Poly(A)-Tail Length Estimation from long ONT sequencing: Motivation and Tool Comparison

Remy Schwab[1,2] and Luchao Qi[1,2]

[1]Department of Biomedical Engineering, Johns Hopkins University Whiting School of Engineering

[2]Team RemChao, Johns Hopkins University

Abstract

Motivation: Polyadenylation is a co-transcriptional process in which an adenosine monophosphate molecule is added to the 3' end of mRNA. These molecules, which consist solely of ade-nines, are known as Poly(A) tails. On mRNAs, the poly(A) tail protects the mRNA molecule from enzymatic degradation in the cytoplasm and aids in transcription termination, export of the mRNA from the nucleus, and translation. Also, poly(A) lengths (the number of bases that make up the molecule) are believed to have some important biological implications. Along with the advent of native RNA sequencing, two tools, nanopolish and tailfindr, have proposed methods for estimating the lengths of Poly(A) tails using the Oxford Nanopore platform. Our project aims to compare these tools.

Results: The tool tailfindr can work on single or multi-fast5 file reads and support data that has been basecalled with Albacore or Guppy. It also supports data that has been basecalled using the newer 'flipflop' model. However, it requires some metadata table and it leads to incompatibility with tailfindr. We observed higher estimation from nanopolish which can be attributed to differences in normalization.

Contact: rschwab6@jhmi.edu, lqi9@jhu.edu

1 Introduction

Polyadenylation is the addition of a poly(A) tail to a messenger RNA. The poly(A) tail consists of multiple adenosine monophosphates; in other words, it is a stretch of RNA that has only adenine bases. In eukaryotes, polyadenylation is part of the process that produces mature messenger RNA (mRNA) for translation[1]. It, therefore, plays a critical role in RNA metabolism including stability, enhanced translation, nuclear export and miRNA mediated gene regulation[2]. The process of polyadenylation begins as the transcription of a gene terminates. The 3'-most segment of the newly made pre-mRNA is first cleaved off by a set of proteins; these proteins then synthesize the poly(A) tail at the RNA's 3' end.

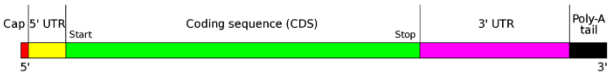

Figure 1: The structure of typical human protein coding mRNA including untranslated regions (UTRs)

Poly(A) tails are necessary for translocation of mRNA to the cytoplasm and directly influence a transcript's stability and translation. While the importance of poly(A) tails is somewhat understood, what effect the length of the poly(A) tail has is not[3]. The initial poly(A) tail length after polyadenylation is reported to be about 250nt however this becomes dynamic after it has left the nucleus and the data so far is difficult to unify. For example, even though only a short poly(A) tail is needed to prevent degradation, hyper-adenylated transcripts degrade quickly in the nuclease. Additionally, there has been evidence showing that long tails during embryonic development are positively correlated with translational efficiency but in *C. elegans* shorter tails are shown to be more actively translated[4].

Early attempts to measure poly(A) tail length were low-throughput and were transcript-specific. High throughput methods enabled transcriptome wide resolution but were severely limited by short read lengths and technical artifacts related to PCR. It is becoming increasingly clear that while the technologies of today may be capable of providing population-level sequencing to both researchers and clinicians, key limitations remain[5]. From a technological perspective, accuracy and coverage across the genome are still problematic, particularly for GC-rich regions and long homopolymer stretches. In addition, the short reads lengths produced by most current platforms severely limit our ability to accurately characterize large repeat regions, many indels and structural variation, leaving significant portions of the genome opaque or inaccurate. Given the limitations and biases of different platforms, it is also likely that accurate genome sequencing will use a combination of technologies.

To solve this problem, the Oxford Nanopore platform is able to sequence mRNA molecules. A protein nanopore is set in an electrically resistant polymer membrane. An ionic current is passed through the nanopore by setting a voltage across this membrane. If an analyte passes through the pore or near its aperture, this event creates a characteristic disruption in current. Measurement of that current makes it possible to identify the molecule in question. The long reads lengths and library preparation protocol allow for sequencing of the full transcript[6].

Our dataset was downloaded from European Nucleotide Archive (ENA), study accession: PRJEB28423. The aim of this study was to generate data from synthetic RNA samples with approximately known poly(A) tail lengths using the Oxford Nanopore MinION.RNA standard transcripts of differing known poly(A) tail lengths (10X-100X poly(A))

DNA templates for in vitro RNA transcription Linear DNA templates for RNA transcription were made through a nested PCR protocol.

