Computational methods for studying RNA caps and poly(A)-tails at single-molecule resolution with Nanopore sequencing

Adnan Muhammad Niazi

Thesis for the degree of Philosophiae Doctor (PhD)
University of Bergen, Norway
2022

UNIVERSITY OF BERGEN

2022 Computational methods for studying RNA caps and poly(A)-tails at single-molecule resolution with Nanopore sequencing • Adnan Muhammad Niazi

Computational methods for studying RNA caps and poly(A)-tails at single-molecule resolution with Nanopore sequencing

Adnan Muhammad Niazi

Thesis for the degree of Philosophiae Doctor (PhD)
at the University of Bergen

Date of defense: 24.06.2022

Year: 2022

Title: Computational methods for studying RNA caps and poly(A)-tails at single-molecule resolution with Nanopore sequencing

Name: Adnan Muhammad Niazi

Print: Skipnes Kommunikasjon / University of Bergen

Scientific Environment

All the computational work was performed at Valen Lab in the Computational Biology Unit, Department of Informatics, University of Bergen, Norway. All biological experiments and data collection were performed at Valen group's laboratory in Sars International Centre for Marine Molecular Biology, University of Bergen, Norway.

The work was supported by the Bergen Research Foundation, the Sars International Centre for Marine Molecular Biology core funding, the University of Bergen core funding, and the Norwegian Research Council grants.

Dedicated to Dad

You taught me that anything coming out of my hands needs to be my best. Although you are not here today to see it, you are in every page. I was lucky to have you as my father.

Acknowledgement

I would like to extend my deepest gratitude to my supervisor, Eivind Valen, for taking me under his wing back in 2018 when I was a noob and did not know much about biology or bioinformatics. From the get-go, he made me feel at home in his group and helped me develop as a scientist. I like to focus on one thing or one project at a time — rather than frequently switching back and forth between different projects; Eivind was very accomodating of it, and never pushed me to spread my tentacles to more than one project at a time. This allowed me to work with such a focus and flow during the four years of my PhD that I genuinely enjoyed my PhD work. Thank you Eivind for being so flexible with me and for your valuable and kind feedback at every stage in my PhD.

I am deeply indebted to Maximillan Krause, who immediately took me under his daily supervision after I joined Eivind's lab. He groomed me at every initial stage of my PhD — from doing a paper blitz to presenting at group meetings, to developing my understanding of biology. As time went by, we became partners in science: he would generate the data, I would analyze it, and together we would decide what experiments to do next. Our daily discussions would last hours — it was an ideal partnership. My work on poly(A) tails and cap structures would not have been possible without the hard work that Max had put into designing and optimizing the sequencing protocols in the lab. Sadly, our partnership ended when I was in the last year of my PhD when Max moved on to a new position back in his homeland. Thank you Max for everything you did for me — from the great personal and career advice you gave me to the robust and hard-to-die indoor plants that you shared with me; you were a great colleague and an equally great mentor.

I am also grateful to Jan Inge Øverbo, my new partner-in-science, who took over the biology side of things after Max's departure. Thank you for your hard work in generating the crucial bits of data in such a short time; it helped me immensely in completing my thesis on schedule. Although I have written this thesis with much of our work on cap structures still cooking in the lab, I am sure that our partnership in the future during my postdoc will bear fruit. My work on decoding cap structures would not be possible without the hard work that you are putting into designing and testing the protocols that generate the data I need.

I very much appreciate the unconditional help that my friend and colleague Teshome Bizuayehu has provided to me throughout these years. Whenever I had a problem understanding something, or if I needed a second opinion on the correctness of my methods, Teshome was my go-to guy to discuss it with. He would listen to me and try his very best to patiently understand my reasoning, and would always give me extremely valuable feedback. Thank you Teshome for sharing your in-depth knowledge with me, Max, and Jan Inge, and for helping us perfect our methods.

I would also like to thank all my lab mates Håkon Tjeldnes, Preeti Kute, Thomas Stautland, and Yamilla for their useful feedback during these years, and to Kornel Labun for teaching me in detail how to get started with the CBU server and for his helpful pointers on machine learning.

I feel very indebted to my master's thesis supervisors in the Netherlands — Marcel van Gerven and Mannes Poel. Without the countless reference letters that they sent to different institutions on my behalf when I was looking for a PhD position, I would not have scored a PhD position at UiB in Norway. Thank you both for your unwavering support in the progression of my career.

I would not have been able to complete my PhD without the unconditional love and support of my wife who patiently tolerated my long working hours and weekend visits to the office, and never complained. My PhD would not have been as easy as it was without the nice food that you always made, the house that you made squeaky clean, and our lovely daughter that you raised all by yourself because I was so busy with my PhD. Thank you for all you have done for me.

I would like to express my sincere gratitude to the members of my PhD committee — Albin Sandelin, Anthony Mathelier, and Sushma Grellscheid — for their valuable feedback and for their time and effort. I hope to pay it forward one day.

I am also thankful to my secondary supervisor Pekka Parviainen for his kind support, time, and effort during my PhD.

Lastly, I would like to thank my mother, my siblings, my in-laws, and my close friends back home — Naeem, Zaheer, and Arbab Waheed — for being a great support for me throughout my PhD.

Publications

Peer-reviewed publications

†Krause, M., †**Niazi, A. M.**, Labun, K., Müller F.S., Cleuren Y. N. T., Valen E. (2019) *tailfindr*: Alignment-free poly(A) length measurement for Oxford Nanopore RNA and DNA sequencing. RNA. DOI: 110.1261/rna.071332.11

†*Contributed equally*

Book chapters

†**Niazi, A. M.**, †Krause, M., & Valen, E. (2021). Transcript isoform-specific estimation of poly(A) tail length by Nanopore sequencing of Native RNA. In Methods in Molecular Biology (pp. 543–567). Springer US. DOI: 10.1007/978-1-0716-1307-8_30

†*Contributed equally*

Other contributions

Begik, O., Liu, H., Delgado-Tejedor, A., Kontur, C., **Niazi, A. M.**, Valen, E., Giraldez, A. J., Beaudoin, J.-D., Mattick, J. S., Novoa, E. M. (2021). Nano3P-seq: Transcriptome-wide analysis of gene expression and tail dynamics using end-capture Nanopore sequencing. 10.1101/2021.09.22.461331

Bizuayehu, T. T., Labun, K., Jakubec, M., **Niazi, A. M.**, Jefimov, K., & Valen, E. (2020). Single-Molecule Structure Sequencing reveals RNA structural dependencies, breathing, and ensembles. DOI: 10.1101/2020.05.18.101402

Software tools developed

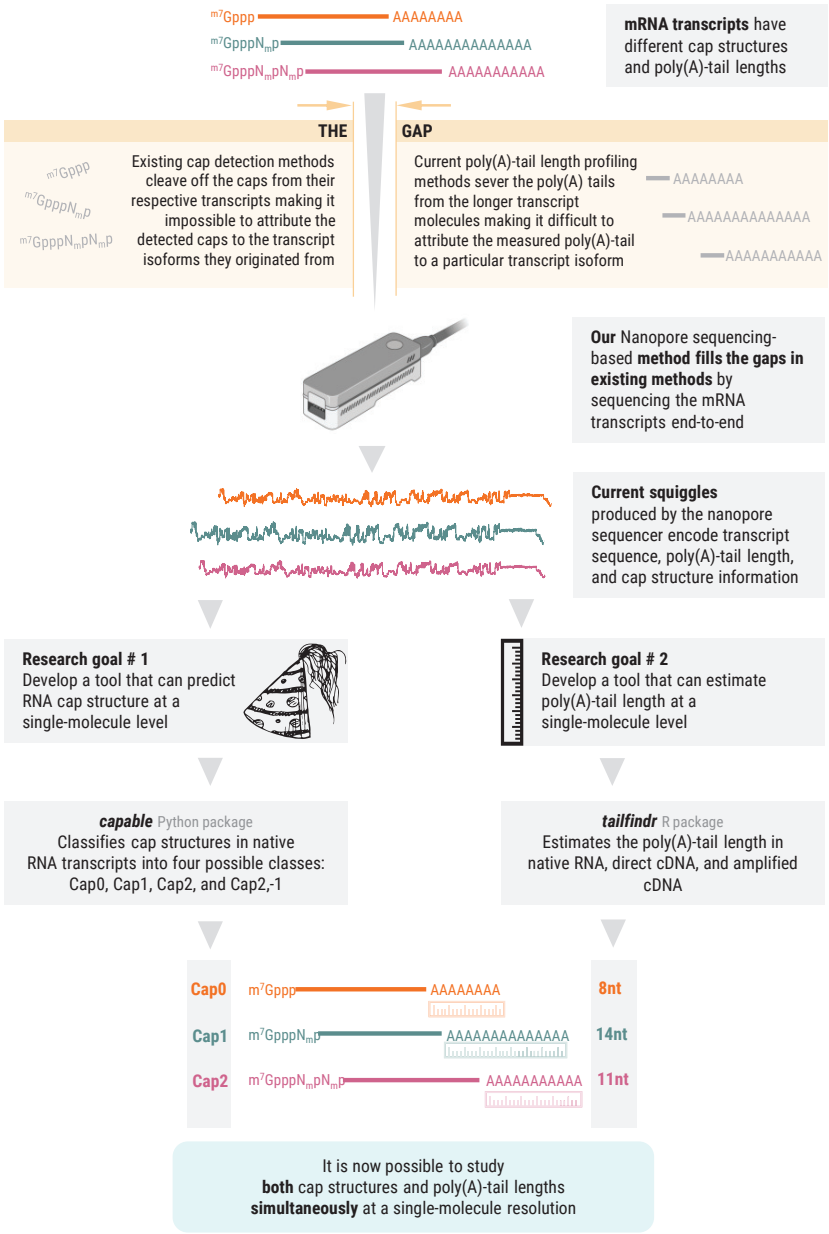
1. ***tailfindr*** | An R package for estimation of poly(A)-tail lengths in RNA and cDNA at a single-molecule level

   `https://github.com/adnaniazi/tailfindr`

2. ***Capable*** | A Python package for cap type prediction at a single-molecule level

   `https://github.com/adnaniazi/capable`

Graphical Abstract



Abstract

mRNA caps are 5'-terminal modifications that happen co-transcriptionally on Pol-II transcribed transcripts in eukaryotes. These caps can be of different types depending on the modifications present on the first and second transcribed nucleotides of the mRNA. Together with poly(A) tails — a homopolymer stretch of adenosines — at the 3'-end of the mRNA, the caps help the mRNA form into a pseudo circular loop that recruits ribosomes in the cytosol for translating the message encoded in the messenger RNA into protein. Both the poly(A) tail length and cap type can therefore have important consequences in the expression of a transcript, and understanding this interaction is important in understanding the complex world of RNA biology.

Currently, there exists no method that can study these two distant ends of the mRNA simultaneously in the same assay. Most of the methods for cap structure determination are bulk methods that rely on severing the cap from their respective transcripts and then finding their relative abundance in the sample. Once the caps are cleaved off from their respective transcripts, it is impossible to attribute the quantified caps to their respective transcripts. Thus existing cap methods cannot provide transcript isoform-level or even gene-level information about the cap structures in a transcriptome-wide manner. Similarly, most of the existing poly(A) tail profiling methods can only sequence a small portion of the transcript proximal to the measured poly(A) tail. This small sequenced transcript fragment may not have sufficient discriminative power to distinguish between transcript isoforms that share the same 3' polyadenylation site but have different 5' compositions. Thus an estimated poly(A) tail could not be unambiguously assigned to a particular isoform. In such cases, only gene-level poly(A) tail length assessment can be done.

We have developed a Nanopore sequencing-based method to sequence a native RNA molecule end-to-end. In doing so, we can study both the poly(A) tail and RNA caps — and possibly other RNA modifications in the future — simultaneously in a single assay with single-molecule resolution. The *capable* frame I have developed uses a machine learning model trained on Nanopore current-based features of different caps to predict the cap structures on individual RNA molecules. On the very same reads, the *tailfindr* framework that I developed can be used to estimate the poly(A) tail lengths. Together, these two tools enable us to simultaneously study the caps and the poly(A) tails transcriptome-wide — one long molecule at a time.

Abstract (Norwegian)

mRNA caps er 5'-terminale modifikasjoner som skjer co-transkripsjonelt på Pol-II transkriberte transkripsjoner i eukaryoter. Disse hettene kan være av forskjellige typer avhengig av modifikasjonene som er tilstede på de første og andre transkriberte nukleotidene til mRNA. Sammen med poly(A) haler — en homopolymer strekning av adenosiner — i 3'-enden av mRNA, hjelper kappene mRNA til å danne seg til en pseudo sirkulær løkke som rekrutterer ribosomer i cytosolen for å oversette meldingen kodet i messenger RNA til protein. Både poly(A) halelengde og cap type kan derfor ha viktige konsekvenser i uttrykket av et transkripsjon, og forståelsen av denne interaksjonen er viktig for å forstå den komplekse verden av RNA-biologi.

For tiden eksisterer det ingen metode som kan studere disse to fjerne endene av mRNA samtidig i samme analyse. De fleste metodene for bestemmelse av cap-struktur er bulkmetoder som er avhengige av å skille hetten fra sine respektive transkripsjoner og deretter finne deres relative overflod i prøven. Når hettene først er klippet av fra deres respektive transkripsjoner, er det umulig å tilskrive de kvantifiserte hettene til deres respektive transkripsjoner. Dermed kan ikke eksisterende cap-metoder gi transkripsjonsisoform-nivå eller til og med gen-nivå informasjon om cap-strukturene på en transskriptomomfattende måte. Tilsvarende kan de fleste av de eksisterende poly(A)-haleprofileringsmetodene bare sekvensere en liten del av transkripsjonen proksimalt til den målte poly(A) halen. Dette lille sekvenserte transkripsjonsfragmentet har kanskje ikke tilstrekkelig diskriminerende kraft til å skille mellom transkriptisoformer som deler det samme 3'-polyadenyleringssetet, men har forskjellig 5'-sammensetninger. Dermed kunne ikke en estimert poly(A)-hale utvetydig tilordnes en bestemt isoform. I slike tilfeller kan kun poly(A) halelengde-vurdering på gennivå gjøres.

Vi har utviklet en Nanopore-sekvenseringsbasert metode for å sekvensere et naturlig RNA-molekyl ende-til-ende. Ved å gjøre det kan vi studere både poly(A)-hale og RNA-hettene — og muligens andre RNA-modifikasjoner i fremtiden — samtidig i en enkelt analyse med enkeltmolekyloppløsning. Den *capable* rammen jeg har utviklet bruker en maskinlæringsmodell trent på Nanopore-strømbaserte funksjoner til forskjellige caps for å forutsi cap-strukturene på individuelle RNA-molekyler. På samme måte kan *tailfindr*-rammeverket som jeg utviklet, brukes til å estimere poly(A)-halelengdene. Sammen gjør disse to verktøyene oss i stand til å studere hettene og poly(A)-halene i hele transkriptomet samtidig — ett langt molekyl om gangen

Contents

1 Introduction **1**

1.1 Current state-of-the-art and aims of this thesis 2
1.2 Methodology in brief 3
1.3 Significance of this work 4
1.4 Thesis structure 5

2 Background **7**

2.1 Sequencing 7
2.2 Short-read sequencing of RNA 7
2.2.1 Shortcomings of Illumina sequencing 8
2.3 Long-read sequencing of RNA 10
2.3.1 Advantages of long-read Nanopore sequencing 13
2.3.2 Challenges with long-read sequencing 13
2.4 Nanopore sequencing methods for RNA 14
2.4.1 Native or direct RNA sequencing 14
2.4.2 PCR-Amplified cDNA sequencing 14
2.4.3 Direct cDNA sequencing 15
2.5 Nanopore output format (FAST5) 15
2.6 Basecalling — with Guppy 16
2.6.1 Move and Trace tables in basecalled FAST5 files 16
2.7 Basecalling — with Bonito 20
2.8 Alignment 20

3 Single-molecule level prediction of mRNA cap types with *capable* **21**

3.1 Introduction 21
3.2 Biological role of caps 25
3.2.1 Cap 0 25
3.2.2 Cap 1 26
3.2.3 Cap 2 26
3.2.4 Cap m6Am 26
3.2.5 TMG cap 27

3.2.6 NAD+ and NADH caps 27
3.2.7 FAD caps 28
3.2.8 UDP-Glc and UDP-GlcNAc caps 28
3.3 Existing methods for cap type prediction 28
3.3.1 Radio-isotope labeling-based method 28
3.3.2 Mass spectrometry-based methods 30
3.3.3 NGS-based method 31
3.3.4 Nanopore sequencing-based method 33
3.4 Limitations of existing methods 33
3.5 Nanopore sequencing of RNA cap and the challenges involved 35

REDACTED

4 Single-molecule prediction of poly(A) tail length in Native RNA and cDNA with *tailfindr* 75
4.1 Poly(A) tails and their biological role 75
4.2 Poly(A)-tail profiling 76
4.3 State-of-the-art for poly(A)-tail profiling 77
4.3.1 extension PolyA-tail test (ePAT) 77
4.3.2 Poly(A) profiling by sequencing (PAL-seq) 77
4.3.3 TAIL-seq 78
4.3.4 Poly(A) inclusive RNA isoform sequencing (PAIso-seq) 81
4.4 Limitations of existing poly(A)-tail profiling methods 81
4.5 Nanopore sequencing of poly(A) tails and the challenges involved . . 83
4.6 Our method – *tailfindr* – in brief 84
4.7 Our published work on *tailfindr* with more details 86

4.8 Poly(A)-tail profiling in PCR-amplified cDNA (in-house protocol) . . 99
4.9 Poly(A)-tail profiling of rolling circle-amplified cDNA (in-house protocol)100
4.10 Poly(A)-tail profiling cDNA (ONT protocol) 104
4.11 Discussion and future perspectives 107

5 Conclusion and future outlook **109**

Bibliography **113**

A Appendix **133**

1 Introduction

Ribonucleic Acid or RNA is a central component of all life. In eukaryotes, RNA polymerase II (pol II) transcribes DNA in the cell's nucleus to nascent messenger RNA (or pre-mRNA).

When the transcribed pre-mRNA is 20–30nt long, a series of enzymatic reactions add a methylated guanosine (m^7G) to the 5'-end of the nascent RNA with a 5'-5' triphosphate bridge. The terminal inverted m^7G base is called cap0 (also represented by the notation m^7GpppNp-RNA, where N is the first transcribed nucleotide of the pre-mRNA). A cap0 prevents the degradation of the growing RNA strand by exonucleases and helps recruit the translation initiation factors for RNA translation[1, 2].

Cap0-capped transcripts can be used as substrates by cap methyltransferase 1 (CMTR1) enzyme that imparts a methyl group on the ribose sugar backbone of the first transcribed nucleotide to form a cap1 structure (m^7GpppNmp-RNA). A cap1 acts as a maker that a cell uses to distinguish its RNA from viral RNA as viral RNAs typically lack cap1 structure [3].

On some of the cap1 transcripts, an additional enzyme, cap methyltransferase 2 (CMTR2), can methylate the ribose sugar of the second transcribed nucleotide resulting in a cap2 structure (m^7GpppNmpNmp-RNA) [4]. As much as half of the poly(A)-tailed RNA in humans have a cap2 cap [5]. Recently, it has been found that methylation of second transcribed nucleotide in cap2 impacts protein production level in a cell-specific manner and contributes to RNA immune evasion [6].

Recently, more cap structures have come to light in both eukaryotes and prokaryotes. The new caps include cap0-m^6A (m^7Gpppm^{6}Ap-RNA), cap1-m^6Am (m^7Gpppm^{6}Amp-RNA), and caps that contain metabolic cofactors such as nicotinamide adenine dinucleotide (NAD), flavin adenine dinucleotide (FAD), uridine diphosphate glucose (UDP-Glc), and uridine diphosphate N-acetylglucosamine (UDP-GlcNAc) [**Bird2018-sr**, 7, 8]. All these different cap structures constitute an additional layer of epitranscriptomic complexity that may have a crucial role in determining the fate of the RNA in health and disease.

When the transcription of a capped pre-mRNA is close to completion, it undergoes polyadenylation in which a poly(A) polymerase adds a long stretch of adenosines — or a poly(A) tail — to the 3'-end [9]. The newly-formed poly(A) tails have a tightly constrained species-specific length which ensures that the poly(A)-binding proteins (PAB) can bind efficiently to the mRNA and export it out of the nucleus and into the cytoplasm [10]. In the cytoplasm, the poly(A) tails may shorten [11], and the two ends of the mRNAs form a pseudo-circular loop [12, 13]. Finally, ribosomes get recruited to these loops and start translating the mRNAs into proteins [14].

The interaction between different 5'-cap types and different length 3' poly(A) tails may have significant consequences for the fate of mRNA and gene expression. However, methods for studying these two opposite ends of the mRNA simultaneously at a transcriptome-wide single-molecule level have been non-existent — until now. This thesis is a small step in developing tools that allow us, for the very first time, to study RNA caps and poly(A)-tails — and possibly more RNA modifications in the future — simultaneously at a single-molecule level. The tools developed will be instrumental in understanding the complex world of RNA — one molecule at a time.

1.1 Current state-of-the-art and aims of this thesis

Existing methods for determining cap structures on mRNA transcripts require severing off the cap from their respective transcripts and then using either gel- or mass-spectrometry-based methods to separate and identify the different cap types. These bulk methods lack transcript-level specificity — or even gene-level specificity, for that matter — and can, at the most, only give a relative abundance estimate of different cap structures present in an RNA sample. The lack of methods for cap structure prediction at single-molecule resolution represents a significant bottleneck in understanding the transcriptome-wide role of different cap structures. A single-molecule cap prediction method can shed light on the factors that influence the presence of one or the other cap type on a transcript and inform about the role that these different caps play in the fate of their respective transcripts. Thus, our first research goal is:

R1: *To develop a method that can predict the cap structure present on an individual RNA transcript molecule*

The 5'-caps interact with the 3'-poly(A) tails in the cytoplasm. Many methods exist for estimating the poly(A)-tail length. However, we cannot study poly(A)-tails at a transcript isoform resolution because current methods rely on Illumina sequencing that sequences only a tiny part of the transcript proximal to the measured poly(A). The small transcript fragment does not have sufficient discriminatory power to help us unambiguously assign the measured poly(A) tail length to a particular isoform when the poly(A) cleavage site is identical for the different isoforms. As a result, in such cases we can only attribute the measured poly(A) tail to a gene, but not to any of its isoforms. However, the poly(A) tails of different transcript isoforms of a gene might have gone through varying levels of poly(A) tail length regulation. Thus, collapsing the poly(A) tail lengths of different transcript isoforms across a gene — as all existing methods do — may produce a blurry and misleading picture of the poly(A)-tail dynamics. Therefore, there is a need for a technique that can shed light on transcript isoform-specific differences in poly(A) tail lengths. Therefore, our second research goal is:

R2: *To develop a method that can estimate poly(A) tail length of individual RNA transcript molecules*

As both the 5'-cap and 3'-poly(A) tail interact during translation to form a closed-loop structure, therefore, the particular cap structure present on the RNA and length of the poly(A) tail length may have some critical consequences. To date, it has been impossible to assess transcriptome-wide interactions between poly(A) tail and RNA caps, primarily because none of the existing methods can probe the cap structures and poly(A) tail lengths simultaneously for individual RNA molecules. I aim to develop the cap structure prediction and poly(A)-tail profiling methods in such a way that we can study both of them simultaneously for every RNA molecule.

1.2 Methodology in brief

Our method for simultaneous probing of cap structure and poly(A)-tail length in mRNA molecules relies on Nanopore sequencing. This nascent technology enables us, for the very first time, to sequence Native RNA, i.e., sequencing RNA directly without reverse transcribing it to cDNA first. Competing short- and long-read sequencing methods from Illumina and Pacific Biosciences, respectively, cannot

sequence native RNA and require it to be converted into cDNA first, which flushes away base modification information on the RNA.

In Nanopore sequencing, the RNA is fed through a protein nanopore suspended in a membrane that separates two ionic buffer-filled wells. A voltage applied across the membrane sends a current through the pore which is disrupted by a translocating RNA strand. Any modifications on RNA, including poly(A) tails and cap methylations, result in a distinct signature in the pore current, which can, theoretically, be decoded to predict the type of modification.

