2 overlaps (options: -t 24 -x ava-ont), and RATTLE clustering (options: -t 24 -k 10 -s 0.20 –v 1000000 –iso-score-threshold 0.30 –iso-kmer-size 11 –iso-max-variance 25 –p 0.15).

MinION sequencing and SIRV reads

We used data from cDNA (ERR2680377) (cDNA 1) and direct RNA (ERR2680375) (RNA 1) sequencing of mouse brain including the E2 SIRVs[14]. Additionally, we performed Nanopore sequencing on two total RNA samples from human brain (Ambion - product num. AM7962; lot num. 1887911) and heart **(**Ambion - product num. AM7966; lot num. 1866106). Unless otherwise noted, kit-based protocols described below followed the manufacturer's instructions. Regular quality controls using qBIT, nanodrop and Bioanalyzer were performed according to manufacturer's protocols to assess the length and the concentration of the samples. rRNA depletion was performed using Ribo-Zero rRNA Removal Kit Human/Mouse/Rat (Epicentre - Illumina). 12 ug of total RNA from each sample were prepared and divided into 3 aliquots (4 ug of total RNA each). 8ul of a 1:100 dilution (1 ng total) of synthetic controls (E2 mix lot number 001418 from SIRV-set, Lexogen) were added to each total RNA aliquot. Resulting ribosomal depleted RNAs were purified using 1.8X Agencourt RNAClean XP beads (Beckman Coulter). Samples were finally resuspended with 11 ul of RNA-free water and stored at -80ºC. The cDNA was prepared using 50 ng of rRNA depleted RNA. The cDNA synthesis kit (Takara) based on SMART (Switching Mechanism at 5' End of RNA Template) technology coupled with PCR amplification was used to generate high yields of full-length double-stranded cDNA. The sequencing libraries were prepared using 1 ug of full-length double-stranded cDNA following the standard ONT protocol SQK-LSK108 for an aliquot of the brain sample (cDNA 2), protocol SQK-LSK109 for 1 aliquot of the brain sample (cDNA 3) and 1 aliquot of the heart sample (cDNA 4). The direct RNA sequencing library was prepared using 500 ng of previously prepared ribosomal depleted sample (RiboZero kit, catalog num. MRZH11124, Epicentre-Illumina) from heart (RNA 2) with standard direct RNA ONT protocol SQK-

RNA002, following manufacturer's instructions. The final libraries were loaded on an R9.4.1 flowcell, and standard ONT scripts available in MinKNOW were used for a total of 48 hours run for each flowcell. ONT sequencing data was basecalled using Guppy 2.3.1+9514fbc (options: --qscore_filtering --min_qscore 8 --flowcell FLO-MIN106 --kit <kit> --records_per_fastq 0 --recursive --cpu_threads_per_caller 14 --num_callers 1), where <kit> is the corresponding protocol, as specified above (SQK-LSK109, SQK-LSK108 or SQK-RNA002). To select reads corresponding to SIRVs, we run porechop (https://github.com/rrwick/Porechop) and mapped the reads to the SIRV genome (SIRVome) with Minimap2 (options: -t 24 -cx splice --splice-flank=no --secondary=no). We used seqtk (https://github.com/lh3/seqtk, option subseq) to extract the subset of reads with a hit on the SIRVome and being at least 150nt in length. These reads were then considered for further analyses.

Clustering accuracy with SIRV reads

We first built SIRV isoform clusters by mapping reads to SIRV isoforms with Minimap2[16] and selecting for each read the SIRV isoform with the best mapping score. All reads that mapped to the same SIRV gene were then considered a gene-cluster. We then clustered reads with RATTLE, CARNAC and isONclust and measured the accuracy of the predicted clusters by comparing with the built SIRV gene clusters using the same metrics as with the simulated data. We used subsets of 25,000 reads (subsampled with seqtk) from each sample, since we could not make CARNAC run for some of the complete datasets.

Assessment of error correction accuracy

Reads were mapped to the SIRV transcripts with Minimap2 before and after read correction. Here we used the complete dataset of SIRV reads. Each read was assigned to the best matching transcript according to the mapping score. From the CIGAR string of the SAM output the error rate was calculated as the sum of insertions, deletions and substitutions divided by the length of the read, and the percentage identity as the number of nucleotide matches divided by the total length of the aligned region. We compared RATTLE (options: -t24 –g 0.3 –m 0.3 –s 200) with CONSENT[15] (options: consent-correct --type ONT), Canu[8] (options: minReadLength=200 stopOnLowCoverage=0.10 executiveMemory=16 executiveThreads=24), LORMA[9] (options: -s -n -start 19 -end 61 -step 21 -threads 24 -friends 7 -k 19), and TranscriptClean[4]. TranscriptClean was run using as input the reads mapped with Minimap2 (options: -t 12 -cx splice --splice-flank=no --secondary=no), but with no exon-intron information from the SIRV annotation.