However, it is very difficult to sequence repetitive regions with Oxford Nanopore and so two tools (nanopolish and tailfindr) have attempted to reconcile this. Our results show that both tools have very interpretable results. Overall, performance seems pretty similar while both tools agree in the majority of cases in the definition of poly(A) segments, we routinely observed slightly higher estimates from Nanopolish which can be attributed to differences in normalization[7].

2 Methods

Based on results from nanopore sequencing, estimation of length of poly(A) tails were performed on fast5 files. Oxford Nanopore Technologies (ONT) Sequencing allows for the sequencing of full-length native RNA molecules containing the entire poly(A) tail by ligation of a double-stranded DNA adapter to the 3'-end of each RNA molecule[8].

Figure 2: Schematic representation of Oxford Nanopore RNA Sequencing. The Motor protein (red) is attached to the native RNA molecule (yellow) at the 3'-end by T4 DNA ligation via a double-stranded adapter (light red) with oligo-T overhang. The motor protein thus feeds the RNA strand to the pore from 3' to 5'[7].

Here, two software packages (Nanopolish and Tailfindr) were used to help perform the analysis.

Nanopolish is a suite of algorithms for analyzing Oxford Nanopore Sequencing data. It can calculate an improved consensus sequence for a draft genome assembly, detect base modifications (m6a, m5c), call SNPs and indels with respect to a reference genome and more. Also, it has a component, nanopolish polya, to estimate poly(A) tail lengths.

This method combines a hidden markov model (HMM) with an estimator of the translocation rate of the read through the pore. The foundation of the algorithm involves separating the raw "squiggle" into states which correspond to the states that make up the HMM. These states are denoted as follows:

- Start (S): an optional state appearing before the "leader"
- Leader (L): the sequencing adaptor, attached to, and sequenced prior to the RT splint adaptor
- Adaptor (A): RT splint adaptor which is directly attached to the poly(A) tail
- PolyA (P): the polyadenylated region of the read
- Cliff (C): a state to model brief sequencing artifacts within the poly(A) region of the read
- Transcript (T): The coding sequence

Below is a breakdown into these states from an example squiggle[1]:

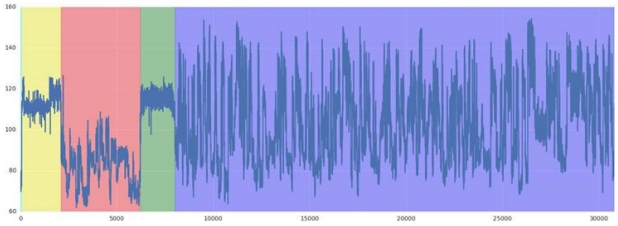

Figure 3: Example Segmented Squiggle. Note the read is fed through the pore in the 3'-5' direction so the last (purple) segment is the transcript. The poly(A) tail segment is shown in green.

And below is a figure showing an example of the HMM:

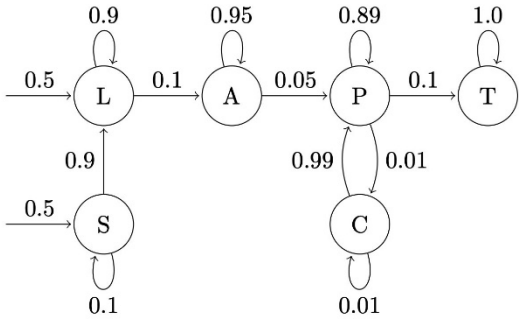

Figure 4: Example of hidden markov model with states: start (S), leader (L), adaptor (A) and polyA (P). As shown in fig.2, the transition probabilities control the way the hidden state at time t is chosen given the hidden state at time.