We have found that methylated cap bases have a different current signature and dwell longer in the pore than unmethylated bases. Furthermore, 2'-O methylations on the first and second transcribed nucleotides also increase the dwell time of 11^{th} and 12^{th} transcribed nucleotides, respectively. A classifier trained on these cap-specific signatures from all possible cap permutations can be used to probe the methylation status, and hence the cap structure, of an individual RNA transcript. Additionally, when the poly(A) tail of a capped-transcript passes through the pore, it results in a monotonic current signal due to the lack of sequence diversity in the poly(A) tail. The time duration of this monotonous stretch of current encodes the length of the poly(A) tail, which can then be used to estimate the poly(A) tail length in nucleotide units. In this way, our methods can simultaneously probe both the cap structure and poly(A) tail length at single-molecule resolution using Nanopore sequencing of mRNA.

1.3 Significance of this work

To say that the world of RNA is complicated is an understatement: In addition to caps and poly(A) tails, RNAs have alternative polyadenylation sites, regulatory sequences in the UTRs, alternative splice sites, and around 170 different modifications, to name just a few. We can unravel the myriad ways these various components of RNA interact in the biology of health and disease only if these components are studied in tandem — and not in isolation. Developing tools for studying RNA cap structure and poly(A) tails simultaneously is a small step in that direction. With nanopores enabling sequencing of anything from DNA [15] to RNA [16] and from proteins [17] to metabolites [18, 19], we will soon be able to shed unprecedented light on the complex inner workings of the RNA — and the tools developed in this thesis are a stepping-stone towards this end.

1.4 Thesis structure

Chapter 2 gives a detailed background on Illumina sequencing and motivates the use of Nanopore sequencing by highlighting some of the critical shortcomings of Illumina sequencing. Next, I explain the inner workings of Nanopore sequencing, its data format, and some of the key steps in processing this data.

Chapter 3 deals with our first research goal (R1) of developing a method for predicting cap structure at a single-molecule resolution using Nanopore sequencing. I first explain the different cap structures and some techniques for cap detection and list their shortcomings that make Nanopore sequencing an ideal candidate for the task. Next, I discuss some of the challenges involved in sequencing RNA caps through Nanopore and how I have attempted to solve these challenges in the method — *capable* — that I developed.

Chapter 4 deals with my second research goal (R2) of developing a method for poly(A)-tail length prediction at a single-molecule level using Nanopore sequencing. First, I motivate our approach by explaining some of the shortcomings of Illumina- and PacBio-based methods for poly(A)-tail profiling and how I bridge these gaps with the Nanopore sequencing-based method – *tailfindr*. The details on how tailfindr profiles poly(A) tails in RNA and unamplified DNA with Nanopore sequencing are then explained in the appended peer-reviewed paper. After this paper, I discuss my subsequent work for poly(A)-tail profiling on amplified cDNA using Nanopore sequencing.

Lastly, chapter 5 summarizes the contributions made in this thesis and discusses the future directions of this research.

2 Background

2.1 Sequencing

Sequencing determines the order of the chemical building blocks that make up an RNA, DNA, or a peptide [16, 15, 17, 20]. In DNA and RNA, these building blocks are called bases or nucleotides, whereas in peptides they are called amino acids.

DNA and RNA have four possible canonical bases: adenosine(A), guanine(G), cytosine(C), and the fourth base is thymine(T) in DNA and uracil(U) in RNA.

The sequence content of DNA and RNA encodes important genetic and regulatory information. For instance, DNA sequencing can tell us which genes are present in the genome, and RNA sequencing can tell us which genes are expressed, i.e., turned ON or OFF, and how much they are expressed.

Two of the most prevalent sequencing technologies today are short-read and long-read sequencing. Below, I will discuss these two methods and their relative strengths and weaknesses.

2.2 Short-read sequencing of RNA

Short-read sequencing, as the name indicates, can sequence only short fragments (50-300 nt) of DNA. Illumina sequencing is one of the most widely used short-read sequencing technologies.

Briefly, in Illumina sequencing, the RNA to be sequenced is first sheared into small fragments that are then used to synthesize complementary DNA (cDNA). Sequencing adaptors attached to the ends of the size-selected cDNA fragments subsequently hybridize to the complementary oligos present on the Illumina flowcell. Bridge amplification amplifies each original fragment into a cluster of spatially-close copies. Sequencing begins by first hybridizing a sequencing primer to each strand in a cluster. Next, fluorescently-tagged nucleotides are added. In each cycle of sequencing, only one of these nucleotides is incorporated in the growing strands of the cluster. As

this happens, a fluorescence signal is emitted which gives away the identity of the incorporated nucleotide and hence the sequence of the original fragment of interest (which is complementary to the incorporated nucleotide). If we want the length of the final reads to be 100 nt, then 100 cycles of fluorescent nucleotide incorporation are performed in the Illumina sequencer.

2.2.1 Shortcomings of Illumina sequencing

Sequencing of Native RNA is not possible

Illumina sequencing depends on clonal amplification of the fragment-of-interest to form a cluster. The formation of a cluster is crucial for the emission of a strong-enough fluorescence signal during each sequencing cycle. If the fragment-of-interest is not amplified, only one fluorescently-tagged nucleotide will be incorporated in each sequencing cycle and its fluorescent signal intensity will be too weak to be detected by the sequencer optics.

Clonal amplification can only be performed if the fragments are DNA and not RNA because currently, there is no viable method for clonal amplification of RNA. Furthermore, RNA is highly unstable. Even if someone invented a method for clonal amplification of RNA, the resulting RNA clusters would severely degrade with each new sequencing cycle as each cycle includes many chemical steps.

Illumina sequencing cannot detect RNA modifications directly

As Illumina sequencing requires the conversion of RNA into cDNA before sequencing, any modifications present on the RNA are lost in this conversion. Hence Illumina-based methods cannot directly read RNA modifications.

Elaborate ways around this limitation thus need to be devised for each individual modification that one wants to detect in the original RNA. For instance, to detect m6A modification, the m6A-seq protocol uses an m^6A-specific antibody to fish out fragments containing an m^6A-modified base [21]. On a similar note, to detect internal 2-O' methylations in RNA, the Nm-seq protocol [22] fragments the RNA and then repeatedly applies oxidation-elimination-dephosphorylation (OED) reaction which removes one base at a time from the 3'-end of RNA fragment in each cycle until a 2'-O methylated base is encountered. Thus all the RNA fragments left after multiple OED cycles have a 2'-O methylated base at the 3'-end. cDNA conversion

and sequencing follows. After aligning the reads to the reference, the 3'-ends of the mapped read clusters inform about internal 2'-O methylation sites.

There are approximately 170 known RNA modifications [23]. Devising bespoke workarounds to indirectly detect each of these modifications using Illumina sequencing is intractable.

The necessity of using PCR introduces biases in the sequenced output

Illumina sequencing relies on PCR for clonal amplification to create clusters. PCR is known to have a sequence bias. GC-repeat-rich regions get compressed, and homopolymer stretches cause the polymerase to stutter, resulting in amplified copies with fewer homopolymer bases than in the original fragment [24]. The PCR errors in the homopolymer stretches can cause incorrect estimates of the poly(A) tail lengths. Furthermore, PCR biases can also lead to inaccurate transcript abundance estimates.

Transcript isoform-level inferences are difficult to make

Short-read sequencing produces short-read fragments that are then aligned to the reference genome or transcriptome like a jigsaw puzzle to find the gene or isoform from which these pieces originated. In such an approach, it is often difficult to make isoform-level inferences because the individual pieces might not cover regions that distinguish between different isoforms. For instance, it may not be possible to assess if there are any isoform-specific differences in poly(A)-tail length in RNA when the isoforms have the same alternative polyadenylation site. As you will see later in chapter 4, this is because Illumina protocols for poly(A)-tail profiling can only sequence a fragment of RNA proximal to the poly(A) tail. These poly(A)-proximal short transcript fragments might be the same across distinct isoforms thereby limiting us from making any isoform-specific inferences about the poly(A) tail length.

The short length of Illumina reads is severely limiting

In Illumina sequencing, the read length is limited to 300 nt. This is because in every new cycle of Illumina sequencing, a new fluorescently-tagged nucleotide is incorporated in each growing strand of a cluster. However, in some cycles, a new nucleotide may not incorporate in one (or more) of the strands of the cluster, which causes these strands to lag behind the other strands — a so-called phasing error

[25]. Over many cycles, more and more of the strands lag behind, and the phasing error grows. This causes the fluorescent signal of the cluster to dilute progressively with each new cycle. Above 300 cycles, the cluster's fluorescent signal becomes so heterogeneous that the associated optics and electronics cannot determine the incorporated base in the cluster in a given cycle. That is why it is futile to perform more than 300 cycles of sequencing in Illumina.

Even with 300 nt read length, during QC, low-quality bases at the end of the reads are usually trimmed off leaving behind fragments that are shorter than 300-nt long reads we started with [26].

With such short read lengths, it is difficult to resolve complex regions in the genome, or study the two ends of transcripts simultaneously in a single read because such short reads cannot span the majority of transcripts end-to-end. For example, it is impossible to study poly(A) tail length in individual RNA isoforms in conjunction with the cap type on that isoform with Illumina sequencing.

2.3 Long-read sequencing of RNA

Long-read sequencing allows us to sequence full-length reads end-to-end. The size of the reads is limited only by the size of the RNA molecules one starts with during library prep. Oxford Nanopore Technologies (ONT) and Pacific Biosciences (PacBio) are at the forefront of this new sequencing paradigm. PacBio sequencing, like Illumina sequencing, cannot sequence RNA directly. Therefore, this thesis will focus on Nanopore sequencing only, which can sequence RNA [16], DNA [15], and — more recently — proteins as well [17]. In this thesis, *Nanopore* (with capital N) refers to the nanopore sequencing method developed by ONT.

In Nanopore sequencing, a protein pore is suspended in a membrane separating two ionic buffer-filled wells (see Fig. 2.1) [27]. A potential difference of approx. –180mV is applied across the membrane which causes a constant ionic current to flow through the pore. If an RNA molecule passes through the pore, it disrupts the otherwise constant current flowing through the pore; the larger the size of this molecule, the more the blockage in the pore current. In this way, any molecule passing through the pore creates characteristic modulations in the pore current that can help us decode that molecule's identity.

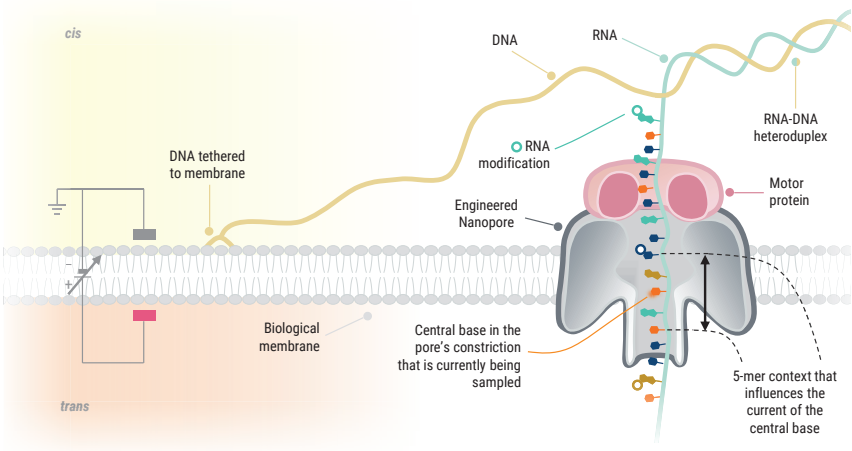


Fig. 2.1. **Nanopore sequencing of a native RNA molecule.** An engineered protein pore is suspended in a membrane which separates buffer-filled wells on *cis* and *trans* sides of a membrane. Across the membrane a voltage is applied. The translocation of the RNA in the DNA-RNA heteroduplex is controlled by the ratcheting action of a motor protein. A base — or any modification present on it — in the central constriction of the pore along with the two bases upstream and downstream of this central base (the 5-mer context) affects the ionic current signal that comes out of the pore.

Nanopore can sequence native RNA, i.e., RNA directly, without requiring it to be converted into cDNA first. Nanopore RNA sequencing, therefore, is also called Native RNA sequencing or Direct RNA sequencing.

To sequence RNA through the nanopore, it is loaded onto the *cis* side of the membrane. As RNA has a negative charge, the *trans* side of the membrane which has a positive potential on it, attracts the RNA, causing the RNA to translocate through the pore. As RNA goes through the pore, the otherwise constant ionic current flowing through the pore is now modulated by the translocating RNA nucleotides.

If the RNA were to pass through the pore under the influence of the applied voltage alone, it would pass through the pore at a very high speed of 1-22 μs/nt (120 mV applied voltage; α-Hemolysin pore) [24]. With electronic circuitry sampling the analog pore current at the rate of 3012 sample/s in a commercially-available ONT sequencer, translocation speed of 1-22 μs/nt translates into acquiring less than one current measurement (sample) per nucleotide — too few to decode these samples later on during basecalling. Each RNA nucleotide should spend at least 5 ms (approx.) in the pore so that at least 10 current samples can be acquired for each nucleotide at a sampling rate of 3012 samples/s. This means that the voltage-driven

RNA translocation should be slowed down by at least 100–1000 times for every base to be sampled sufficiently for basecalling later [28].

A possible strategy to slow down the translocation speed of the RNA to 5ms/nt is to apply a lower voltage. However, a lower voltage results in lower current amplitudes and higher noise in the signal making it difficult to distinguish between different bases. Therefore, to effectively slow down the RNA, a DNA oligo containing a biological motor protein (helicase) is ligated to the 3'-end of poly(A)+ RNA during library prep. During sequencing, this motor protein ratchets the RNA molecule through the pore at a slower more-controlled pace.

To effectively decode RNA nucleotides, the RNA must translocate through the pore at a consistent pace. One factor that can influence this consistency is the RNA secondary structure. RNA can fold onto itself in myriad ways that can impact the ratcheting speed of the motor protein [29]. Reverse transcribing the RNA before sequencing results in an RNA-DNA heteroduplex that can relax the secondary structure of the RNA. This results in 1) less variation in translocation speed during ratcheting of RNA by the motor protein, and 2) more throughput because fewer pore blockages happen due to a lack of secondary structures, which increases the number of bases sequenced in a sequencing run.

The current signature generated by a translocating nucleotide encodes not only for that nucleotide but also for its neighboring bases as well. Two neighboring bases on both sides of the central base — the so-called 5mer context — influence the current signature of the central nucleotide the most. Thus, the same nucleotide in different contexts may generate completely different current signatures when translocating through the pore.

The Nanopore's current signal (also called a squiggle) is saved in an array in a FAST5 file by MinKNOW — the data acquisition and experiment management software provided by ONT. The squiggle encodes the current signatures of the translocated RNA nucleotides — and possible modification on them. A base caller pre-trained on these signatures can then use the information in the squiggle to predict the original RNA sequence.

2.3.1 Advantages of long-read Nanopore sequencing

Native RNA can now be sequenced

Direct RNA sequencing — something impossible with short-read sequencing — can now be done with Nanopore sequencing. With native RNA sequencing, it is now theoretically possible to study all the different RNA modifications that get lost when converting RNA into cDNA. The ability to sequence RNA modifications opens up a whole new world of opportunities to answer fundamental biological questions that we previously could not answer due to technical limitations.

Long-range interactions can now be studied

Nanopore RNA sequencing read length is limited only by the size of the RNA molecules loaded in the sequencer. Complete RNA molecules can be sequenced, allowing us to get a comprehensive picture of the transcriptome. With reads that can span all transcript isoforms end-to-end, we can now study the two opposite ends of the RNA simultaneously to investigate the interaction of 3'-end poly(A) tails with 5'-caps.

2.3.2 Challenges with long-read sequencing

Basecalling accuracy is low

Nanopore sequencing accuracy for RNA is around 93.6%, which is low compared to Illumina sequencing (99.9%). When Nanopore sequencing became available for the first time, its accuracy of DNA basecalling was around 85%, which has now increased to 99.9% with advances in sequencing chemistry and software. The same progress is expected to happen for Nanopore RNA sequencing in the years to come. But for now, the low accuracy of the RNA Nanopore reads causes challenges in analyzing these reads.

Short RNA reads from Nanopore sequencing are difficult to basecall

Short Nanopore RNA reads (80–150 nt long) cannot be accurately basecalled, which is a major bottleneck in sequencing synthetic RNA oligos that can be, at the most, 100 nt long due to prohibitive costs of synthesizing these oligos any longer. The

inability to basecalled short RNA reads presents a major hurdle — as you will see in Chapter 3 — in creating training data for a classifier that can distinguish different cap structures.

RNA degradation is a major issue during library prep

RNA is highly susceptible to degradation by ubiquitous exonucleases that can degrade bases at both ends of the RNA. Moreover, RNA is very fragile and its strands may break during the library preparation. As we aim to study 3'-poly(A) tail and 5'-cap structures in this thesis, extreme caution is needed to protect these extremities from nucleases and strand breaks.

2.4 Nanopore sequencing methods for RNA

RNA can be sequenced via three different methods using Nanopore sequencing as explained below and summarized in Table 2.1.

2.4.1 Native or direct RNA sequencing

Direct-RNA sequencing (DRS) can sequence a native RNA molecule end-to-end along with any modifications that might be present on it. DRS requires starting with 500 ng of poly(A)+ RNA. Such large amounts of starting RNA may be impossible to obtain in some cases, which is why other sequencing methods (see below) can be helpful.

2.4.2 PCR-Amplified cDNA sequencing

If the starting amount of RNA is much lower than 500 ng, then amplifying it with PCR can generate enough material to do DNA sequencing on Nanopore. This method requires only 4 ng of starting poly(A)+ RNA. The RNA is reverse transcribed, and then strand switching and second-strand synthesis yields double-stranded cDNA that is then amplified using PCR.

As this method relies on converting RNA to cDNA, any RNA base modifications present in the RNA, such as cap methylations, are lost and cannot be studied. However, it is possible to estimate the poly(A)/(T) tail lengths because this information is still present in the amplified DNA.

The throughput of this method is higher due to faster motor proteins used in DNA sequencing. Furthermore, this method can also take advantage of the experimental basecaller, Bonito, to get 99.9% read accuracy.

2.4.3 Direct cDNA sequencing

The PCR-amplification step in the PCR-amplified cDNA sequencing (above) can be skipped to do amplification-free or Direct-cDNA sequencing. PCR amplification leads to PCR bias, and in cases where that is unacceptable, direct cDNA is the way to go. Again, as this is DNA sequencing, RNA modification information is lost. Poly(A)/(T) tails are, however, still intact and their length can be estimated.

	Direct RNA sequencing	PCR-amplified cDNA sequencing	Direct cDNA sequencing
Amount of starting RNA	500ng	4ng	100ng
Sequencing speed	70bps	450bps	450bps
Detection of RNA base modifications	Possible	Not possible	Not possible
Basecalling accuracy	93.6%	99.9%	99.9%
Supports full-length reads	Yes	Yes	Yes
Typical number of reads	1 million full-length per flow cell on MinION/GridION	5-10 million full-length per flow cell on MinION/GirdION	5-10 million full-length per flow cell on MinION/GirdION
Poly(A) tail profiling	Possible	Possible	Possible
Poly(T)-tail profiling	There are no poly(T) tails in RNA	Possible	Possible
Cap type determination	Possible	Not possible	Not possible

Tab. 2.1. **Comparison of different methods for sequencing RNA on Nanopore.**

2.5 Nanopore output format (FAST5)

An RNA strand translocating through a Nanopore modulates the current passing through the pore. This current is analog in nature and is sampled by an analog-to-digital converter at a rate of 3012 samples/nt for RNA and 4000 samples/nt for DNA.

These series of samples or current measurements for each read are stored in an array in a FAST5 file which uses Hierarchical Data Format (HDF5) in the backend.

FAST5 files come in two flavors: single-read FAST5 files and multi-read FAST files. A single-read FAST5 file contains data for just one read, whereas a multi-read FAST5 file packs information about multiple reads (default: 1000 reads). A dataset stored as multi-read FAST5 files has a smaller footprint on disk compared to one with single FAST5 files and is also faster to subsequently process due to less I/O overhead.

The FAST5 files produced by Nanopore's data acquisition software (MinKNOW) are called raw FAST5 files as they contain only raw signal information (Fig. 2.2a). The raw signal measurements when plotted show the fluctuations in pore current when the RNA strands passed through the pore and is commonly referred to as a squiggle.

2.6 Basecalling — with Guppy

To find the sequence of bases encoded in the raw Nanopore signal, it must be basecalled with a basecaller. The production basecaller provided by ONT is called Guppy. For RNA basecalling, Guppy provides a choice between two basecalling models: 1) a fast basecalling model that is fast to basecall with but has lower accuracy (88.6%) and 2) A high-accuracy basecalling model which is slower to basecall with but has high accuracy (93.9%). We use the high-accuracy model for basecalling all our datasets.

2.6.1 Move and Trace tables in basecalled FAST5 files

Guppy uses raw FAST5 file produced by sequencing in input and outputs another set of basecalled FAST5 files which have three additional important pieces of information (Fig. 2.2b):

1. *The basecalls: Present in the form of a **FASTQ** field*
2. *The mapping between signal and basecalls: Present in the form of a **Move** table.*

Guppy produces a basecall prediction for every 10 samples (`block_stride`) of the raw signal (Fig. 2.2c and d). This 10–sample window is called an event. Some events may have a new basecall prediction, while other events may have the same base

persisting from the previous event; this is because some bases may spend more time in the pore, and consequently, more than 10 samples can be acquired for such bases.

If there is a new basecall prediction in an event, the Move table contains a 1 for such an event, and if the base from the previous event is persisting in the new event, the Move table will have a 0 corresponding to the new event. By correlating the basecalls in the FASTQ to the events in the Move table, we can map out what base was predicted for which stretch of the Nanopore signal (Fig. 2.2, f and g).

As events contain 10 samples, the mapping between the signal and the basecalls has a granularity of 10 samples. In older versions of Guppy (v3.4.3), this granularity was 15 samples, which has been reduced to 10 samples in the latest version (v6.0.1). The lesser the granularity, the better, as more precise boundaries between neighboring bases can be obtained. However, this granularity is baked into the trained basecalling model from ONT and the end-user cannot lower it.

3. *The base probability of each of the four bases: Present in the **Trace** table*

   The Trace table contains eight columns: The first four columns are the A, C, G, U (in this order) flip probabilities, and the next four columns are the A, C, G, U (in this order) flop probabilities. This table has one row for each of the events in the Move table. The probability values — encoded as a number between 0–255 — represent the basecaller's confidence in calling a particular base in any event. A lower value represents the lower confidence of the basecaller in calling a base, and vice versa.

   The reason for having flip and flop probabilities (8 columns in total) is because a mechanism is needed to distinguish between bases in homopolymer stretches, i.e., when one base in a homopolymer stretch ends and the other one starts. In a homopolymer, the flip and flop probabilities switch every time the basecalling algorithm thinks a new base has started; so, in essence, the transition point where the flip and flop probabilities switch (i.e. change level), indicates the end of one homopolymer base and the start of the next homopolymer base (see arrows in Fig. 2.3).

   A modified base results in the splitting of the base probabilities among the possible bases indicating the uncertainty that the basecaller has about the real identity of this base (see base probabilities for modified G [mG] in Fig. 2.3). The splitting of base probabilities among the four bases when a modified base is encountered by the basecaller can be a strong feature that can be used by a classifier for detecting modified bases.