To perform the accuracy analysis of the SIRV annotation features, PAF files from the mapping were compared with the annotation using ssCheck (available at https://github.com/comprna/RATTLE/). We developed ssCheck, as gffcompare (https://ccb.jhu.edu/software/stringtie/gffcompare.shtml) did not count correctly the matches when multiple copies of the same annotation feature were present in the reads. ssCheck works by comparing annotation features in the mapped reads with the annotation and calculates the number of unique features as well as the total number of features predicted in the mapped reads. As annotation features, we used introns, intron-chains. internal exons and external exons. An intron-chain was defined as an ordered sequence of introns in an annotated transcript or mapped read. Recall was calculated as the fraction of unique annotated features correctly found; precision was calculated as the fraction of unique predicted features that were in the annotation and read-precision was calculated as the fraction of the total number of predicted features in reads that corresponded to annotated features. Read-precision is influenced by abundance levels but better reflects the accuracy per read.

Assessment of transcript quantification accuracy

We used FLAIR[17] (options: align, correct, collapse, quantify –tpm), StringTie2[5] (options: -p 24 –L) and TALON[18] (talon_initialize_database, talon, talon_summarize, talon_abundance, talon_create_GTF) with the cDNA and RNA reads mapped to the SIRV genome with Minimap2 (options: -t 24 –ax splice –splice-flank=no –secondary=no, with –MD tag for TALON). These methods perform read correction (FLAIR and TALON) and transcript quantification (FLAIR, StringTie2 and TALON) of annotated and novel transcripts using the mapped reads with the help of the annotation. For the same samples, RATTLE was run for clustering (options: -t 24, -k 10, -s 0.20 –v 1000000 –iso-score-threshold 0.30 –iso-kmer-size 11 –iso-max-variance 25 –p 0.15), read correction (options: -t24 –g 0.3 –m 0.3 –s 200) and transcript polishing (options: -t24). As additional comparison, we mapped raw reads directly to SIRV isoforms with Minimap2 (options: -ax map-ont -t24) and estimated abundances with NanoCount (https://github.com/a-slide/NanoCount), which assigns reads to isoforms with an expectation-maximization (EM) algorithm. We also assigned reads directly to SIRV isoforms with Minimap2 (options: -t 24 -cx map-ont --secondary=no). FLAIR, StringTie2 and TALON provides the SIRV isoform ID with the predicted abundance. Sometimes, these methods give twice the same ID with two different abundances and exon-intron structures, likely due to both being equally good approximate matches to the annotation. In these cases, we only considered the prediction with the highest abundance. To assess the accuracy of RATTLE, we matched transcripts predicted by RATTLE to the SIRV isoforms using Minimap2 (options: -cx map-ont --secondary=no). If more than one transcript matched the same SIRV isoform, we selected the RATTLE transcript with the highest abundance.

Analysis of human transcriptomes

We downloaded data from cDNA and RNA sequencing from the Nanopore consortium[2]. For the samples from Johns Hopkins (cDNA 2 replicates, RNA 2 replicates) and UCSC (cDNA 2 replicates, RNA 2 replicates), we predicted the transcripts and their abundances with RATTLE for each sample independently. To calculate the correlation of RATTLE abundances between replicates we mapped RATTLE transcript sequences to the transcripts from the annotation (Gencode v29, after removing pseudogenes, genes from non-standard chromosomes, and transcripts with > 95% percentage identity with other transcripts). Abundances of predicted transcripts mapped to the same annotated transcript were then compared. For comparison, raw reads were mapped to the same annotated transcripts with Minimap2 (options: -cx map-ont --secondary=no). As we built RATTLE transcripts from transcript-clusters with >5 reads, only transcripts in the annotation with >5 reads mapped were considered. The 5-mer content was calculated as the frequency of each 5-mer in all the transcripts predicted by RATTLE and in all annotated transcripts with >5 reads mapped to them.

Time, memory footprint, and saturation analysis

We took the pooled 30 direct RNA runs from the Nanopore sequencing consortium[2] (https://github.com/nanopore-wgs-consortium/NA12878/blob/master/RNA.md) and generated subsamples with 100k, 250k, 500k, 750k, 1M, 2M, and 3M reads. Subsampling was performed with seqtk (https://github.com/lh3/seqtk). RATTLE was run in all these samples (options: -B 0.5 -b 0.3 -f 0.2) (configuration c5 in Supp. Table S7) using a machine with 24 cores. In each run 64Gb RAM were requested although RATTLE only used 20-30Gb of RAM.

Acknowledgements

We thank Roderic Guigó for providing access to the samples with SIRVs. We also thank the Nanopore sequencing consortium for providing early access to their Nanopore transcriptomic sequencing runs. We acknowledge funding support from the Spanish Government and FEDER with grants BFU2015-65235-P, BIO2017-85364-R, PGC2018-094091-B-I00, and MDM-2014-0370; from the Catalan Government (AGAUR) with grant SGR2017-1020; and Spanish National Bioinformatics Institute (INB), funded by ISCIII and FEDER (PRB2-ISCIII; Grant PT17/0009/0014). IdlR was supported by grant PRE2018-083413 from the Spanish Government (MINECO).