The tailfindr algorithm works as follows:

- Read fast5 file
- Z-normalize raw data
- Clip signal values exceeding 3 sd
- Smoothen by sliding window
- Define poly(A)-containing segment by thresholding smoothened signal by 0.3 (this defines the rough poly(A) boundaries
- Compute mean of every 25 samples of clipped signal within rough poly(A) boundaries
- Compute slope between every two consecutive points of mean signal
- Get precise poly(A) boundaries by shrinking rough poly(A) boundaries until slope signal is confined in +/- 0.3 range
- Normalize tail length by read specific translocation rate

This is shown in the below figure:

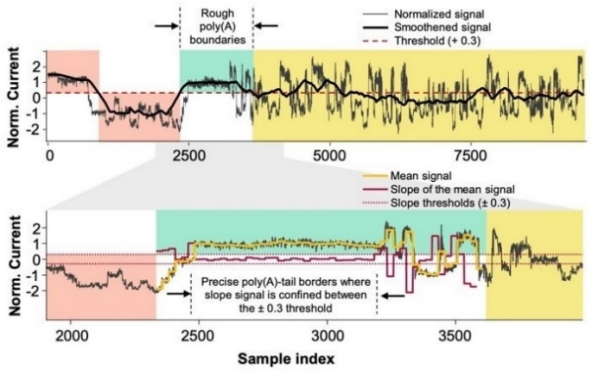

Figure 5[1]: Representative normalised signal data from eGFP-RNA sequencing. Red background indicates ONT adapter signal, green background represents rough borders of poly(A) signal, yellow background highlights signal from RNA sequence. Boundaries are defined based on tailfindr algorithm.

3 Results

We were able to analyze two samples from the dataset using tailfindr to compare to nanopolish. These samples included transcripts with tails with lengths of approximately 10 and 30 bases. Tailfindr performs little better with short tails (10 bases):

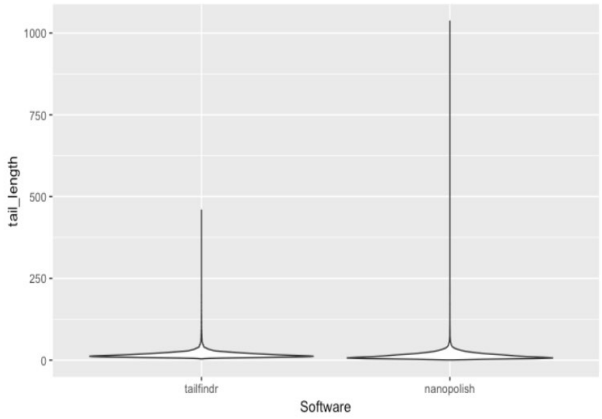

Figure 6: Violin plot of tail lengths using tailfindr and nanopolish for 10 bases tail length.

Concerning the mean and median of lengths, both tailfindr and nanopolish performed similarly while nanopolish tends to estimate longer outliers. The details of results are shown in table 1:

STAT	Tailfindr	Nanopolish
Min	3.87	0.69
1st Q	11.28	7.15
Median	14.83	11.43
Mean	17.58	14.57
3rd	20.24	18.00
Max	458.50	1036.33

Table 1: Summary statistics of tail lengths estimated by tailfindr and nanopolish

To further investigate the performance of tailfindr and nanopolish, we visulized the results using histogram and violin plot as shown in fig.7:

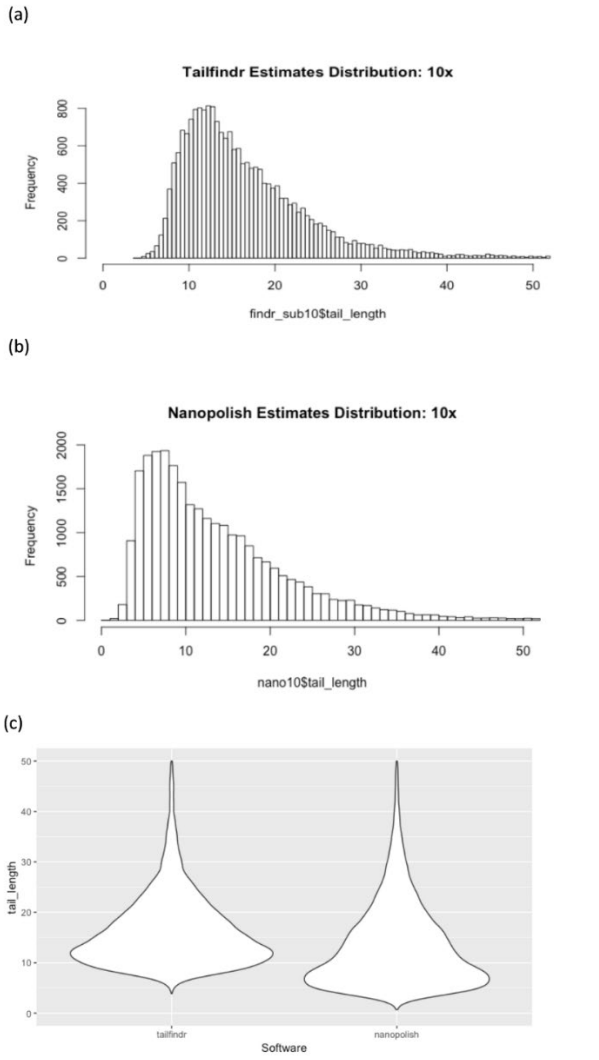

Figure 7: Results of 10 bases poly(A) tail estimation. a). Histogram of tailfindr tail-length estimations not including outliers. B). Histogram of nanopolish tail-length estimations not including outliers c). Violin plot of tail lengths using tailfindr and nanopolish not including outliers

In fig.7(c), with length of 10 bases, the estimation from tailfindr was higher than 10 (around 11) while that from nanopolish was lower than 10 (around 6).

For ploy(A) tails with length of 30 bases, we performed similar analysis and the results of both tools were pretty similar:

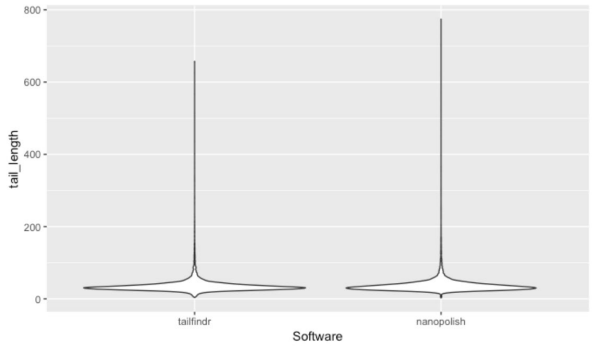

Figure 8: Violin plot of tail lengths using tailfindr and nanopolish on sample with tail lengths of 30.

As aforementioned, we routinely observed slightly higher estimates from Nanopolish which can be attributed to differences in normalization. The details of results are shown in table 2:

STAT	Tailfindr	Nanopolish
Min	3.86	2.77
1st Q	27.22	27.42
Median	31.95	32.89
Mean	35.71	37.48
3rd	38.44	40.52
Max	657.48	774.24

Table 2: Summary statistics of tail lengths estimated by tailfindr and nanopolish for sample with tail lengths of 30 bases

From the table, we observed that all statistics from tailfindr were closer to the real 30 bases tail length, which means tailfindr should be a better choice. However, tailfindr require powerful RAM and CPU which could be time consuming for regular computer.

The results of tailfindr and nanopolish are shown in fig.9:

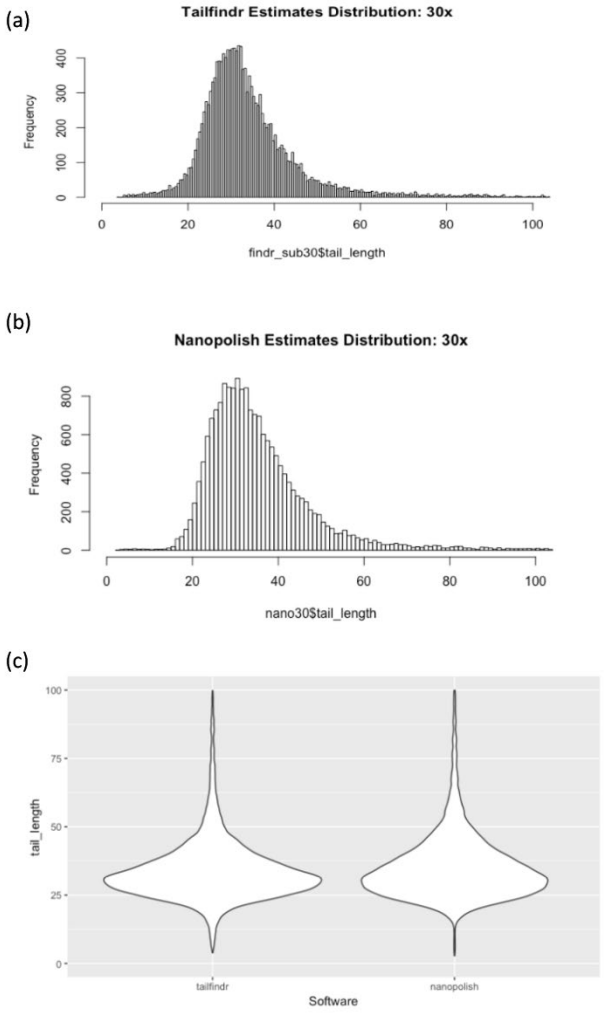

Figure 9: Results of 30 bases poly(A) tail estimation. a). Histogram of tailfindr tail-length estimations not including outliers. B). Histogram of nanopolish tail-length estimations not including outliers c). Violin plot of tail lengths using tailfindr and nanopolish not including outliers

Though nanopolish tends to overestimate the length, the results were pretty similar. Overall, for tails of length 30, both tailfindr and nanopolish performed similarly.