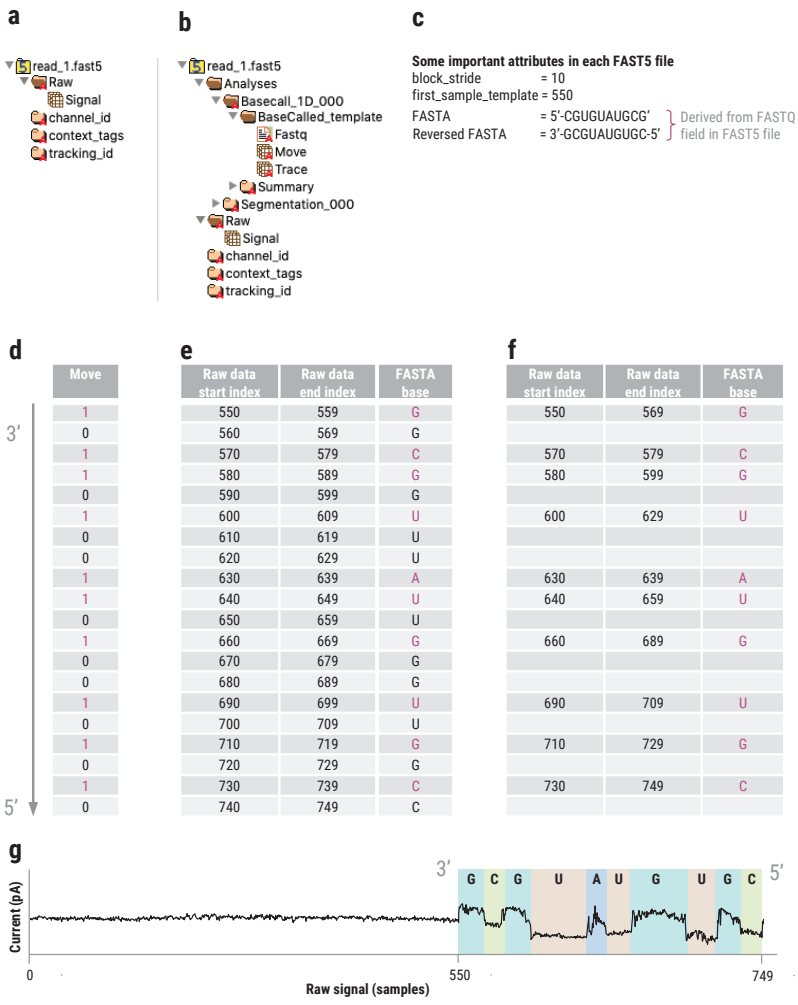


Fig. 2.2. **Nanopore FAST5 file structure information. a)** A Raw FAST5 file contains only the raw signal. **b)** A basecalled FAST5 file contains additional information such as the FASTQ, Move and Trace tables, and **c)** various attributes most important of which are the `block_stride` and the `first_sample_template`. **d)** The Move table in which each row is an event. Each event corresponds to a stretch of raw signal equal to the value of `block_stride`. A move of 1 represents the detection of a new base in that event, while a move of 0 represents that no new base has been detected and that the base from the previous event is still persisting in the current event. **e)** By using the Move table and information `block_stride` and `first_sample_template` attributes, it is possible to create a mapping between the raw signal samples and the basecalled sequence. **f)** Simplified form of the table in e. **g)** Raw data start and end indexes in table f can be used to annotate the raw Nanopore squiggle with base predictions. In this way, one-to-one mapping between the raw signal and basecalls can be obtained.

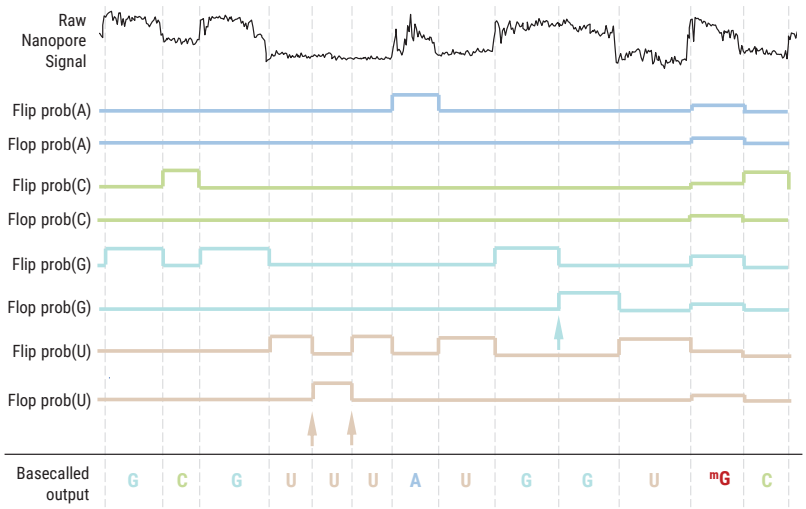


Fig. 2.3. **A plot of the Trace table against the Nanopore signal and the basecalled sequence.** The flip and flop probabilities switch (vertical arrows) whenever the basecaller thinks that one base within a homopolymer has passed and a new homopolymer base has started to pass through the pores constriction. If a modified base passes through the pore (for example ^{m}G), then the base probability gets split up between all the bases rather than being high only for one base. This is a tell-tale sign that the base passing through the pore is different (in other words modified) compared to the bases that the basecaller was trained on.

2.7 Basecalling — with Bonito

ONT also has an experimental basecaller called Bonito (`https://github.com/nanoporetech/bonito`) which basecalls with higher accuracy than the production basecaller Guppy. However, at the time of writing, it works only on DNA, and additionally, it cannot produce FAST5 files as the output (FAST5 files have additional information which helps us to annotate the raw signal with basecalls). Bonito can produce only a FASTQ file. This means that mapping information between signal and basecalls, and Move or Trace tables, are not available if Bonito is used to basecall the data.

2.8 Alignment

Once Nanopore data has been basecalling, often the next task is to align it to the reference genome or the reference transcriptome. Correct alignment of reads to the reference is quintessential for all the downstream processing steps that build upon it.

Short-read alignment tools such as BWA-MEM cannot handle Nanopore's high error-rates and longer read lengths. With the advent of Nanopore sequencing, many new alignment tools have come out but none has been as successful as Minimap2. Minimap2 [30] is now the *de facto* tool used for alignment of Nanopore data. It comes with presets to do spliced alignment of RNA to reference genome (`-ax splice -uf -k14`), or unspliced to a reference transcriptome (`-ax map-ont`). It also provides a convenient python API called *mappy* that can be used to programmatically align a read to its respective reference — a feature that we use heavily in this work (in Chapter 3) to align each read to its custom-made reference during cap-type decoding.

3

Single-molecule level prediction of mRNA cap types with *capable*

Summary: *mRNA caps are 5'-terminal modifications that happen co-transcriptionally on Pol II–transcribed transcripts in eukaryotes. These caps can be of different types depending on the modifications present on the first and second transcribed nucleotides of the mRNA. Existing methods for the detection of these modifications operate at a bulk level and give only a relative abundance estimate of different caps in an RNA sample. To understand the role of different cap types at a single-molecule level, I have developed capable — a software tool that predicts cap types on each individual mRNA transcript. Different mRNA caps produce different signatures when sequenced through a Nanopore. Capable uses machine learning to learn these signature features from training data produced by sequencing synthetically-made cap standards. The trained model can then be used to predict the cap types in different biological mRNA samples at the single-molecule level. The work presented in this chapter is still in progress due to an unprecedented level of challenges involved in creating ground-truth data for training the classifier. This chapter details all these challenges and our strategy going forward.*

3.1 Introduction

RNA capping is an evolutionarily conserved modification in eukaryotes that happens co-transcriptionally at the 5'-end of Pol II-transcribed transcripts. It is the very first modification on mRNA and happens when the growing RNA transcript is about 25–30 nucleotides long [31].

During RNA capping, an inverted methylated guanosine base (m^7G) gets prepended to the 5'-end of the nascent transcripts by a capping complex [32]. Two enzymes — RNA guanyltransferase and 5'-phosphatase (RNGTT) and guanine-N7 methyltransferase (RNMT) — play a key role in a three-step RNA capping process [33] (Fig. 3.1). RNGTT has two active domains: an RNA triphosphatase (TPase) domain and a guanosine triphosphate (GTP) domain. In the first step, the RNA triphosphatase

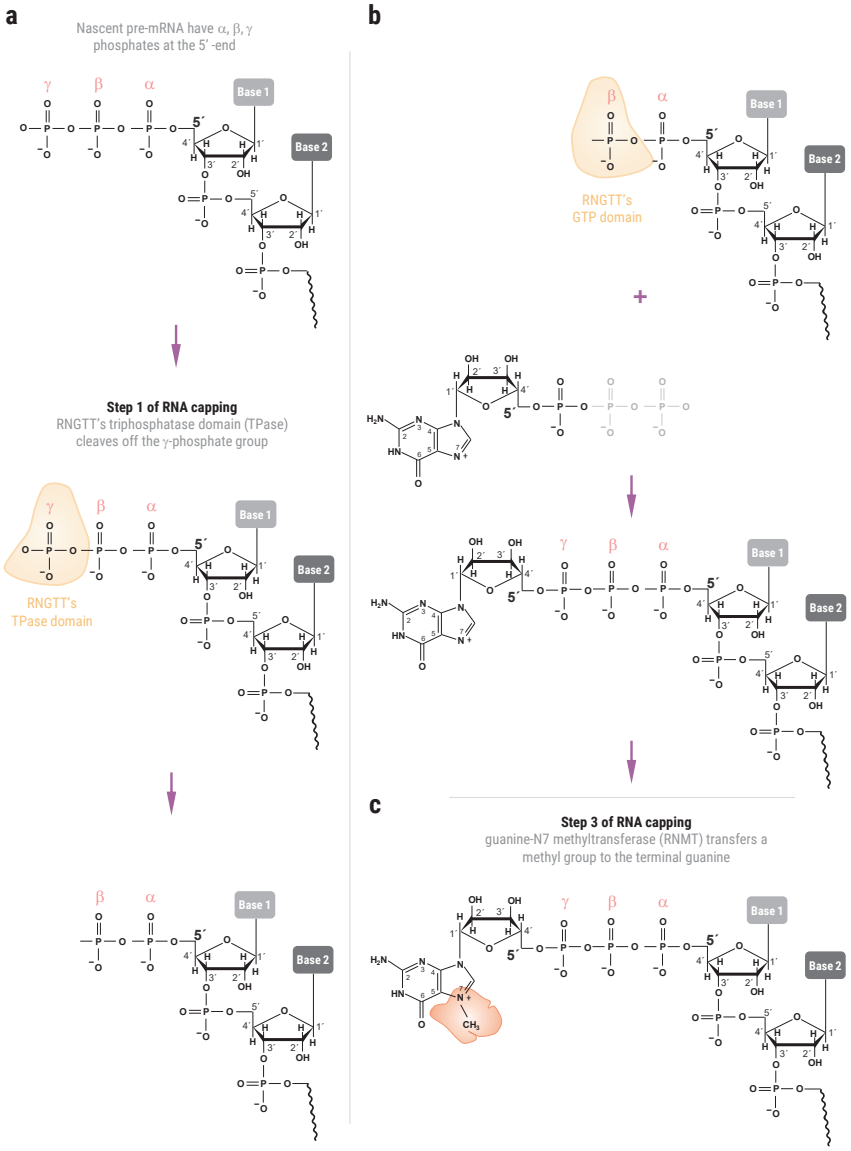


Fig. 3.1. **RNA capping mechanism. a)** Step 1: RNGTT's TPase domain cleaves off γ-phosphate at the end of the nascent pre-mRNA. **b)** Step 2: RNGTT's GTP domain adds guanosine monophosphate to the end of RNA (produced in the first step 1). **c)** Step 3: RNMT methylates terminal guanine.

(TPase) domain cleaves off the γ-phosphate group from the 5'-end of the nascent pre-mRNA after which the second domain RNA guanyltransferase (GTase) converts a guanosine triphosphate (GTP) to guanosine monophosphate (GMP), inverts it, and transfers it to the end of the RNA (Fig. 3.1). Lastly, RNMT imparts a methyl group at N7 of the terminal guanine base. The RNA now has an inverted methylated guanosine base (m^7G) attached to the rest of the RNA. This type of cap structure is called cap0 and can also be written as m^7Gppp–RNA, where N represents the first transcribed nucleotide.

The bond between the terminal m^7G and the first transcribed nucleotide in cap0 is an unusual 5'–to–5' linkage (Fig. 3.1). This 5'–to–5' link only happens at the cap, and nowhere else is the RNA as all the rest of the nucleotides in RNA are connected with 5'–to–3' linkages. The inverted nature of the attachment of terminal m^7G makes the capped RNAs inert to degradation by exoribonucleases, and also — as we will see in later sections — makes it difficult to ligate anything to the capped 5'-end of the RNA.

In addition to cap0, other caps are also known to exist (see Fig. 3.2). The cap methyltransferase 1 (CMTR1) enzyme uses cap0 transcripts as substrate and methylates the hydroxyl group present at carbon 2 of the ribose sugar backbone of the first transcribed nucleotide. This modification is known as 2'-O methylation or Nm modification as it can be present on ribose sugar of any base (N). The resulting cap is called a cap1 (m^7GpppNmp–RNA). The 2'-O methylation can happen on the ribose sugar backbone of any of the four nucleotides, and hence an N in the notation m^7GpppNm. If the first transcribed nucleotide is an adenosine, then an enzyme called cap adenosine N6-methyltransferase (CAPAM) can transfer a methyl group at position 6 of the adenine base to result in a cap1-m^6Am (m^7Gpppm^{6}Amp–RNA)[8] or a cap0-m^6A (m^7Gpppm^{6}Ap–RNA). An additional enzyme — cap methyltransferase 2 (CMTR2) — can methylate the second transcribed nucleotide to result in cap2 (m^7GpppNmpNmp–RNA). It is not known whether cap2,-1 (m^7GpppNpNm) — a cap containing 2'-O methylation on the second transcribed nucleotide but not on the first — exists in biology or not. There is a possibility that cap2,-1 exists because the presence of cap1 2'-O methylation only enhances the activity of CMTR2, but CMTR2 does not need methylations of cap0 or cap1 to methylate the second transcribed nucleotide [4]. Additionally, while cap0 and cap1 methylations happen in the nucleus, cap2 methylation happens in the cytoplasm [34].

The terminal m^7G in cap0 can be further methylated by trimethylguanosine synthase 1 (Tgs1) to form the trimethylguanosine (TMG) / $m_3{}^{2,2,7}G$ cap. These caps are predominantly found on non-coding RNAs such as small nuclear RNAs (snRNAs),

small nucleolar RNAs (snoRNAs) and telomerase RNAs, but have also been detected on protein-coding mRNAs, specifically in subsets of selenoprotein mRNAs [35]. TMG capping takes place both in the nucleus (for snoRNAs) and the cytoplasm (for snRNAs). These mRNAs appear to retain their ability to recruit ribosomes and be translated despite the TMG cap having a low affinity to the cap-binding protein eIF4E [36], potentially as a result of cap-independent translation initiation. Whether TMG caps are present on other mRNAs is unknown.

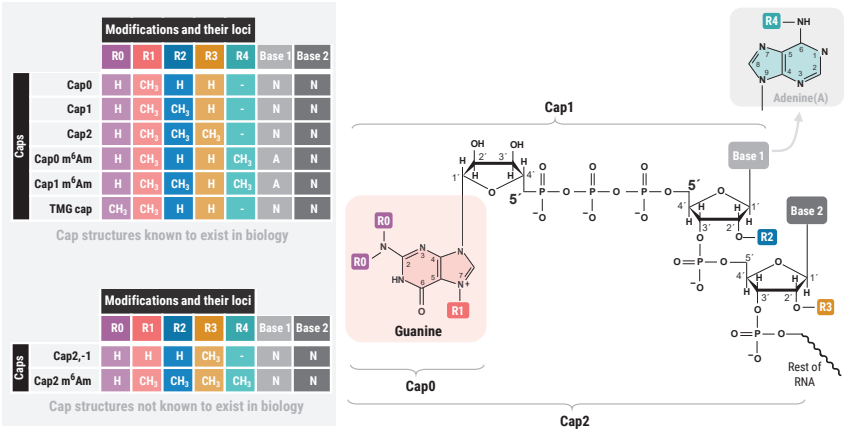


Fig. 3.2. **Chemical structure of different canonical cap structures in mRNA.** The bond between terminal m^7G and the first transcribed nucleotide is formed due to an unusual 5'-to-5' linkage. This 5'-to-5' linkage only happens at the cap and nowhere else is the RNA as all the rest of the nucleotides in RNA are connected to each other with 5'-to-3' linkages. Some canonical are known to exist in biology (top table), while others are not yet known to exist (bottom table)

The caps discussed so far all have a terminal m^7G and these caps are collectively referred to as canonical caps. Recently, a non-canonical (NC) class of caps has been discovered in eukaryotes. These caps have a metabolite effector instead of the terminal m^7G (see Fig. 3.3). Unlike m^7G caps, which are added during transcription, the non-canonical caps initiate the transcription by serving as a non-canonical initiation nucleotide (NCIN). Moreover, the non-canonical cap is added by the RNA polymerase itself in contrast to the canonical caps in which the m^7G cap is added by the capping complex.

Two of the most well-known NC caps are the NAD^+ and NADH caps formed using the oxidized and reduced forms of nicotinamide adenine dinucleotide (NAD), respectively. Other NC caps include the flavin adenine dinucleotide (FAD) caps, uri-

dine diphosphate glucose (UDP-Glc), and uridine diphosphate N-acetylglucosamine (UDP-GlcNAc) [37]. Many more non-canonical caps such as those containing the different variations dinucleoside polyphosphates (Np_nNs) have been found to exist in bacteria [38].

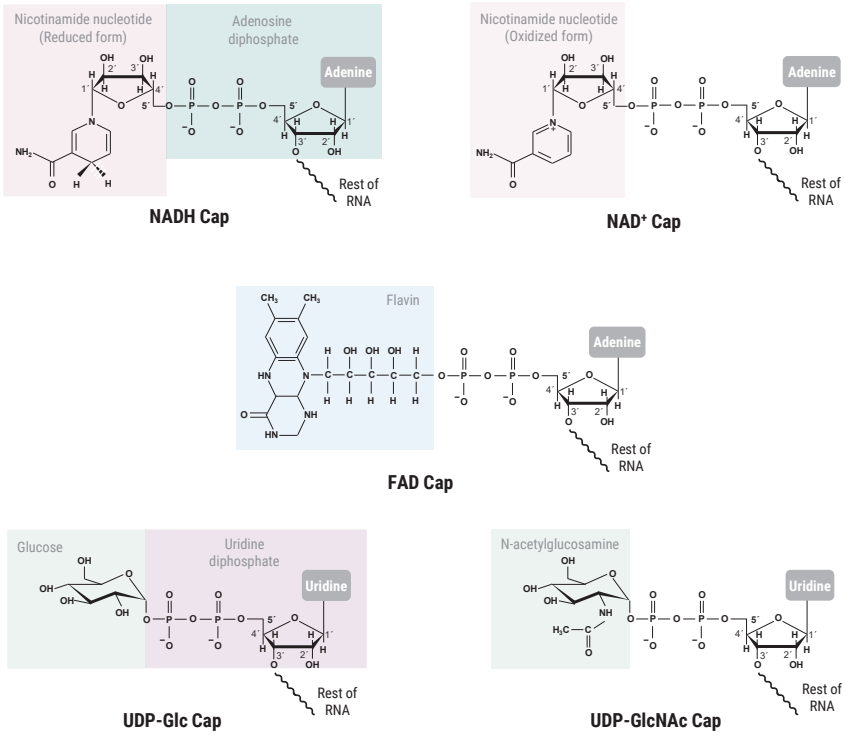


Fig. 3.3. **Chemical structure of eukaryotic non-canonical initiating nucleotide (NCIN) caps.**

3.2 Biological role of caps

3.2.1 Cap 0

The m^7G of the minimal cap structure cap0 plays a major role in various processes that happen during the life-cycle of mRNA. In the nucleus, the cap-binding complex binds to the m^7G to prevent degradation of the RNA and to facilitate the export of

the mRNA from the nucleus to the cytoplasm [39]. In the cytoplasm, the translation initiation factor eIF4E binds to m^7G and helps in the circularization of RNA into a loop for efficient translation of the mRNA into protein by ribosomes [40]. The m^7G also serves as a binding site for decapping enzymes that degrade the mRNA once it is no longer needed [41].

3.2.2 Cap 1

The 2'-O methylation in cap1 is used by a cell to differentiate its own (self) from viral RNA. Viral RNAs with 5'-ppp and double-stranded blunt ends, serve as a ligand for Retinoic Acid Inducible Gene-I (RIG-I) — a cytosolic innate immune receptor that can distinguish cellular self RNAs from pathogenic non-self RNAs. Once RIG-I is activated, it triggers a signaling pathway that leads to Type-I interferon (IFN) production which ultimately destroys the viral RNA. It has recently been shown that cap0 double-stranded RNA activates RIG-I, but cap1 double-stranded RNA does not [42]. Thus the 2'-O methylation of cap1 abrogates RIG-I activation. Many viruses have evolved a mechanism to cap their genomes and/or transcripts with cap1 [43] or snatch them from the host RNA (cap snatching) [44], both of which help them evade the immune response by preventing recognition by RIG-I.

3.2.3 Cap 2

There is no consensus yet on the role of cap2 methylations. Cap2 methylations are reported to present in as much as 50% of the transcripts [45]. Furthermore, cap2 mRNA has been found to be 3-fold more enriched in polysomal fractions compared to non-polysomal fractions, whereas the amount of cap1 transcripts was the same in both fractions [4]. This indicates that cap2-capped mRNA may have an increased affinity for ribosomes, or alternatively, methylation of cap2 occurs after the ribosomes bind to the mRNA. Recently, it has been found that methylation of second transcribed nucleotide in cap2 impacts protein production level in a cell-specific manner and contributes to RNA immune evasion [6].

3.2.4 Cap m6Am

The N6 methylation in m^7Gpppm^6Amp–RNA transcripts increases the stability of the RNA against decapping when compared to $m^7GpppAmp$–RNA transcripts [46]. The

half-life of m^7Gpppm^{6}Amp–RNA transcripts is 2.5 hours longer than m^7GpppNmp–RNA transcripts. Furthermore, the N6 methylation in m^7Gpppm^{6}Amp–RNA transcripts makes them less susceptible to microRNA-mediated degradation [47].

3.2.5 TMG cap

The TMG cap modifications are highly conserved in eukaryotes. TMG caps are believed to be necessary for the snRNAs to fulfill their cellular functions [48]. TMG-capped ncRNAs have also been found to have higher expression levels compared to snRNAs lacking TMG-caps [49].

3.2.6 NAD+ and NADH caps

These caps have been found in bacteria and yeast [50, 51], and more recently in humans [52] and plants as well [53]. NAD-capped transcripts constitute a small proportion of the total transcript pool from any gene, but they are enriched in the polysomal fraction and associate with the translating ribosomes [53]. In mitochondria, NAD^+-capped RNA levels can reach up to 60% of mitochondrial transcripts[54]. NAD gets incorporated in mRNA transcript by Pol II in a largely statistical manner that reflects the competition of NAD with the canonical initiator ATP [55]. Unlike canonical caps which impart stability to their respective transcripts, NAD-caps have been shown to promote the decay of their respective transcripts [52]. Additionally, NAD-capped transcripts are, on average, shorter than non-NAD-capped transcripts, and are also not translatable in vitro [55].

Whether NAD^+-capped transcripts are capable of being translated is still somewhat unclear. The caps are present in mRNAs that are both spliced and poly-adenylated [52, 51], but these appear unable to be translated during in vitro translation experiments [55]. In contrast, polysome fractions from A. thaliana found an enrichment of NAD^+-capped mRNAs associated with translating ribosomes [53]. Whether translation of NAD^+-capped transcripts is particular to certain cells or species is therefore unknown, but it is possible that these transcripts could only be translated under specific circumstances and potentially make use of cap-independent translation through e.g., internal ribosome entry sites (IRES).

3.2.7 FAD caps

FAD caps appear to be enriched in shorter RNAs (<200 nt) [56] and can be decapped (deFADed) by Nudt12 and Nudt16 [57]. The nature of RNAs capped with FAD caps is unknown because we currently do not have any method that can specifically enrich transcripts carrying these caps [56].

3.2.8 UDP-Glc and UDP-GlcNAc caps

These uridine-containing NCINs compete with uridine triphosphate (UTP) for use by RNA polymerase as initiating nucleotides. The UDP-GlcNAc caps may be among the most abundant non-canonical caps, even more than NAD^+, and have been shown to respond to oxidative and alkylation stresses in yeast [58]. However, no enzymes involved in its processing have been discovered and no hypotheses as to its specific function have been forwarded.

Nothing is currently known about the role of UDP-Glc caps.

3.3 Existing methods for cap type prediction

There are many methods for cap structure determination — each with its own strengths and weaknesses — and can be broadly classified into four main categories, as described below:

3.3.1 Radio-isotope labeling-based method

This method relies (Fig. 3.4) on incorporating a radioactive ^{32}P in the RNA cap by transcribing the total RNA in a cell extract in the presence of one or more radioactively-labeled NTPs (such as α-^{32}P[ATP]) [59]. Once all the caps have a radioactive label, a nuclease T2 treatment in the presence of alkaline phosphatase cleaves them off from their respective RNAs. Nuclease T2 cannot cleave the phosphodiester bond of the ribose sugar with 2'-O methylation. It, therefore, cleaves the phosphodiester bond of the first non-methylated base 3' of the 2'-O methylated base. Thus m^7Gppp^*NmpNp is produced from cap1 transcripts, and $m^7Gppp^*NmpNmpNp$ is produced from cap2 transcripts.