Author contributions

EE and IdlR designed the algorithms in RATTLE with inputs from MMA. IdlR prototyped and implemented the algorithms. IdlR and JI carried out the benchmarking analyses. SCS and JL generated the experimental data. EE, IdlR and MMA wrote the paper with inputs from all authors.

Potential competing interests

Eduardo Eyras has received support from Oxford Nanopore Technologies (ONT) to present the results from this manuscript at scientific conferences. However, ONT played no role in the algorithm or software developments, study design, analysis, or preparation of the manuscript.

References

1. Garalde, D. R. *et al.* Highly parallel direct RNA sequencing on an array of nanopores. *Nat. Methods* **15**, 201–206 (2018).
2. Workman, R. E. *et al.* Nanopore native RNA sequencing of a human poly(A) transcriptome. *Nat. Methods* **16**, 1297–1305 (2019).
3. Eid, J. *et al.* Real-time DNA sequencing from single polymerase molecules. *Science* **323**, 133–8 (2009).
4. Wyman, D. & Mortazavi, A. TranscriptClean: variant-aware correction of indels, mismatches and splice junctions in long-read transcripts. *Bioinformatics* **35**, 340–342 (2019).
5. Kovaka, S. *et al.* Transcriptome assembly from long-read RNA-seq alignments with StringTie2. *Genome Biol.* **20**, 278 (2019).
6. Koren, S. *et al.* Hybrid error correction and de novo assembly of single-molecule sequencing reads. *Nat. Biotechnol.* **30**, 693–700 (2012).
7. Fu, S. *et al.* IDP-denovo: de novo transcriptome assembly and isoform annotation by hybrid sequencing. *Bioinformatics* **34**, 2168–2176 (2018).
8. Koren, S. *et al.* Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *Genome Res.* **27**, 722–736 (2017).

9. Salmela, L., Walve, R., Rivals, E. & Ukkonen, E. Accurate self-correction of errors in long reads using de Bruijn graphs. *Bioinformatics* **33**, 799–806 (2017).
10. Xiao, C.-L. *et al.* MECAT: fast mapping, error correction, and de novo assembly for single-molecule sequencing reads. *Nat. Methods* **14**, 1072–1074 (2017).
11. Li, Y. *et al.* DeepSimulator: a deep simulator for Nanopore sequencing. *Bioinformatics* **34**, 2899–2908 (2018).
12. Marchet, C. *et al.* De novo clustering of long reads by gene from transcriptomics data. *Nucleic Acids Res.* (2018). doi:10.1093/nar/gky834
13. Sahlin, K. & Medvedev, P. De novo clustering of long-read transcriptome data using a greedy, quality-value based algorithm. in *International Conference on Research in Computational Molecular Biology* 227–242 (Springer, 2019).
14. Sessegolo, C. *et al.* Transcriptome profiling of mouse samples using nanopore sequencing of cDNA and RNA molecules. *Sci. Rep.* **9**, 14908 (2019).
15. Morisse, P., Marchet, C., Limasset, A., Lecroq, T. & Lefebvre, A. CONSENT: Scalable self-correction of long reads with multiple sequence alignment. *bioRxiv* 546630 (2019). doi:10.1101/546630
16. Li, H. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics* **34**, 3094–3100 (2018).
17. Tang, A. D. *et al.* Full-length transcript characterization of SF3B1 mutation in chronic lymphocytic leukemia reveals downregulation of retained introns. *Nat. Commun.* **11**, 1438 (2020).
18. Wyman, D. *et al.* A technology-agnostic long-read analysis pipeline for transcriptome discovery and quantification. *bioRxiv* 672931 (2019). doi:10.1101/672931
19. Trincado, J. L. *et al.* SUPPA2: Fast, accurate, and uncertainty-aware differential splicing analysis across multiple conditions. *Genome Biol.* **19**, (2018).
20. Rivas, M. A. *et al.* Human genomics. Effect of predicted protein-truncating genetic variants on the human transcriptome. *Science* **348**, 666–9 (2015).
21. Singh, M. *et al.* High-throughput targeted long-read single cell sequencing reveals the clonal and transcriptional landscape of lymphocytes. *Nat. Commun.* **10**, 3120 (2019).
22. Ruiz-Reche, A., Srivastava, A., Indi, J. A., de la Rubia, I. & Eyras, E. ReorientExpress: reference-free orientation of nanopore cDNA reads with deep learning. *Genome Biol.* **20**, 260 (2019).
23. Lee, C., Grasso, C. & Sharlow, M. F. Multiple sequence alignment using partial order graphs. *Bioinformatics* **18**, 452–64 (2002).
24. Fu, L., Niu, B., Zhu, Z., Wu, S. & Li, W. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics* **28**, 3150–2 (2012).
25. Vinh, N. X., Epps, J. & Bailey, J. Information theoretic measures for clusterings comparison: Variants,

properties, normalization and correction for chance. *J. Mach. Learn. Res.* **11**, 2837–2854 (2010).

26. Rosenberg, A. & Hirschberg, J. V-measure: A conditional entropy-based external cluster evaluation measure. in *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)* 410–420 (2007).