4 Discussion

In this project, we presented tailfindr, an R package to estimate poly(A) tail length on ONT long-read sequencing data. tailfindr operates on unaligned, basecalled data. It measures poly(A) tail length from both native RNA and DNA sequencing, which makes poly(A) tail studies by full-length cDNA approaches possible for the first time. We assessed

tailfindr's performance across different poly(A) lengths, demonstrating that tailfindr is a versatile tool providing poly(A) tail estimates across a wide range of sequencing conditions.

Nanopolish, a software package for signal-level analysis of Oxford Nanopore sequencing data, can calculate an improved consensus sequence for a draft genome assembly, detect base modifications, call SNPs and indels with respect to a reference genome and more. In our project, we demonstrated the poly(A) tail length estimation using nanopolish.

Also, this project presented a number of challenges. First, because native RNA Seq is a very new technology, there are very few datasets available. Additionally, the tools we used are still in pre-print and were very difficult to install, especially without using a virtual environment. We also had to wait on answers from the author of tailfindr since there were a few bugs that made it difficult to process the FAST5 files. Time was most likely the biggest issue for us however. The file sizes were quite large and required numerous time-consuming steps; transferring, untarring, basecalling, and finally tail length approximation.

With regard to the results, both tools output similar estimates. Even though the author of tailfindr suggests that nanopolish tends to overestimate tail lengths, this was not consistent with our results for the tails of length 10. However, we would need to process more samples to come to a more solid conclusion.

While tailfindr boasts that it is alignment free, we found the process to be very time consuming and computational intensive. Using multiple cores had a significant impact on runtime. Additionally, it should be noted that while nanopolish does require alignment to estimate tail lengths, it is not solely used to do this and is in fact a suite of tools for working with nanopore data.

Going forward, it would be necessary to process more datasets. The paper on nanopolish also reports that its estimates are sensitive to the sequence that comes before the poly(A) tail[1]. We were unfortunately unable to process this sample that was included in the dataset but it would be interesting to see if tailfindr has the same issue. Also of interest would be to compare estimates made using native RNA seq and cDNA (therefore measuring poly(T) tail length). And finally, once an established "best practices" have been established, it will be interesting to investigate the impact of poly(A) tail length on protein levels to illuminate biological function.

Acknowledgements

Thank you to Roham Razaghi from the Timp lab for all the help. Also, thanks to Sam Kovaka and Michael Schatz from the schatz lab.

References

1. Workman, R.E., et al., *Nanopore native RNA sequencing of a human poly(A) transcriptome.* bioRxiv, 2018: p. 459529.
2. Proudfoot, N.J., *Transcriptional termination in mammals: Stopping the RNA polymerase II juggernaut.* Science, 2016. **352**(6291).
3. Jalkanen, A.L., S.J. Coleman, and J. Wilusz, *Determinants and implications of mRNA poly(A) tail size - Does this protein make my tail look big?* Seminars in Cell & Developmental Biology, 2014. **34**: p. 24-32.
4. Lima, S.A., et al., *Short poly(A) tails are a conserved feature of highly expressed genes.* Nature Structural & Molecular Biology, 2017. **24**(12): p. 1057-+.
5. Reuter, J.A., D.V. Spacek, and M.P. Snyder, *High-Throughput Sequencing Technologies.* Molecular Cell, 2015. **58**(4): p. 586-597.
6. Niedringhaus, T.P., et al., *Landscape of Next-Generation Sequencing Technologies.* Analytical Chemistry, 2011. **83**(12): p. 4327-4341.
7. Krause, M., et al., *tailfindr: Alignment-free poly(A) length measurement for Oxford Nanopore RNA and DNA sequencing.* bioRxiv, 2019: p. 588343.
8. Quick, J., et al., *Real-time, portable genome sequencing for Ebola surveillance.* Nature, 2016. **530**(7589): p. 228-+.