The cleaved caps are separated on DEAE-cellulose paper electrophoresis which is then placed against an x-ray film. White bands form in the developed x-ray film in regions where the radioactive caps had migrated and settled. Different caps migrate different distances and this is used to resolve between the different cap types (Fig. 3.4).

Fig. 3.4. **Radio-isotope labeling-based method for quantifying different cap types in a sample**

This method is technically challenging and requires the use of radioactively-labeled nucleotides that have the potential to create cellular toxicity artifacts [60]. Furthermore, although this method is sensitive, it lacks specificity [34].

3.3.2 Mass spectrometry-based methods

Due to the low resolution of radio-isotope labeling-based methods, newer methods use LC-MS to resolve the different caps better. One of the most recent examples using this technique is CAP-MAP (Cap analysis protocol with minimal analyte processing) [61]. This method uses oligo(dT) affinity beads first to enrich poly(A)-tailed and capped RNAs. Next, a Nuclease P1 treatment hydrolyzes the phosphodiester bonds in the RNA yielding m^7GpppNm dinucleotides and nucleotide 5-'monophosphates. The sample is then passed through a porous graphitic carbon column coupled to a triple quadrupole mass spectrometer operating in negative ion mode and programmed to detect only the cap dinucleotides. The mass spectrometer then gives abundance estimates of different cap-dinucleotides in the sample (Fig. 3.5).

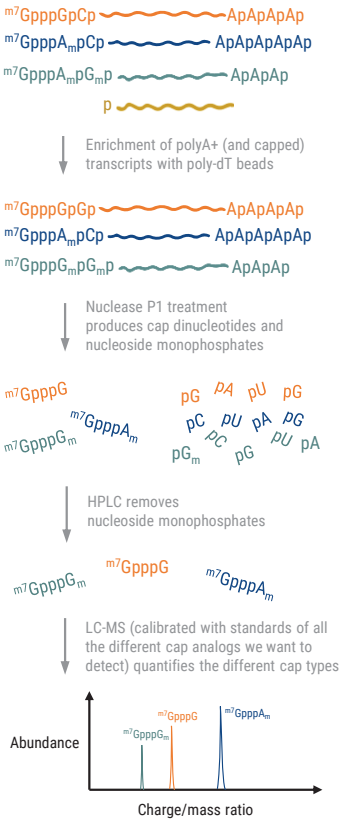


Fig. 3.5. Mass-spectrometry based method for quantifying different cap types present in a sample

Similar approaches for cap quantification include CapQuant [58] and LC-QqQ-MS [62].

A major flaw of all these methods is that they cannot assess the 2'-O methylation of the second transcribed nucleotide. Only the m^7G cap and the first transcribed nucleotides (i.e., $m^7GpppNm$) can be investigated with these methods. This is partly due to the exponentially large number of standards required to calibrate the MS and the difficulty and costs involved in synthesizing these different standards.

3.3.3 NGS-based method

One of the most recent methods – CapZyme-Seq – quantifies the relative abundance of NCIN-capped and uncapped reads in a sample using NGS [63]. The sample is first aliquoted into two: The first aliquot is treated with Rai1/NudC enzyme that cleaves off the NCIN-cap from the NCIN-capped transcripts leaving behind a phosphate group on these transcripts; this treatment does not modify the uncapped transcripts with triphosphate ends or the canonical capped mRNAs. The second aliquot is treated with Rpp which cleaves the phosphodiester bond between α and β phosphates of uncapped transcripts leaving behind a monophosphate; this treatment does affect the NCIN-capped transcripts. 5'-adaptors containing barcodes are added to both aliquots, followed by RT primer annealing, cDNA synthesis, and NGS sequencing. The NGS data from both aliquots are then sequenced with NGS to quantify the levels of NCIN-capped and non-capped transcripts (Fig. 3.6).

While this method can identify transcripts and transcript start sites associated with non-canonical caps, it cannot distinguish as to which transcript had which particular non-canonical cap type.

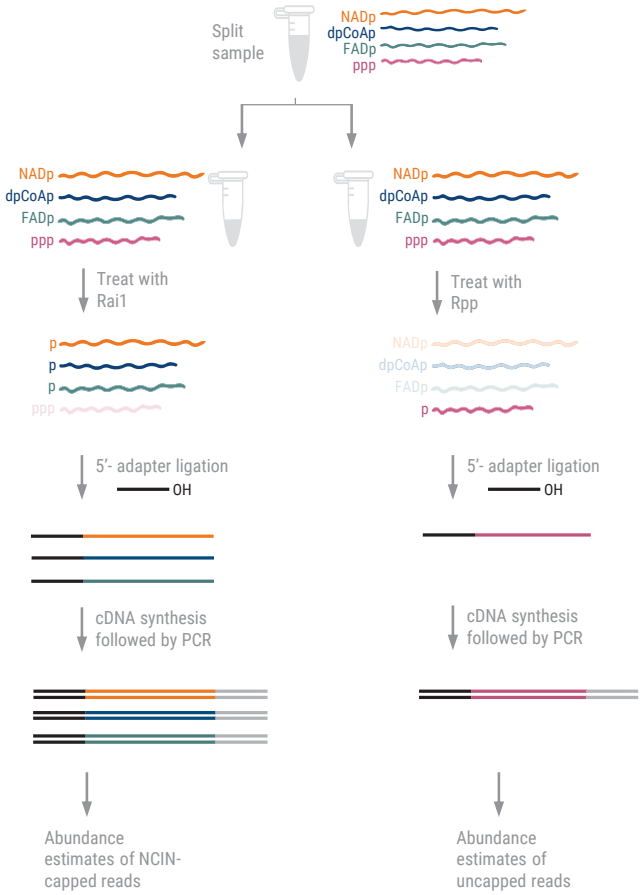


Fig. 3.6. **NGS-based method for quantifying different cap types in a sample**

3.3.4 Nanopore sequencing-based method

Recently, NAD tagSeq II was developed to identify NAD transcripts using Nanopore sequencing [64]. Briefly, the approach specifically extends the 5'-end of only NAD transcripts with an oligonucleotide tag, while other transcripts remain intact. Nanopore sequence of the resulting library and identification of reads that carry the oligonucleotide tag reveals transcripts that originally were carrying NAD caps.

To tag NAD transcript with an oligonucleotide, the RNA is first treated with 3-azido-1-propanol in the presence of ADPRC (Fig. 3.7). The 3-azido-1-propanol replaces the nicotinamide of the NAD-RNA. The azide-functionalized NAD-RNA molecule is then ligated (through SPAAC) to a synthetic RNA oligonucleotide (tagRNA) carrying a DBCO group at its 3' end. This is followed by poly(A)-tailing of the library, followed by reverse transcription, and sequencing on a Nanopore. Only NAD transcripts have the tagRNA. Finding Nanopore reads with tagRNA at their 5'-ends help identify NAD transcripts.

The bulky click-chemistry involved in the junction between tagRNA and the NAD transcripts results in signal corruption of bases upstream and downstream of this junction, and therefore this method cannot accurately identify the transcription start sites.

3.4 Limitations of existing methods

The radio-isotope labeling-based methods and the mass spectrometry-based methods, can only quantify the relative abundance of different cap types in a given RNA sample. For these methods to work, the cap must be severed from their respective transcript before quantification, making it impossible to attribute the quantified caps back to their respective transcripts. Thus, isoform-level, or even gene-level, cap type prediction is not possible with these available methods. Furthermore, mass spectrometry-based methods cannot study cap2 structures because that requires a nuclease that does not cleave the link between the first and second transcribed nucleotides. To date, no such nuclease has been found. The Nuclease T2 can keep the link between first and second transcribed nucleotides intact, but it does not sever the cap after the second transcribed nucleotide, instead it severes the cap after the third nucleotide. This means by using nuclease T2, we would end up with m^7GpppNmNmNp fragments that we must quantify using mass spectrometry. However, to do so would require providing all the 256 different possible combinations

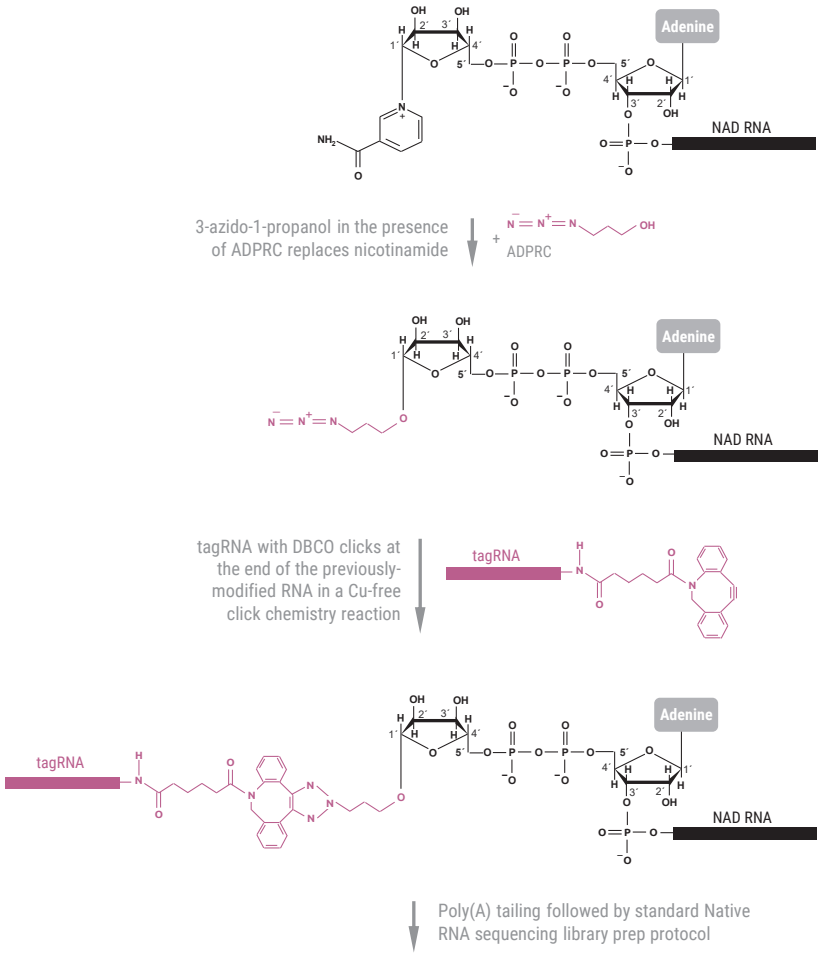


Fig. 3.7. **NAD tagSeq II protocol for studying NAD-capped RNA transcripts with Nanopore sequencing**

of m^7GpppNmNmNp (the 256 combinations are formed when A, C, G, and U are substituted for N and for different combinations for methylations on the first and second nucleotide) — and 32 different possible combinations for m^7GpppNmNp as standards to calibrate the mass spectrometer. Synthesizing such a large number of standards is a monumental task.

The NGS-based methods, such as CapZyme-Seq, can identify which transcript sequences are NCIN-capped and which of them are uncapped. However, it cannot tell which of the NCIN-capped transcripts have which particular non-canonical cap type. Furthermore, this method due to its reliance on Illumina sequenceing will only sequence a short segment of transcript proximal to the 5'-end of the transcripts. Such short sequenced fragments may not be enough to distinguish between transcript isoforms which may differ downstream of the sequenced fragments.

On the other hand, the latest Nanopore-based method, NAD tagSeq II, can only study NAD capped transcripts. If a transcript has a canonical cap, or has a non-canonical cap other than NAD cap, then such transcripts are impossible to study with this method due to its reliance on the transglycosylation reaction that can only work on the nicotinamide present in the NAD caps. While this method can sequence full-length NAD transcripts, around 20-30 bases are erroneously basecalled at the 5' of the transcripts due to the bulky click-chemistry corrupting the signal of these bases. Hence the 5'-end bases of the NAD-capped transcripts, which may carry crucial information about the promoter sequences, cannot be studied with this method.

In short, current methods cannot study all the different caps that might be present in a sample in a transcriptome-wide manner. This represents a major knowledge gap in our understanding of the transcriptome-wide role of all the different caps — a gap this thesis aims to bridge.

3.5 Nanopore sequencing of RNA cap and the challenges involved

During Nanopore sequencing of RNA, a motor protein feeds it in the 3'-to-5' direction through the pore at a slow and controlled speed (Fig. 3.8a). If there was no motor protein to control the speed of RNA, the RNA would, under the influence of applied voltage, pass through the pore at such a fast pace that it would not be possible to acquire enough current measurements per base to properly decode the translocated bases during basecalling. The motor protein, therefore, ensures that each translocating base spends a good amount of time in the pore so that enough current measurements can be recorded for accurate basecalling later on.

When the ratcheting motor protein reaches the 5'-end of the RNA, it loses its grip on the RNA and falls off from the RNA strand (Fig. 3.8b). Consequently, approximately 10-20 terminal nucleotides of RNA whiz through the pore at such a fast speed that

less than one current sample/nt is acquired. Such a small number of samples/nt are not enough to properly decode these bases. As a result, when the RNA reads are basecalled, approx. 10-20 nucleotides from their 5'-end — including the cap nucleotides — are always missing (Fig. 3.8c). Thus, the default RNA nanopore sequencing protocol cannot sequence the RNA caps.

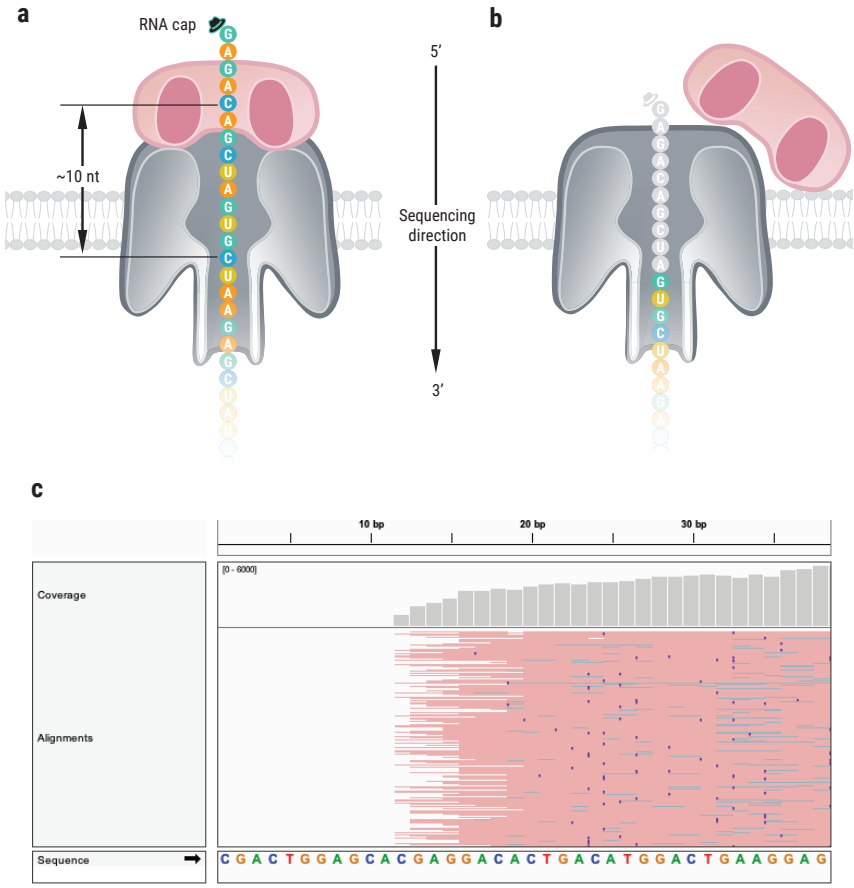
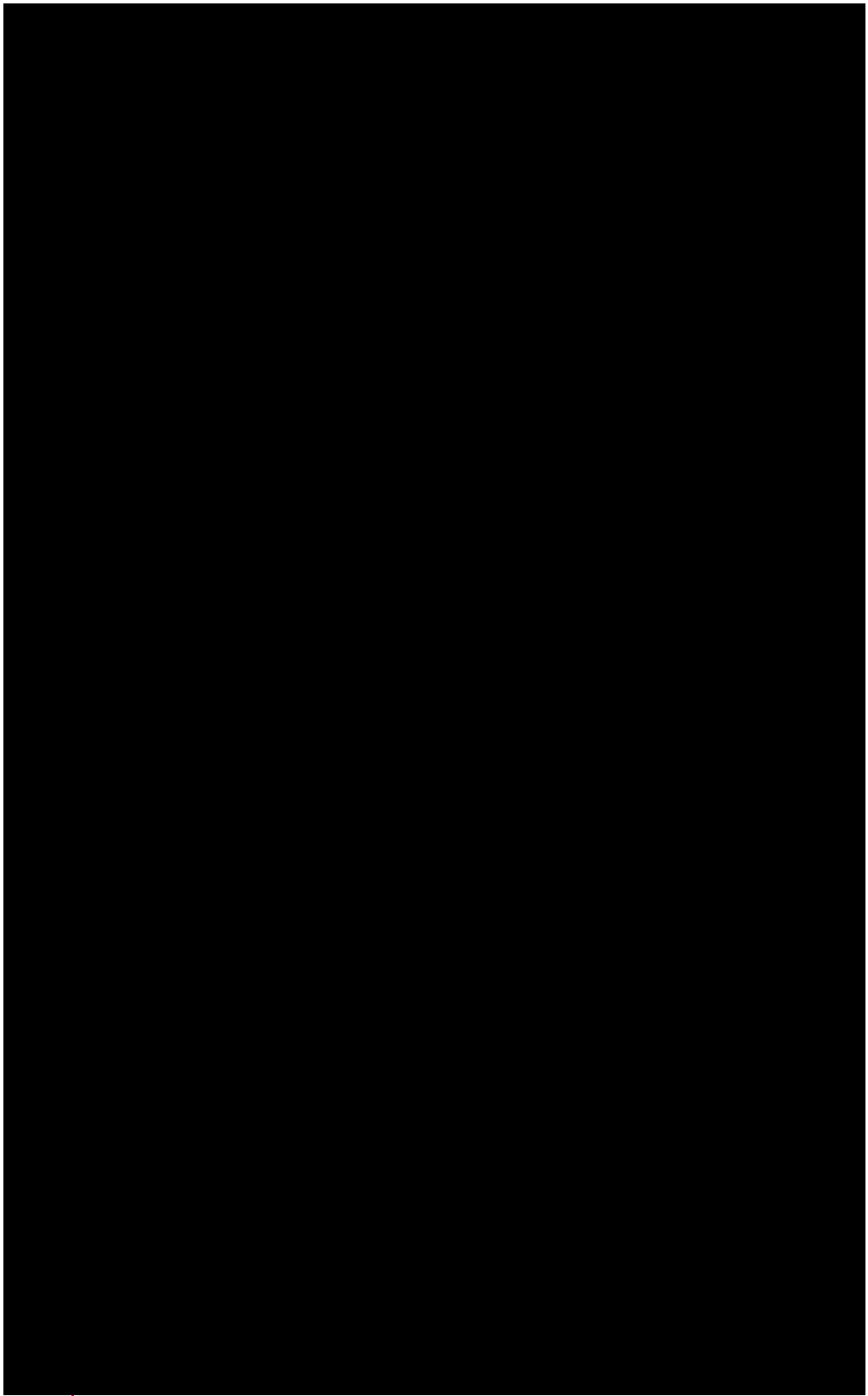


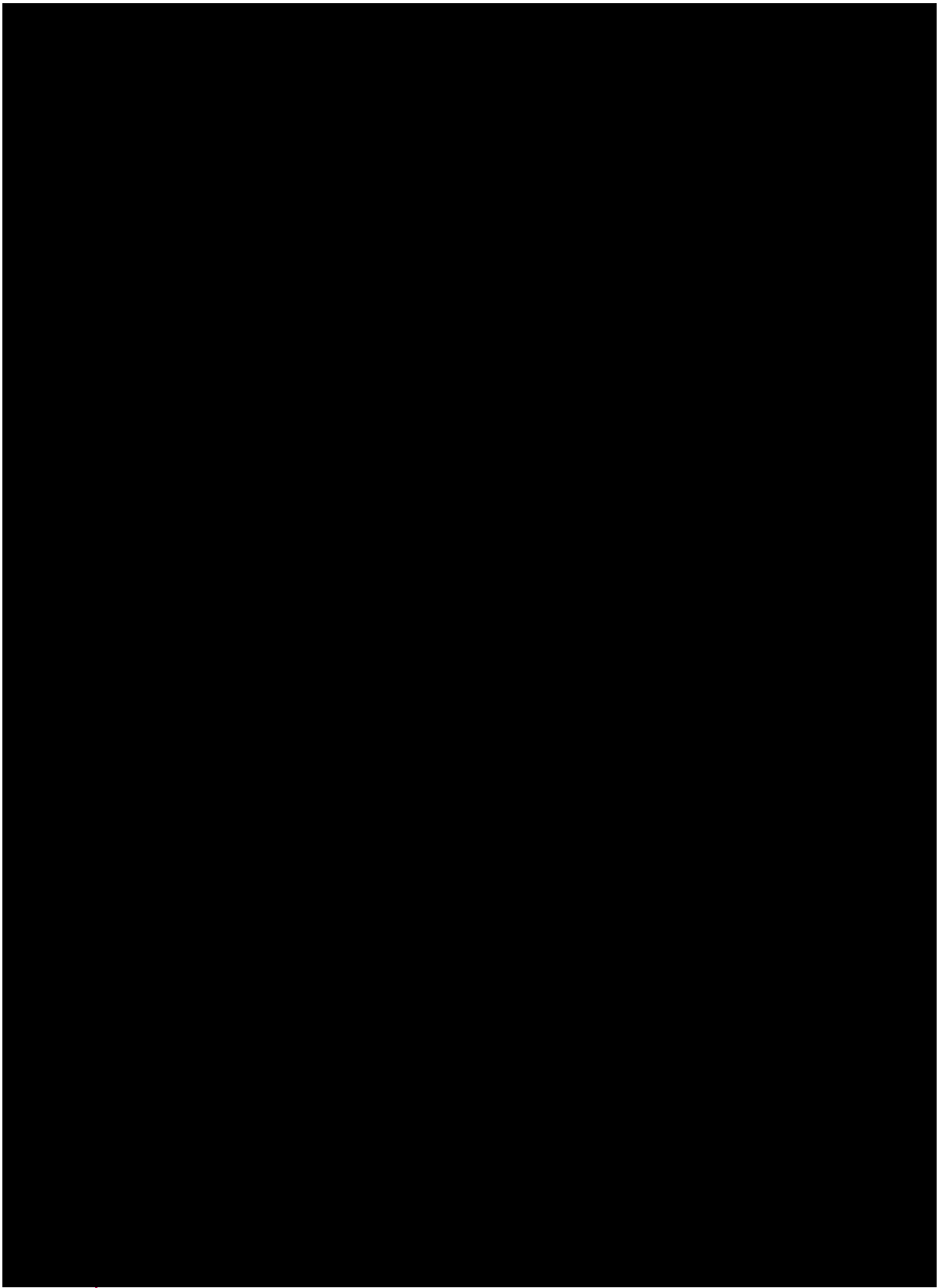
Fig. 3.8. **Loss of processive control of 5'-end of RNA during Nanopore sequencing. a)** The motor protein ratchets RNA at a slow controlled speed until it reaches the 5'-end of the RNA. **b)** When the sequencing reaches the very end of the RNA molecule, the motor enzyme can no longer grab onto the molecule and falls off. With the processive control of the motor enzyme now gone, the ten nucleotides (shown in gray) that still need to be sequenced, go through the pore so fast that their current signature is undecipherable. **c)** IGV view of the alignment of basecalled reads with the reference shows that 10-20 bases are mostly missing from the 5'-end.

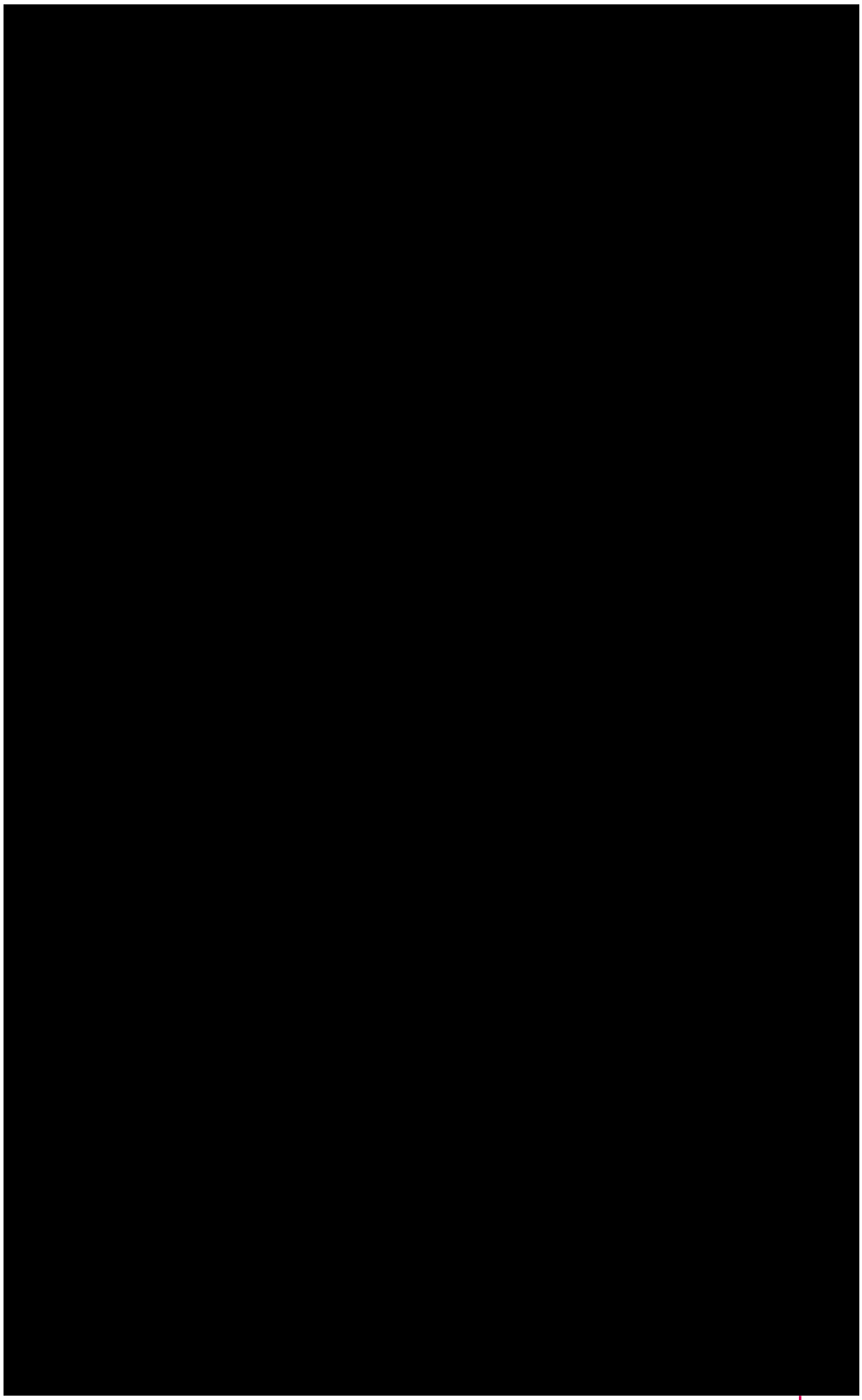
REDACTED

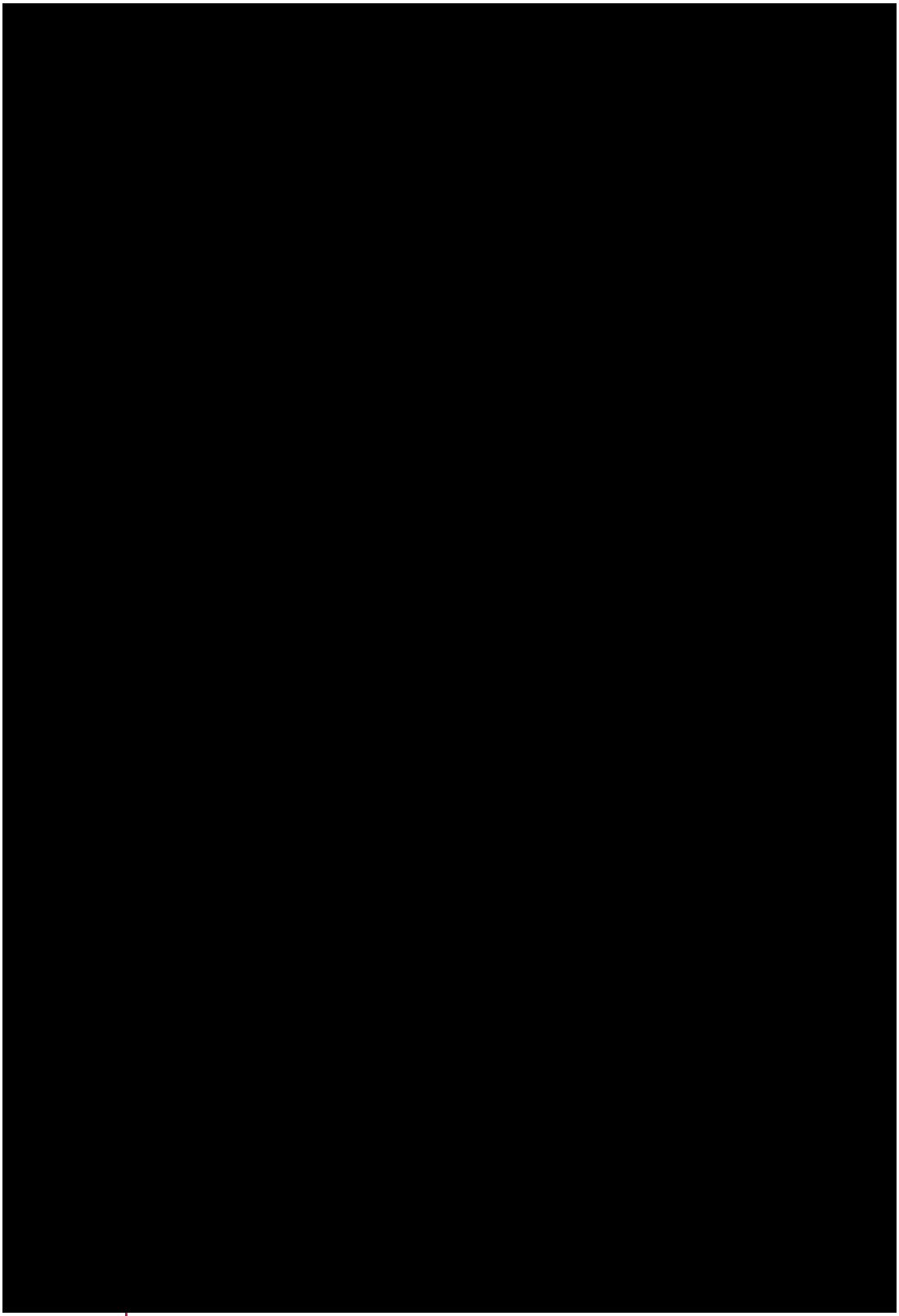


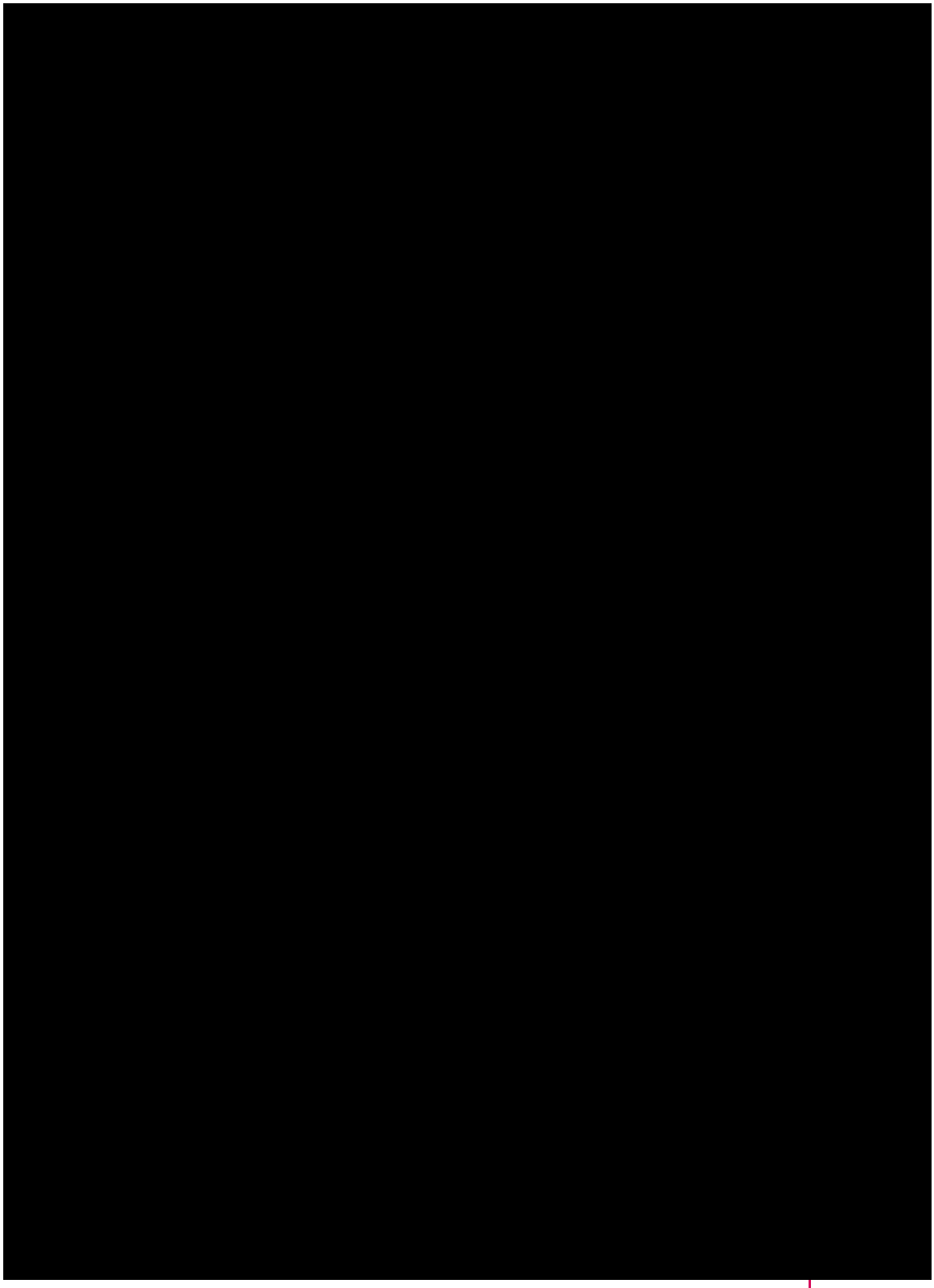
Fi

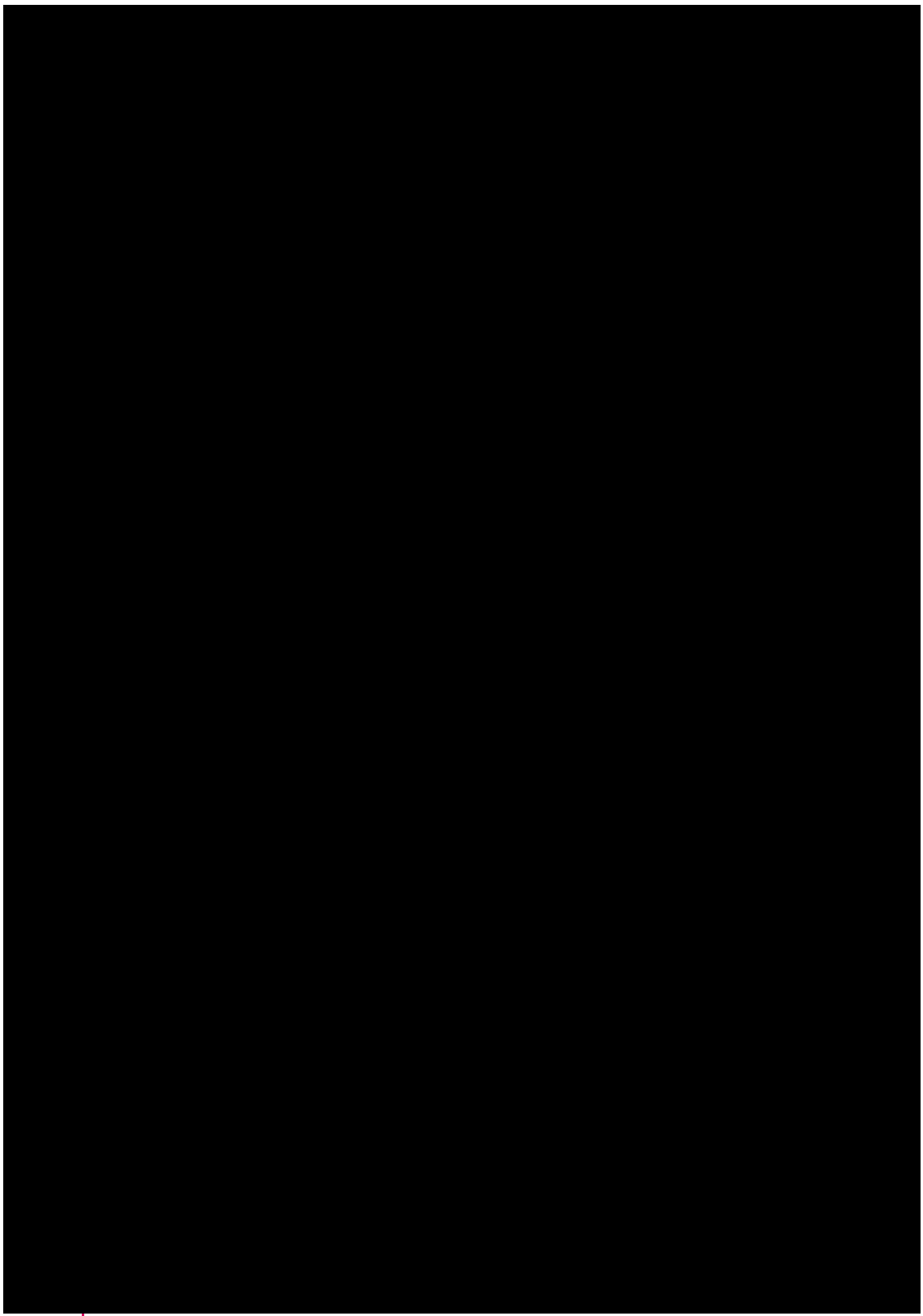
[redacted]

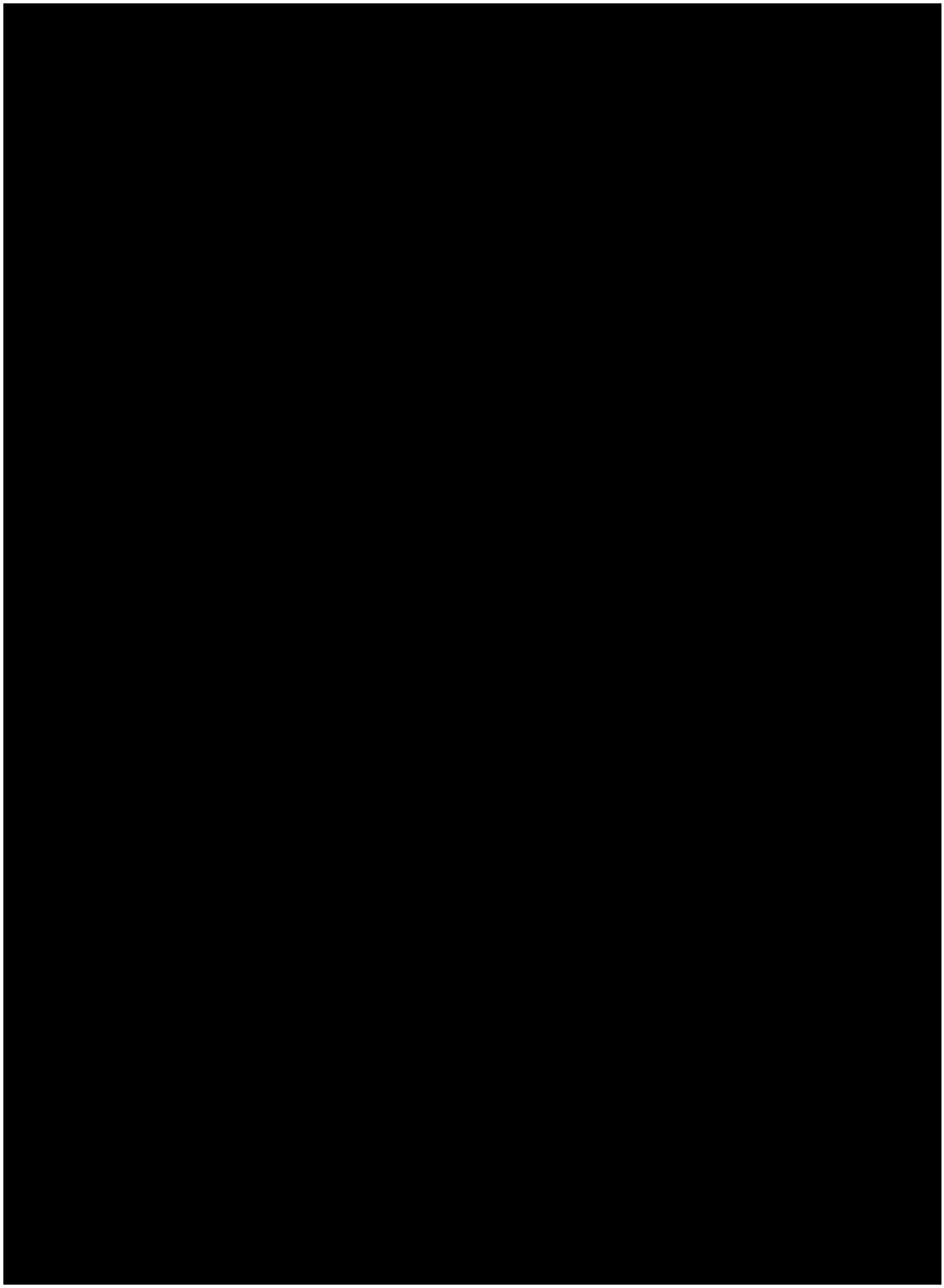


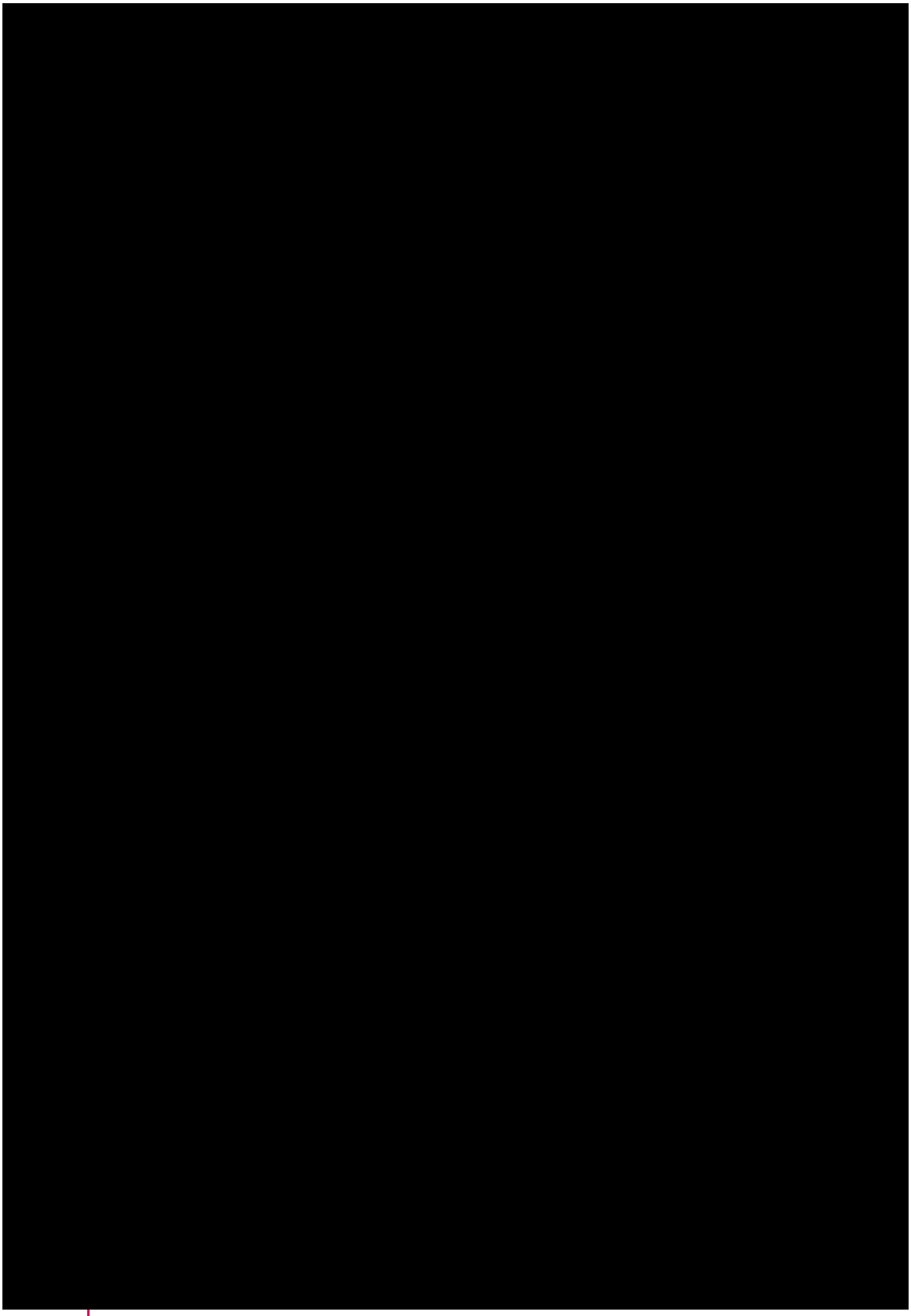


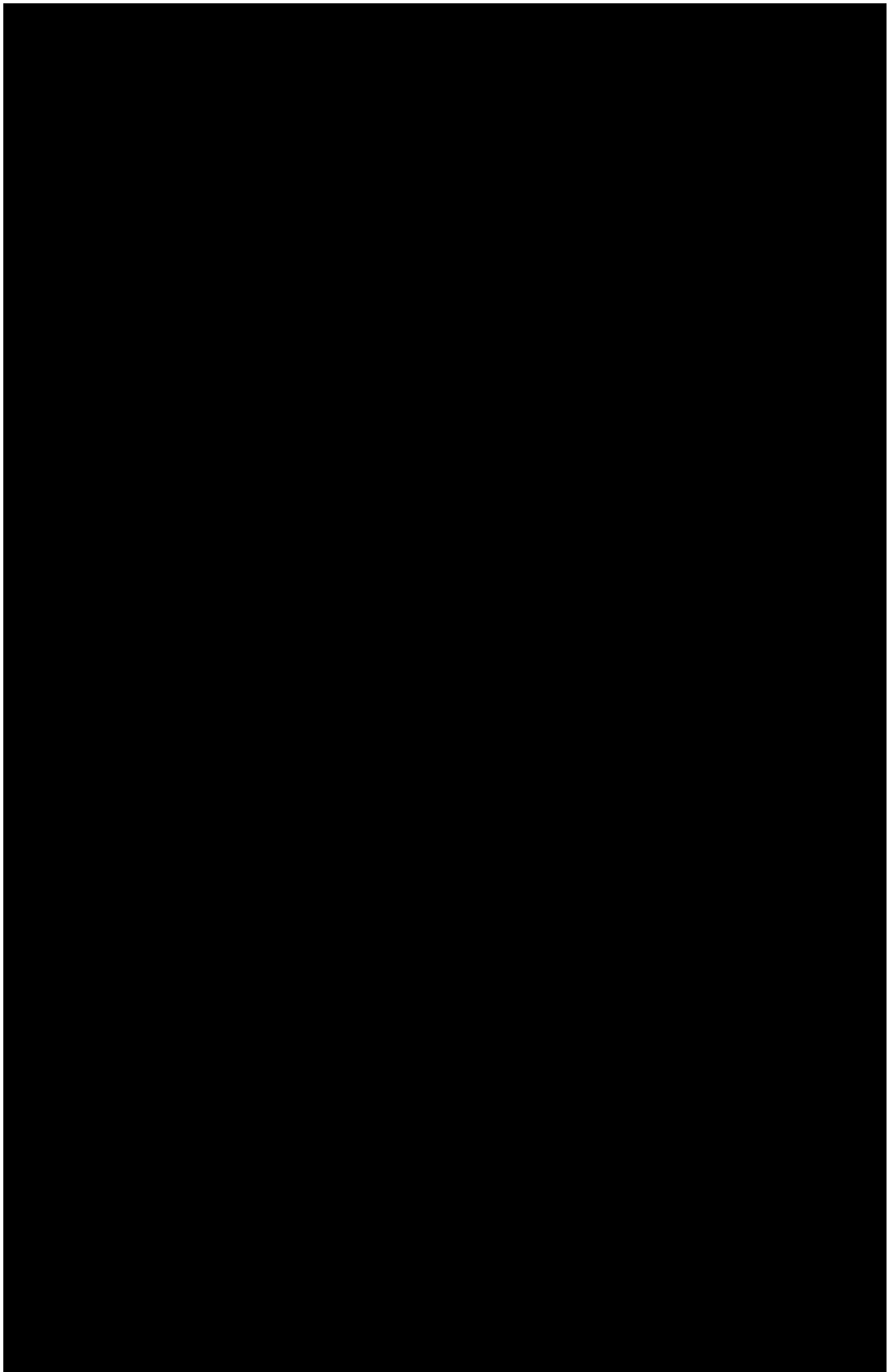


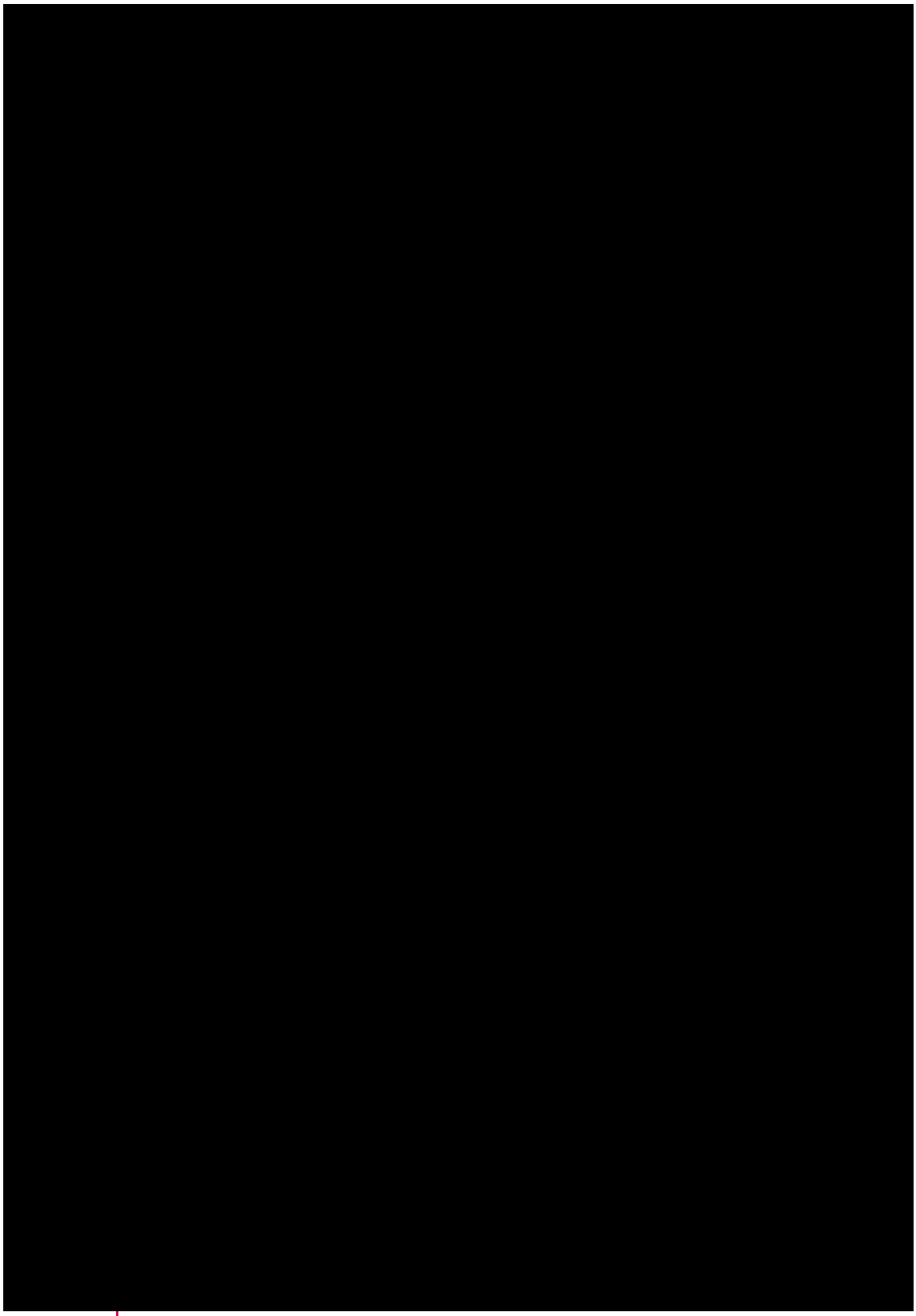


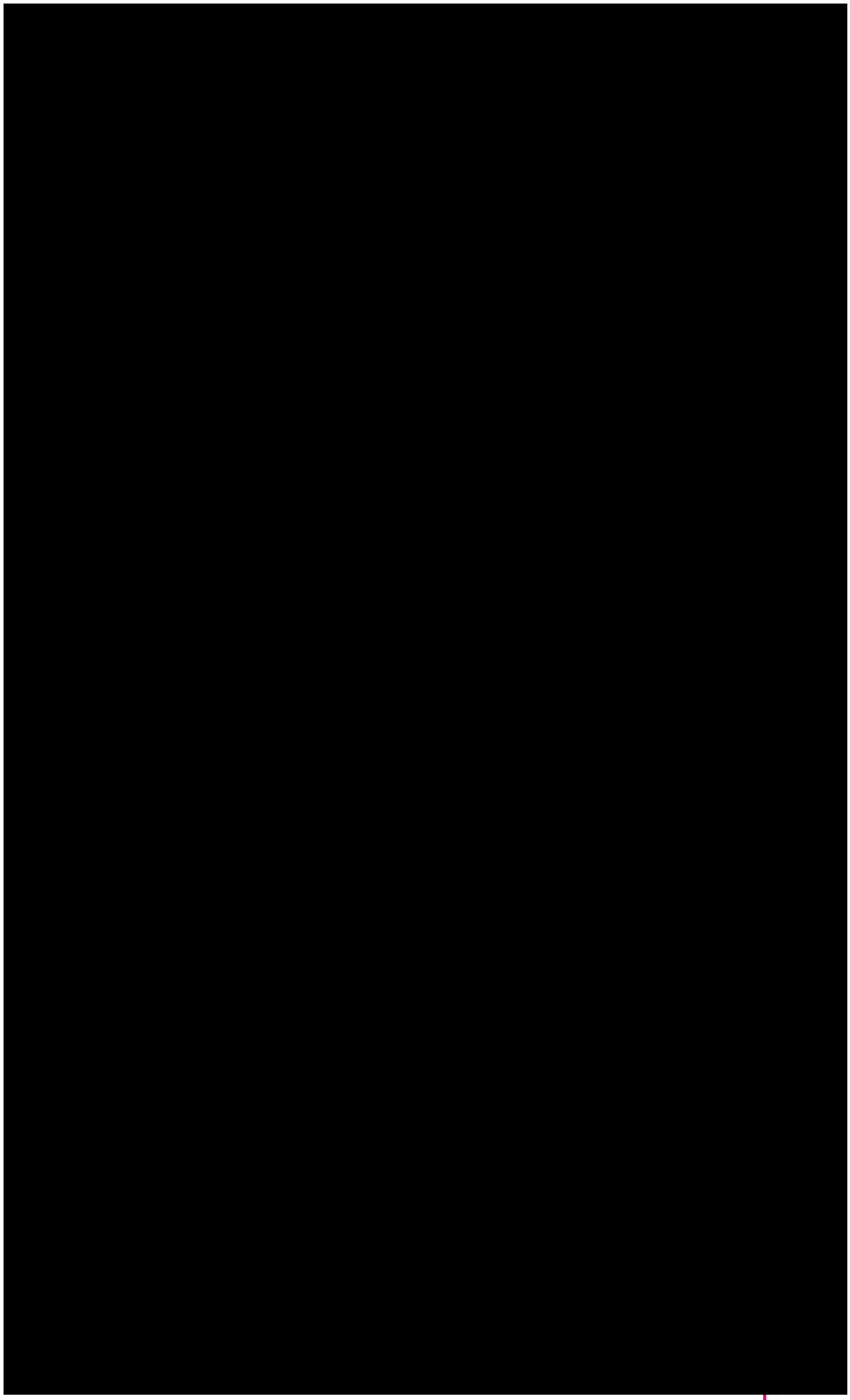












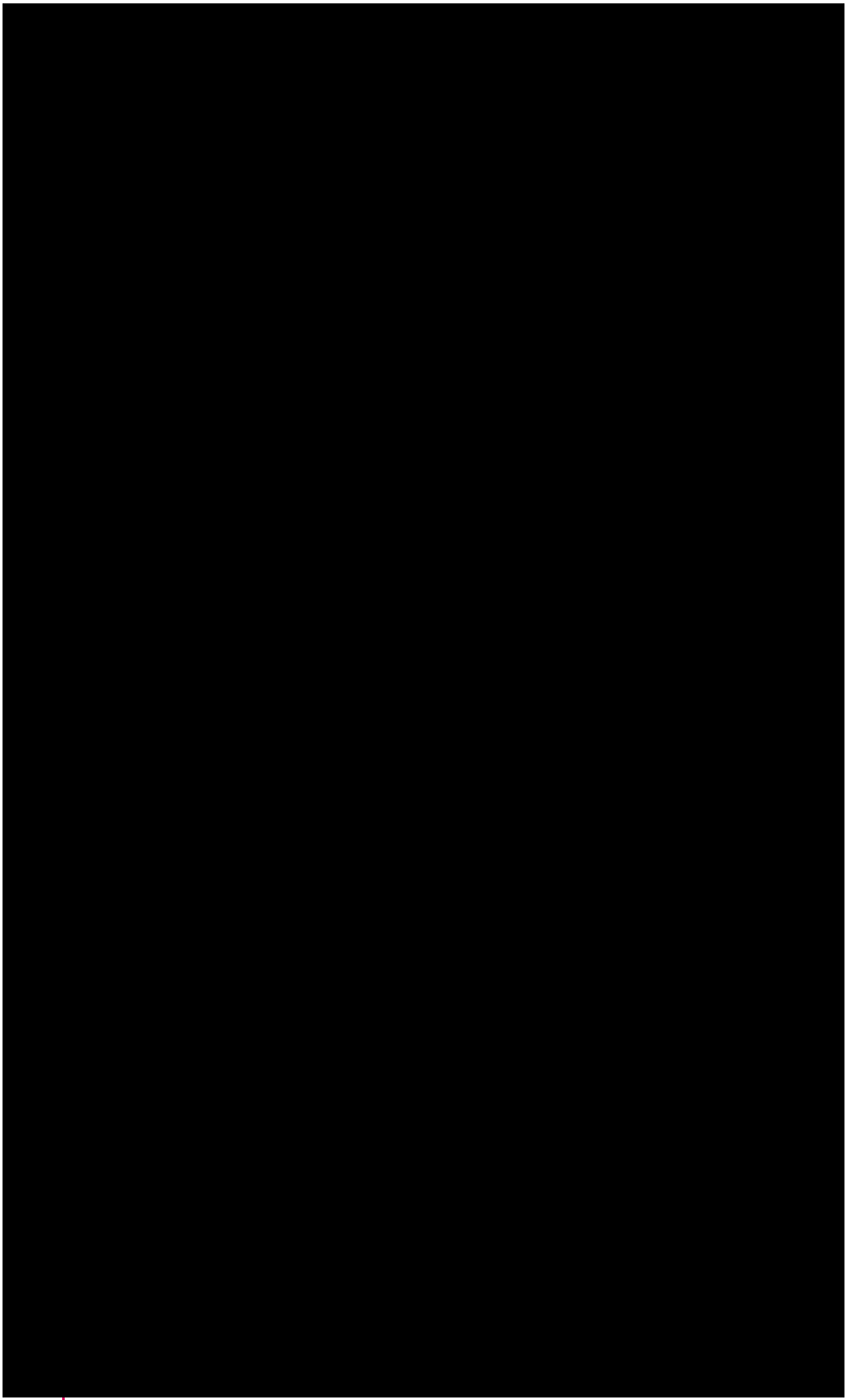
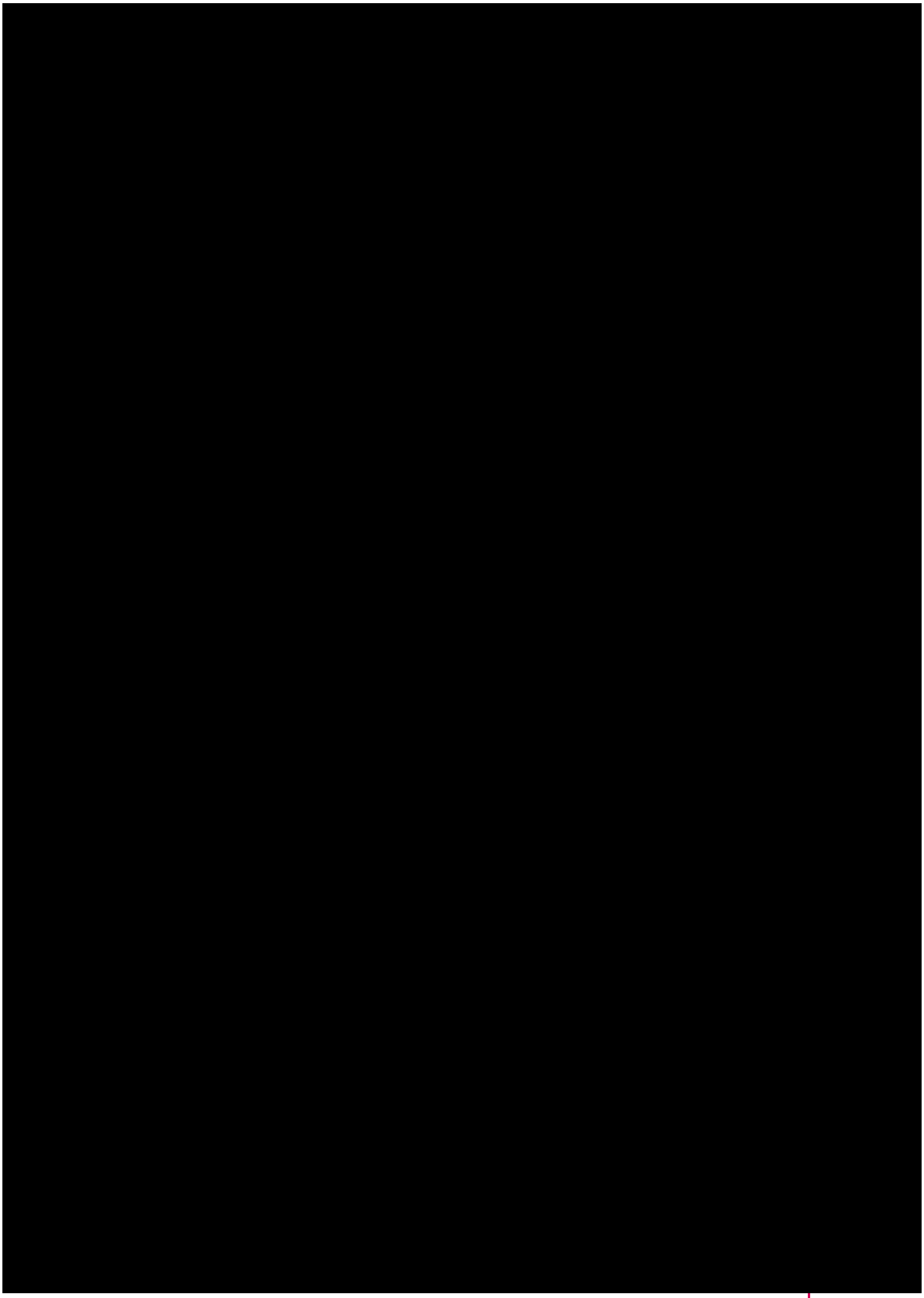
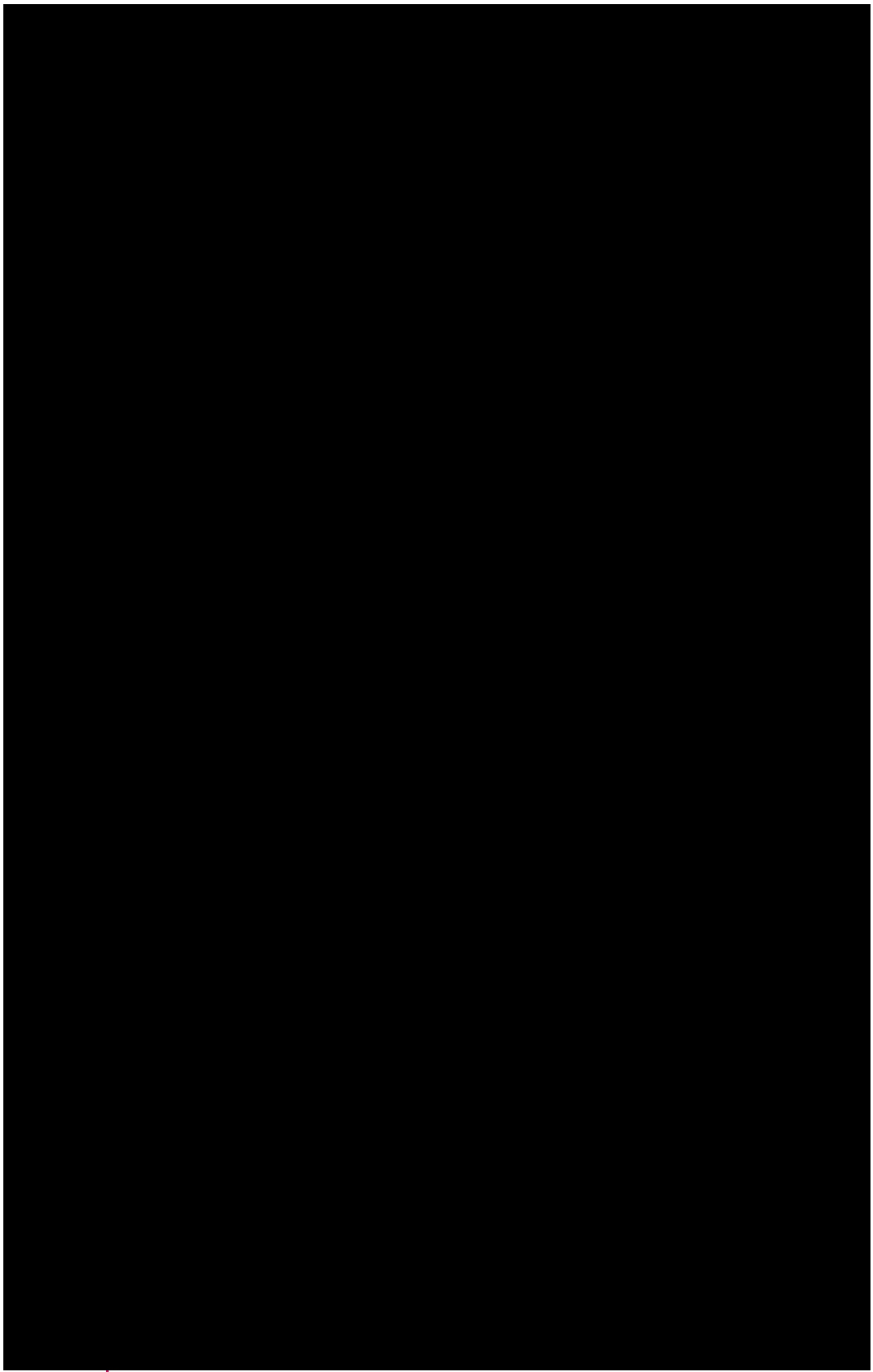
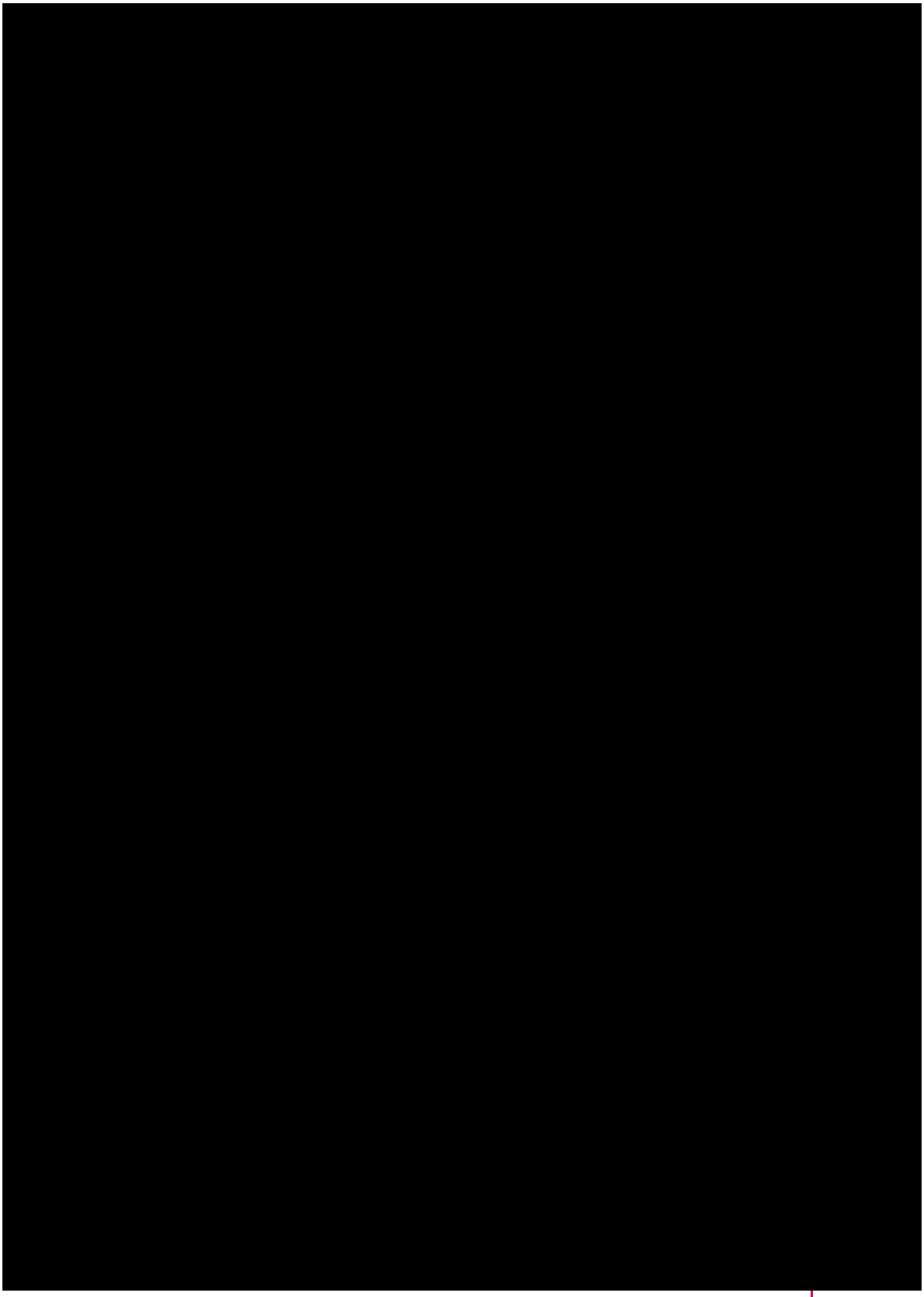
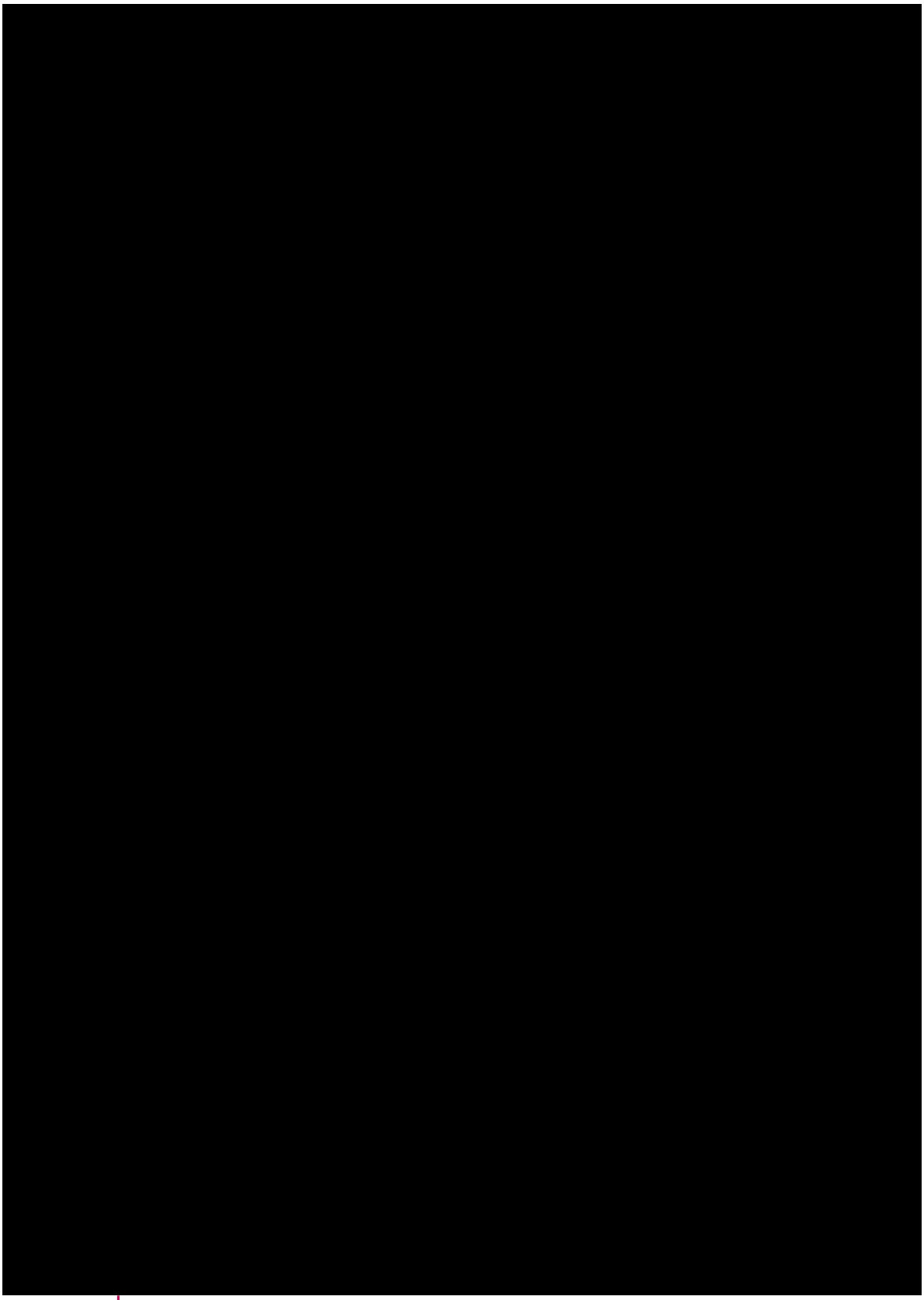


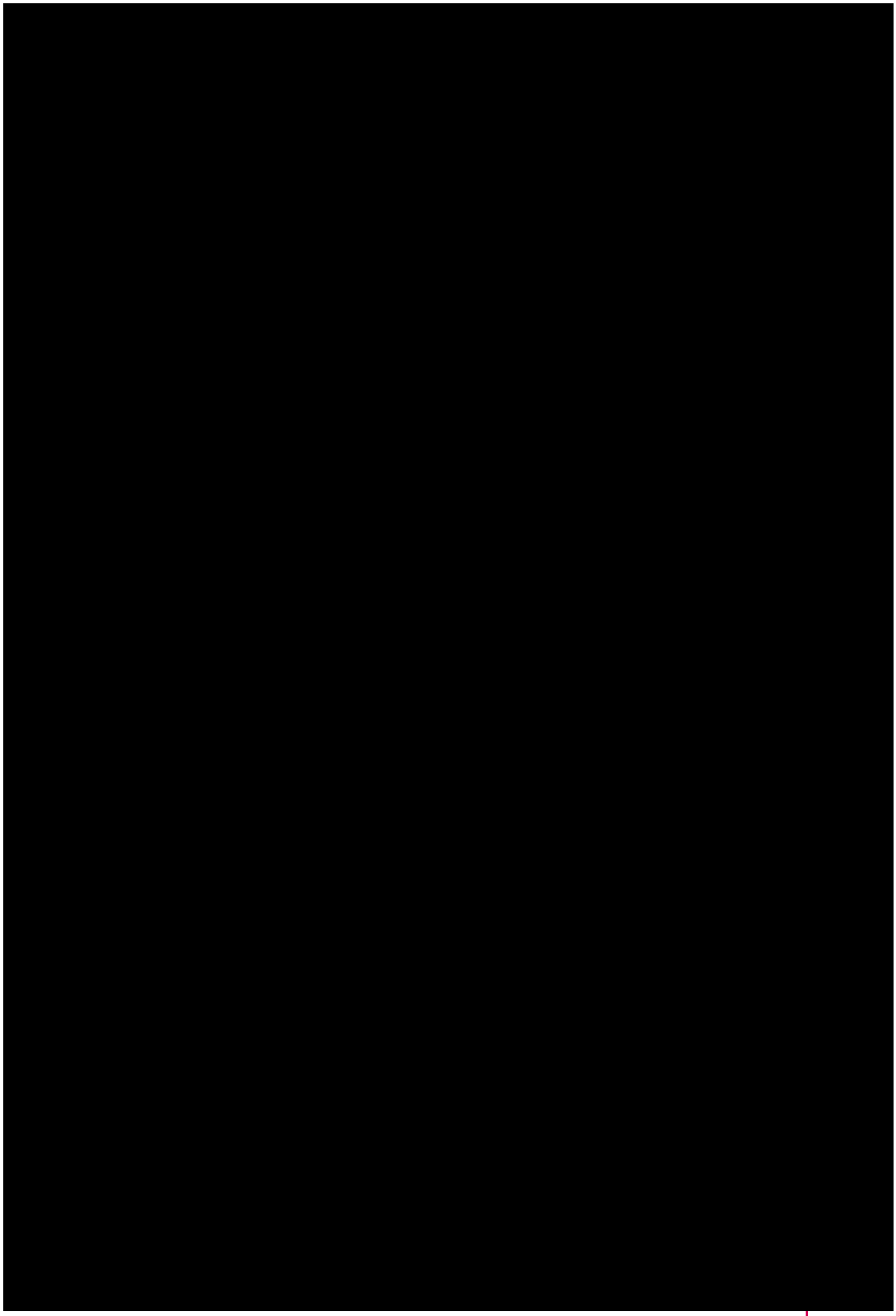
Fig.

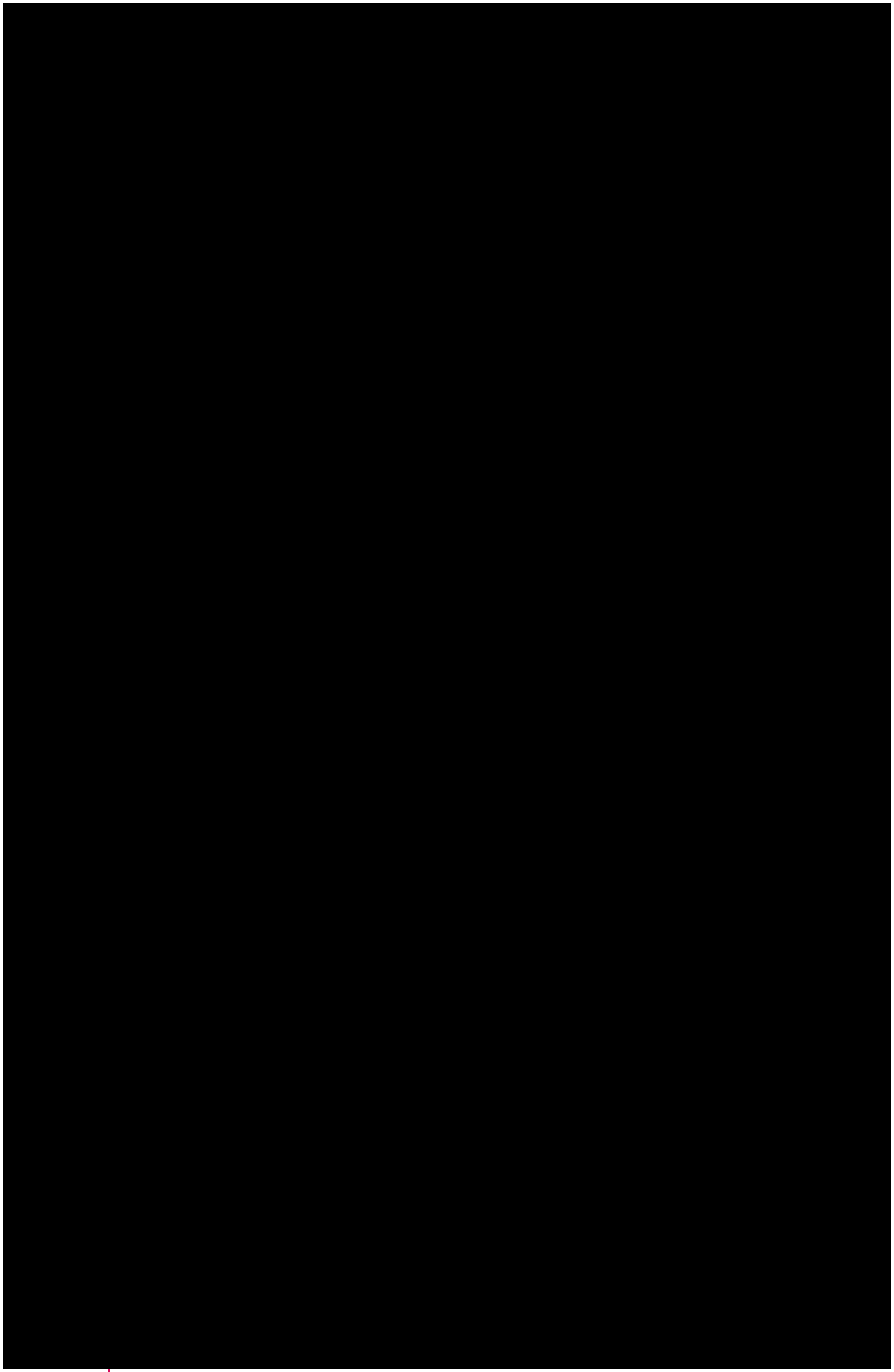


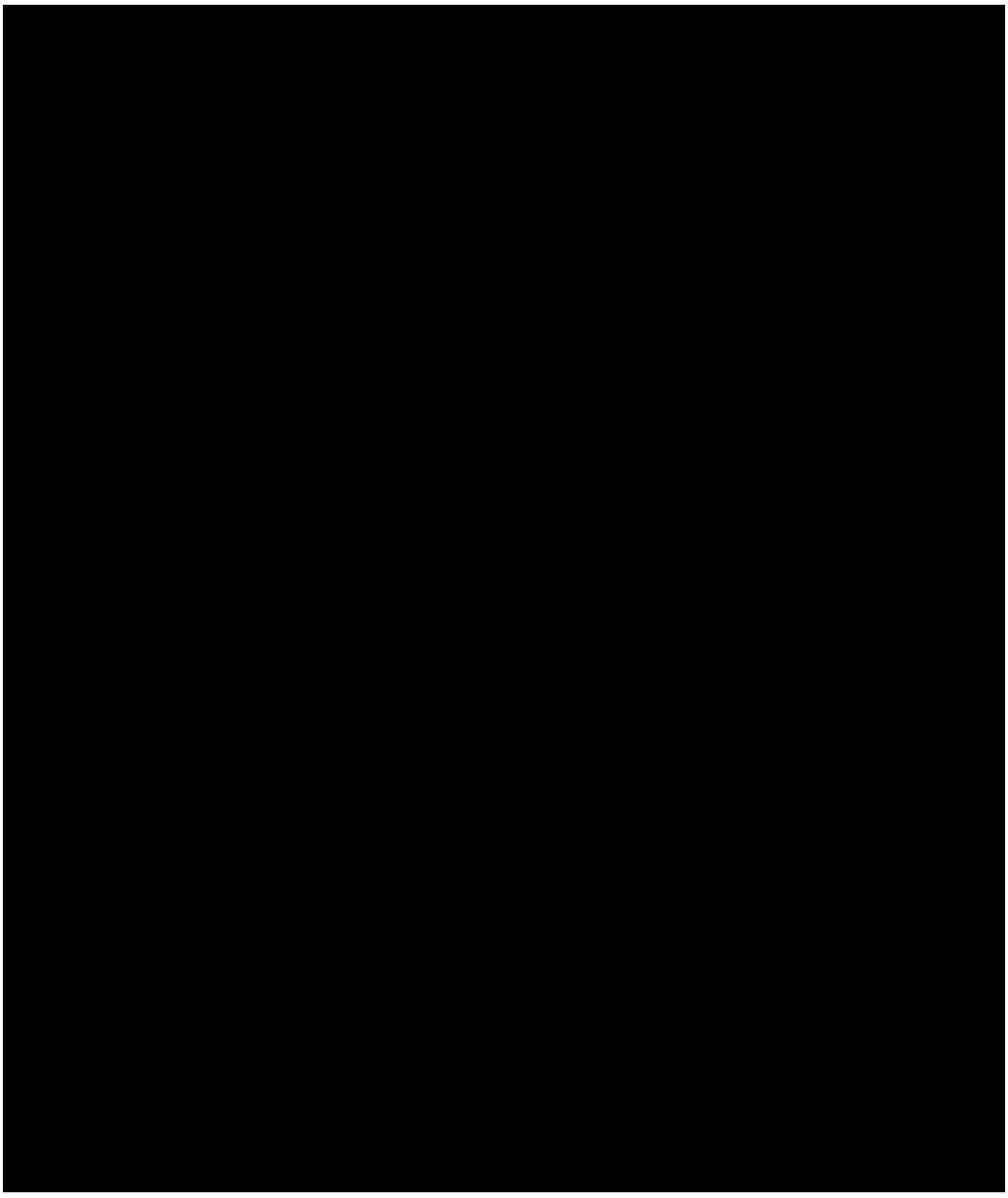


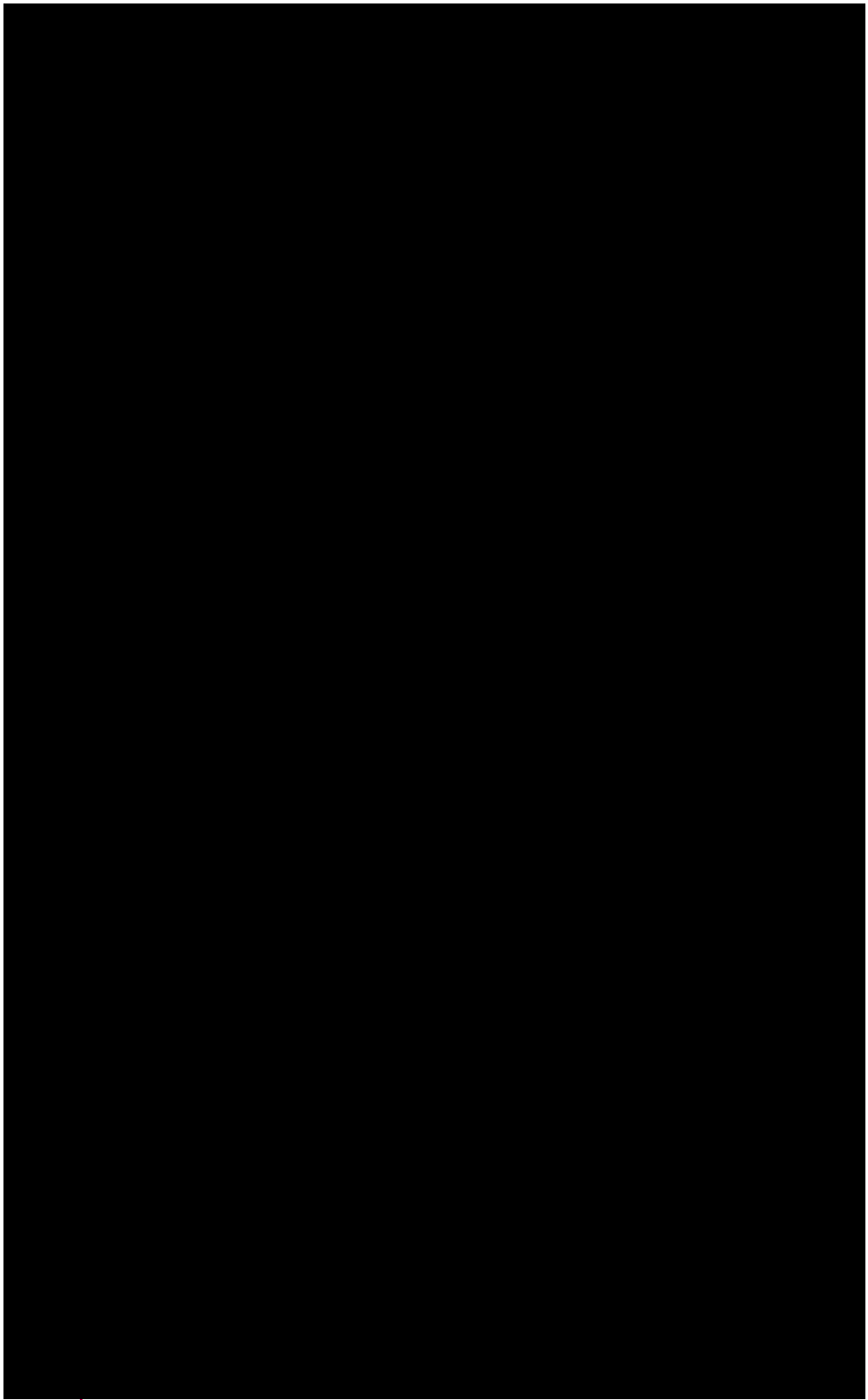


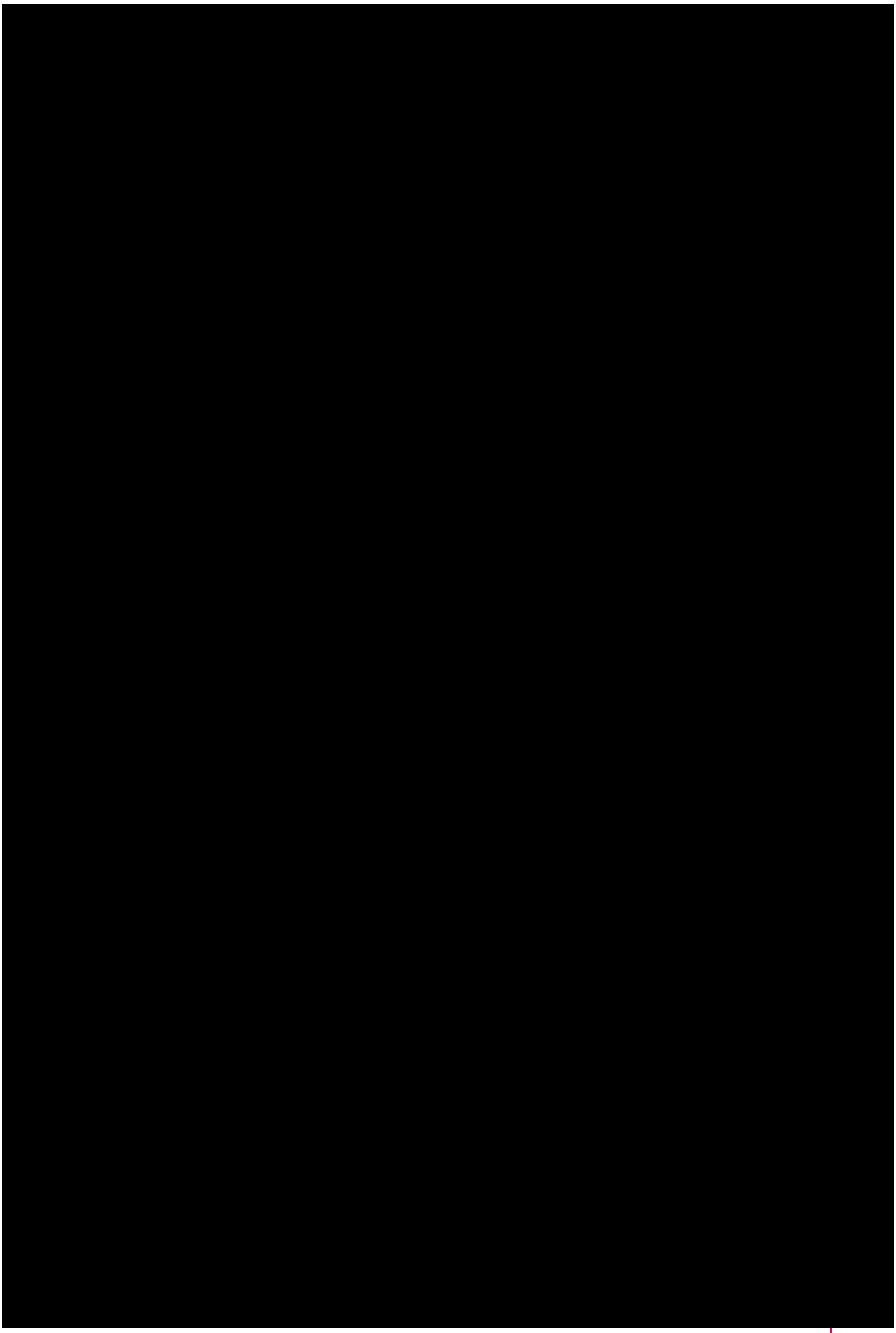


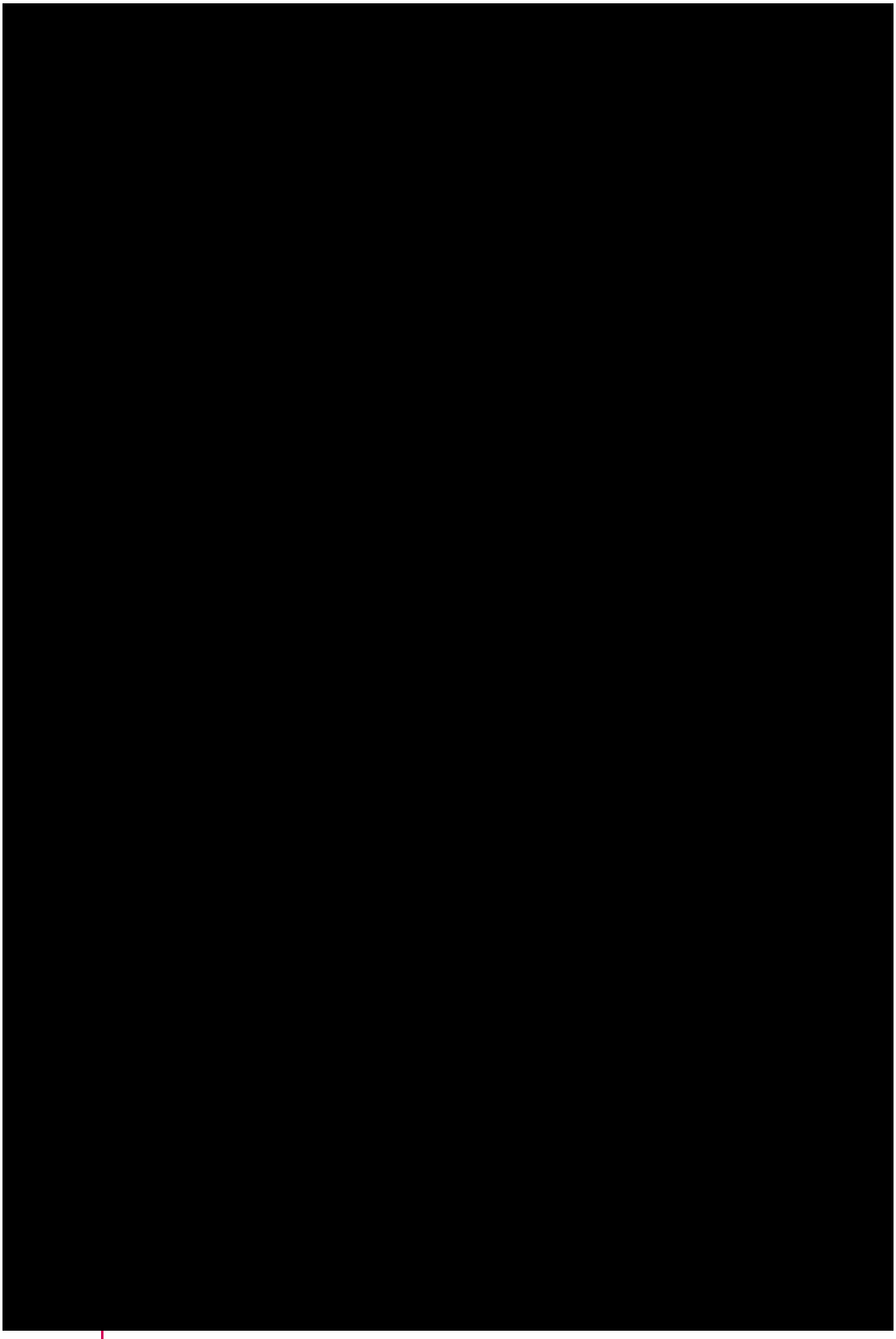


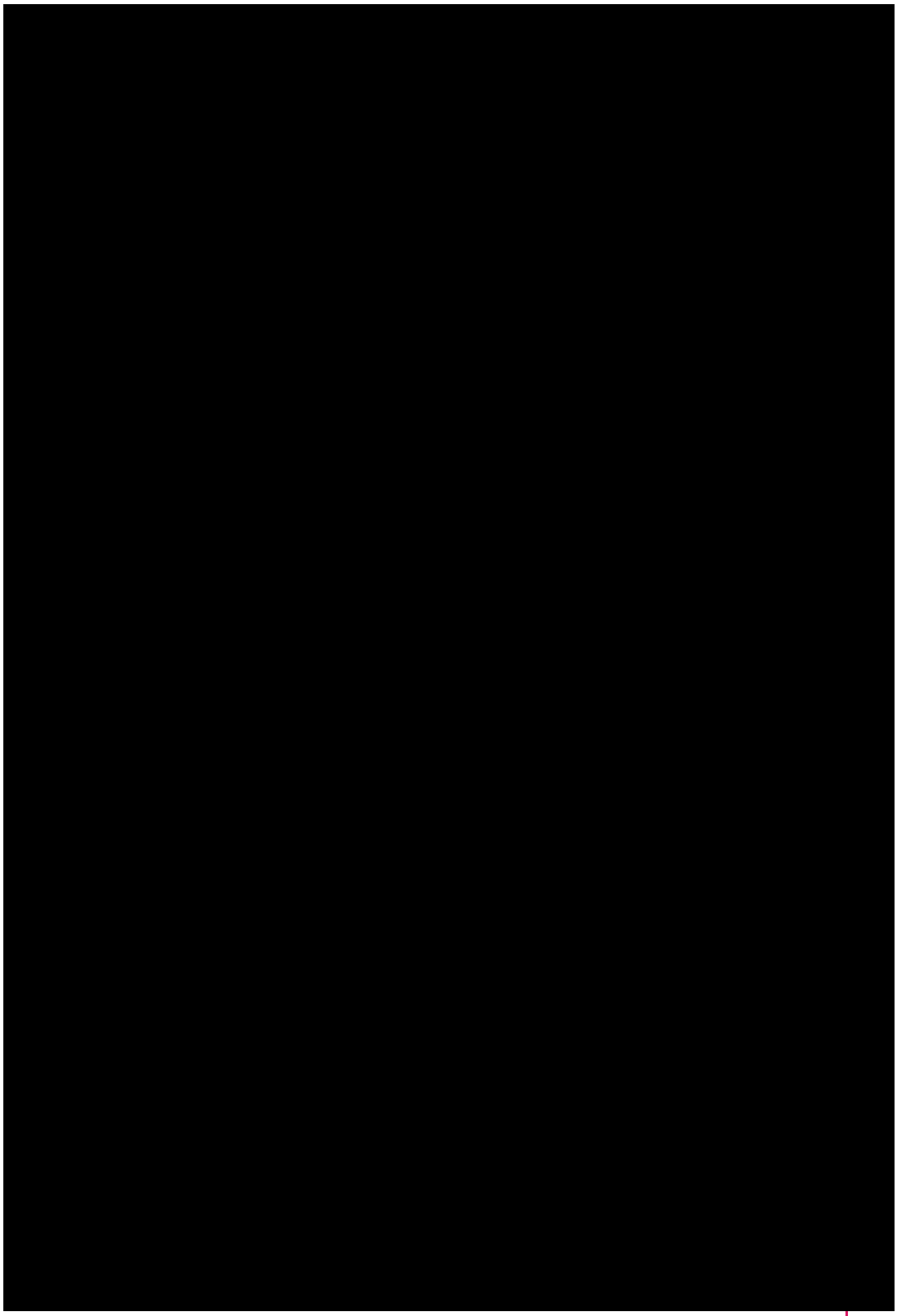


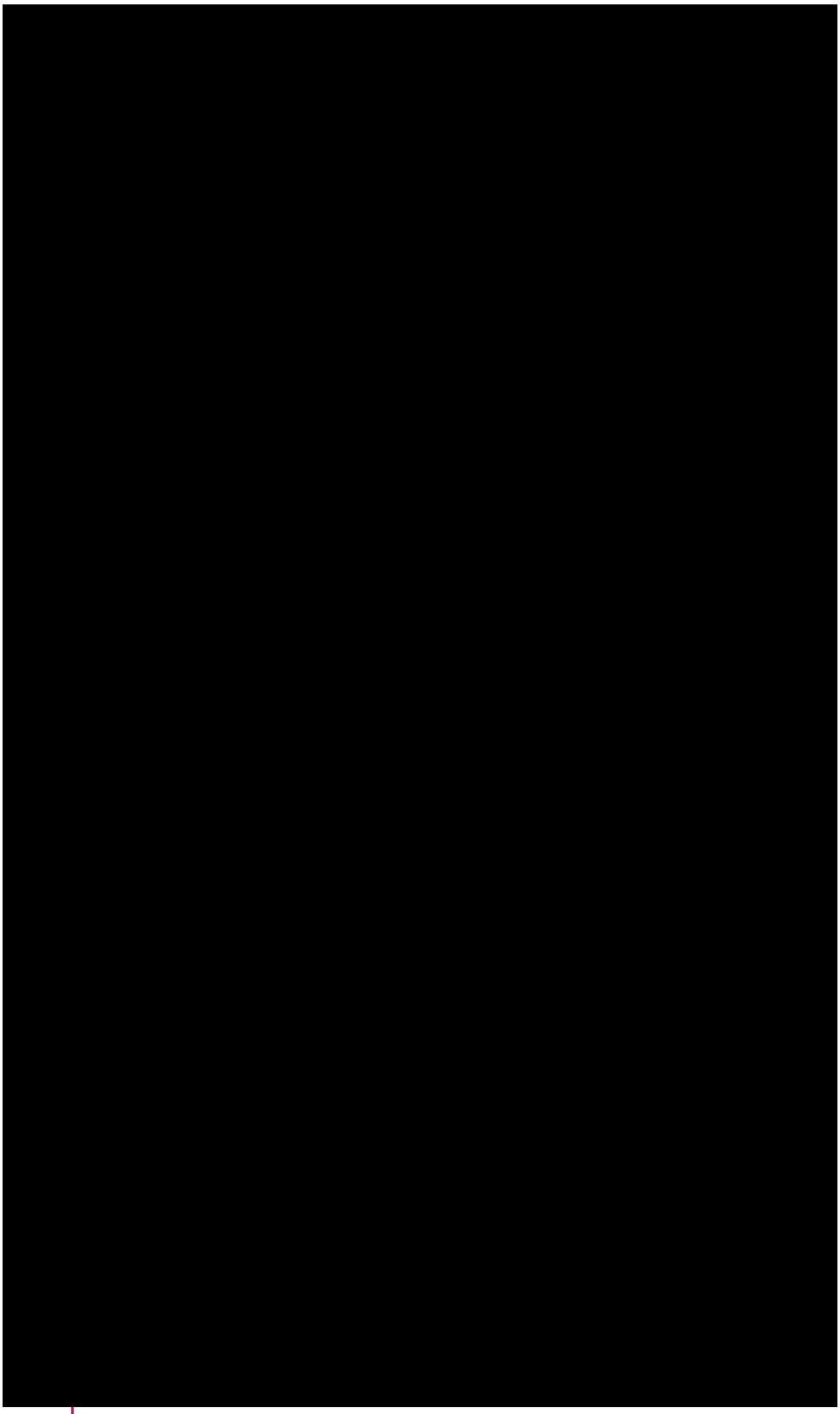


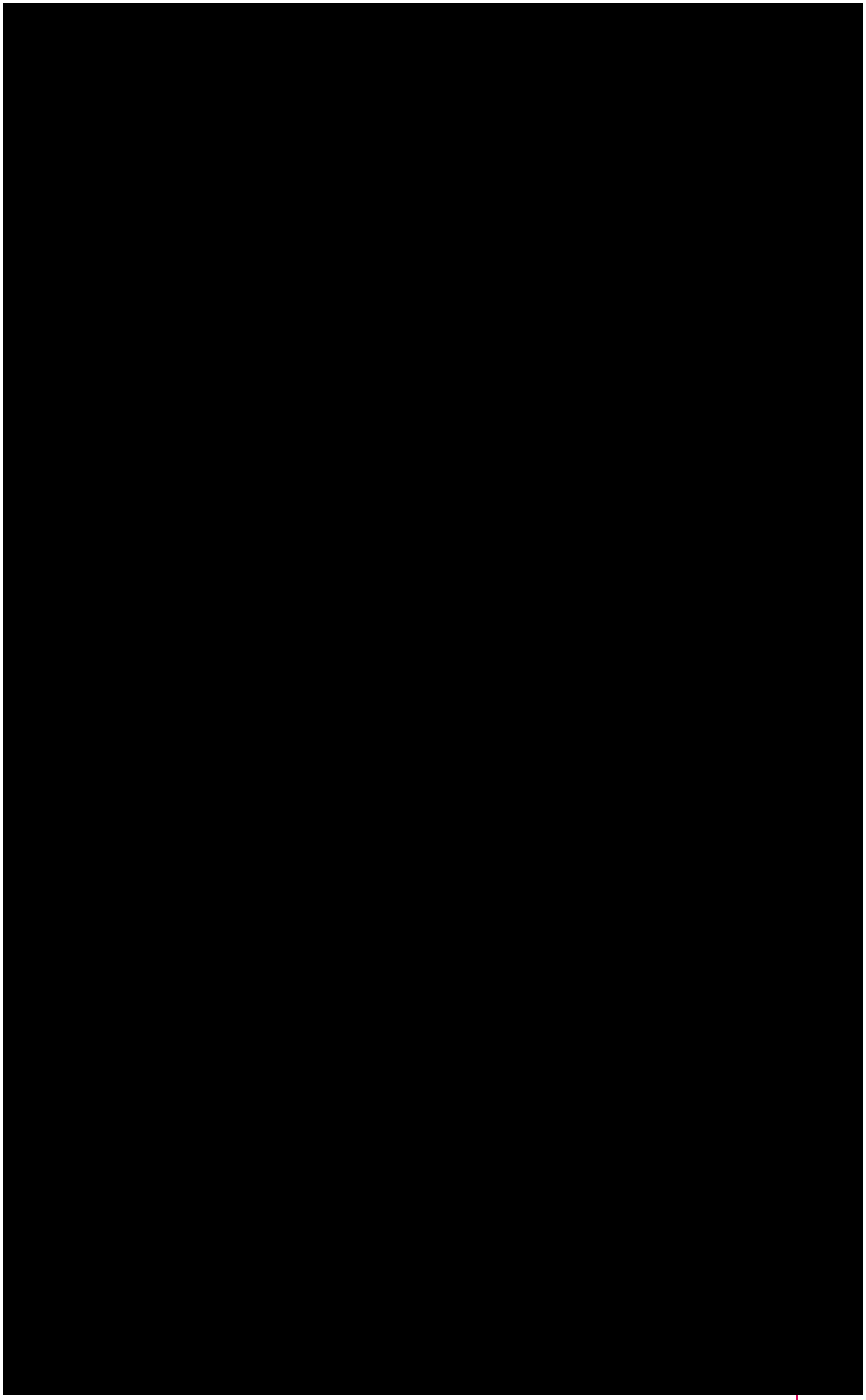


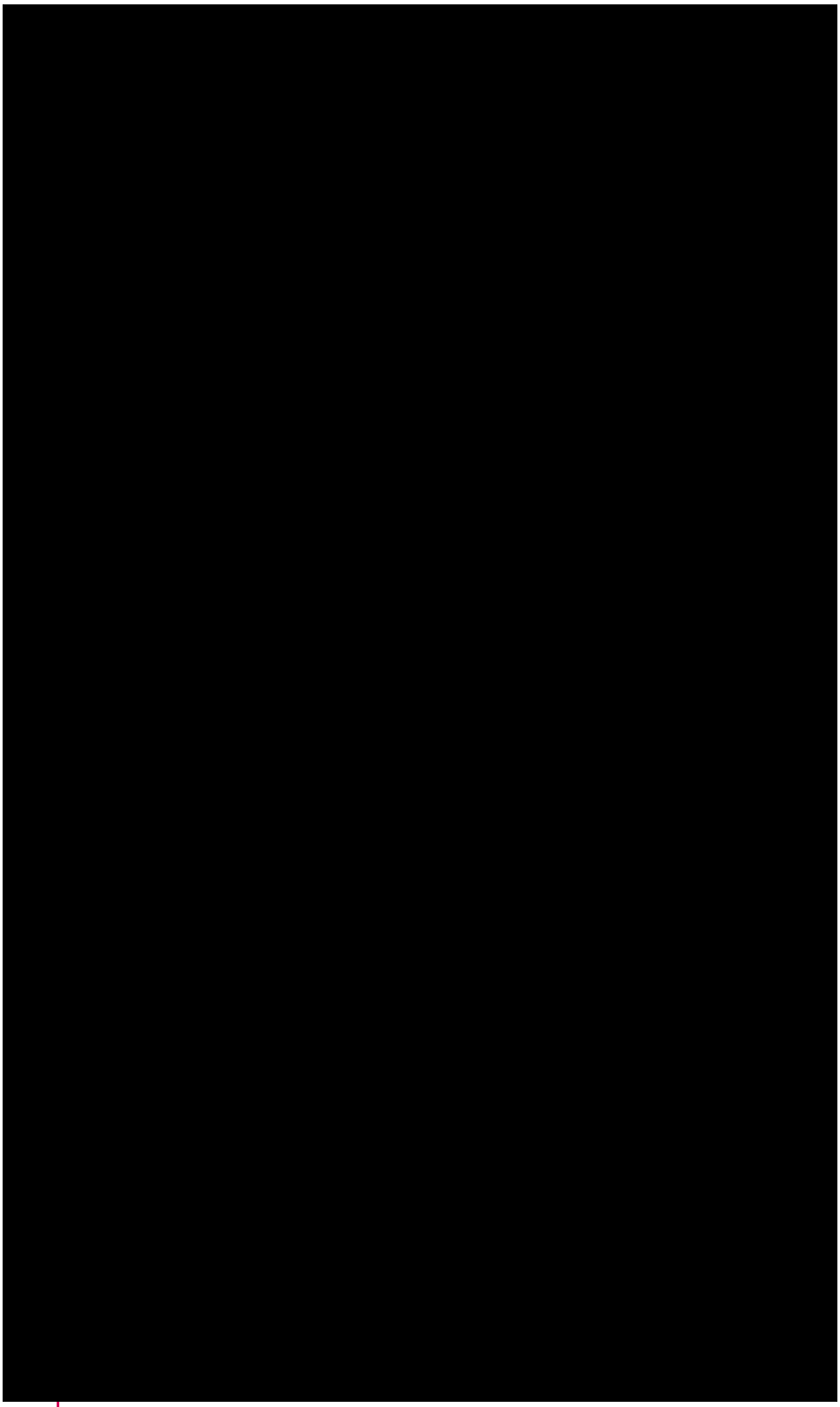


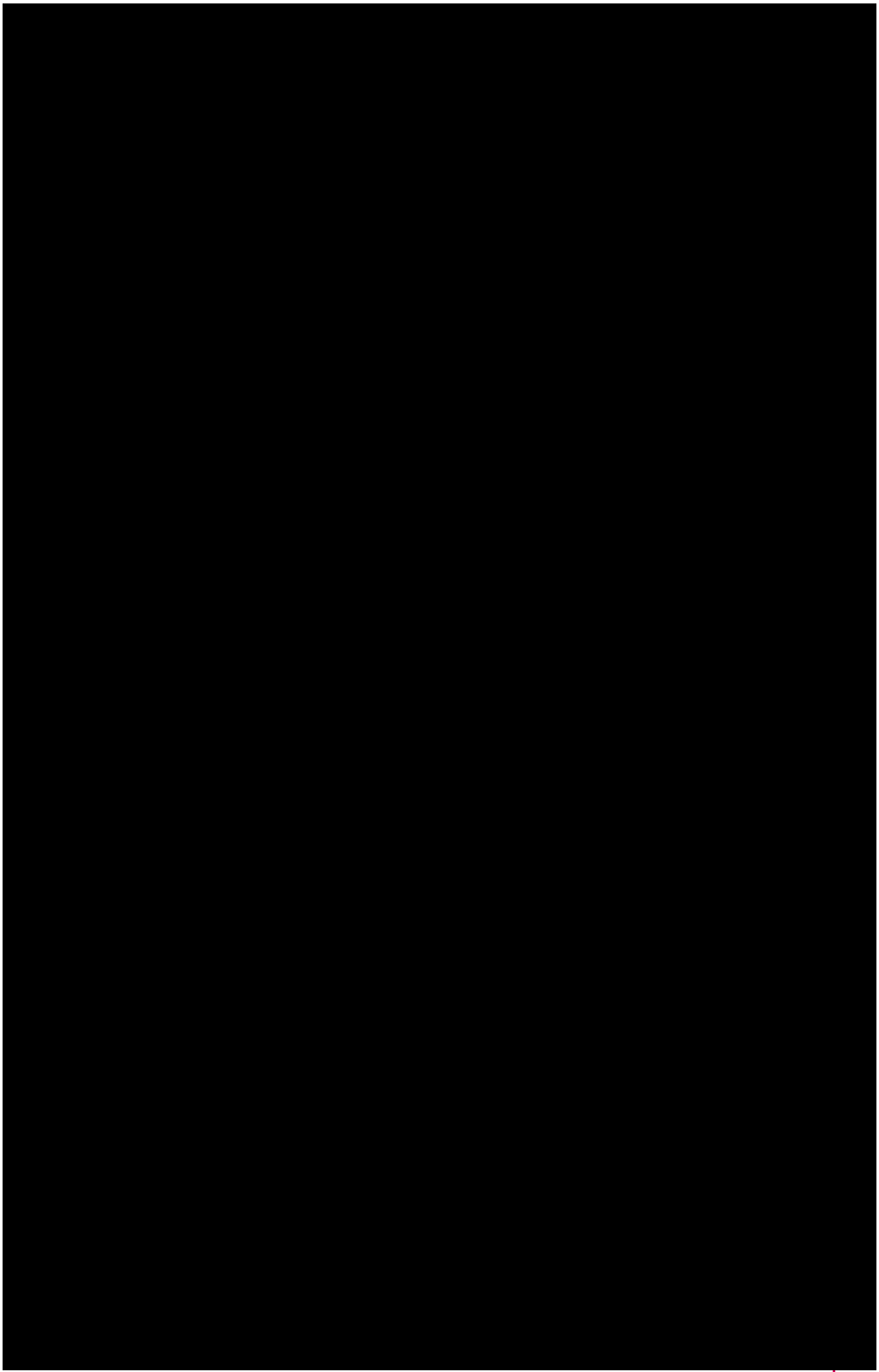


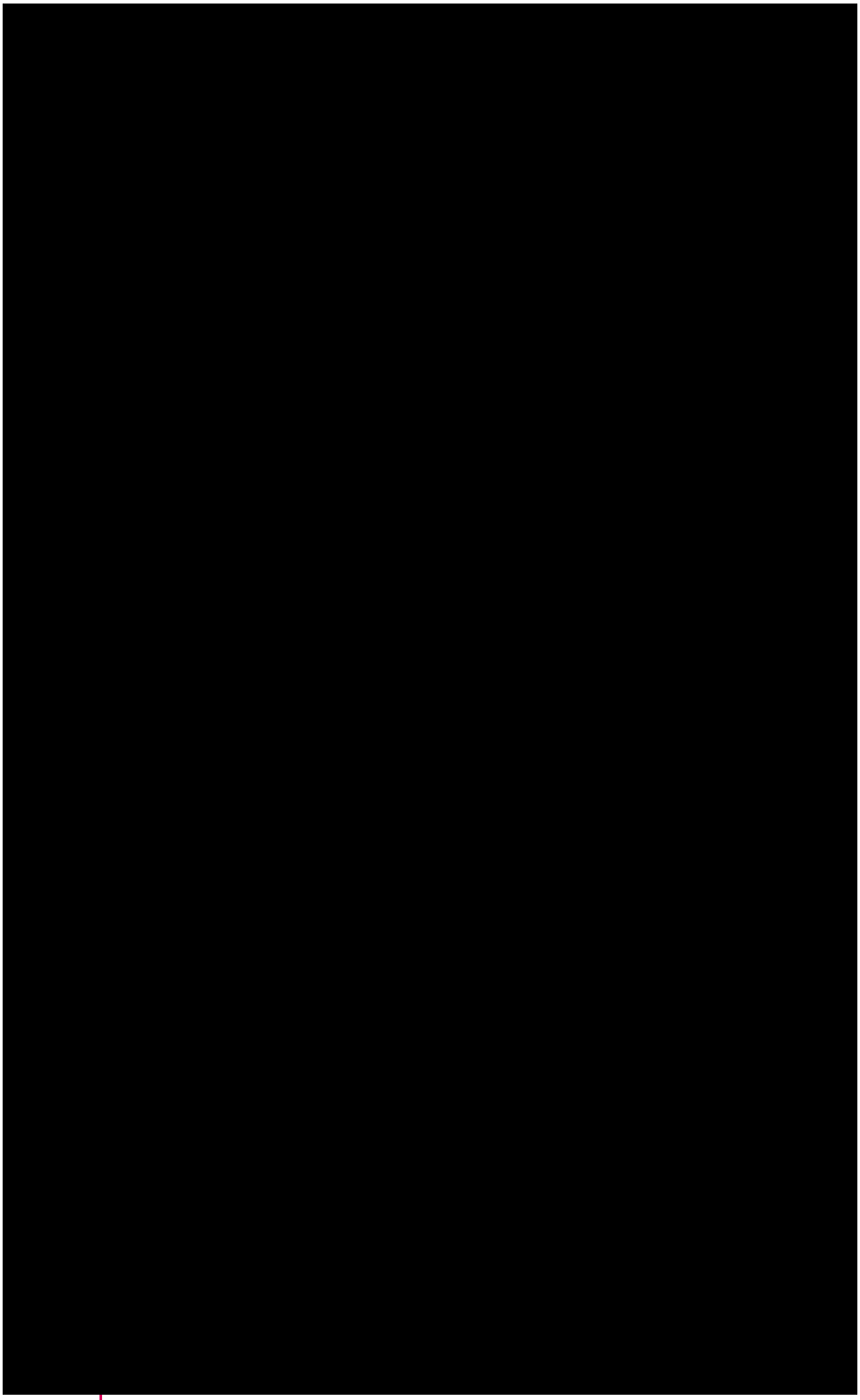


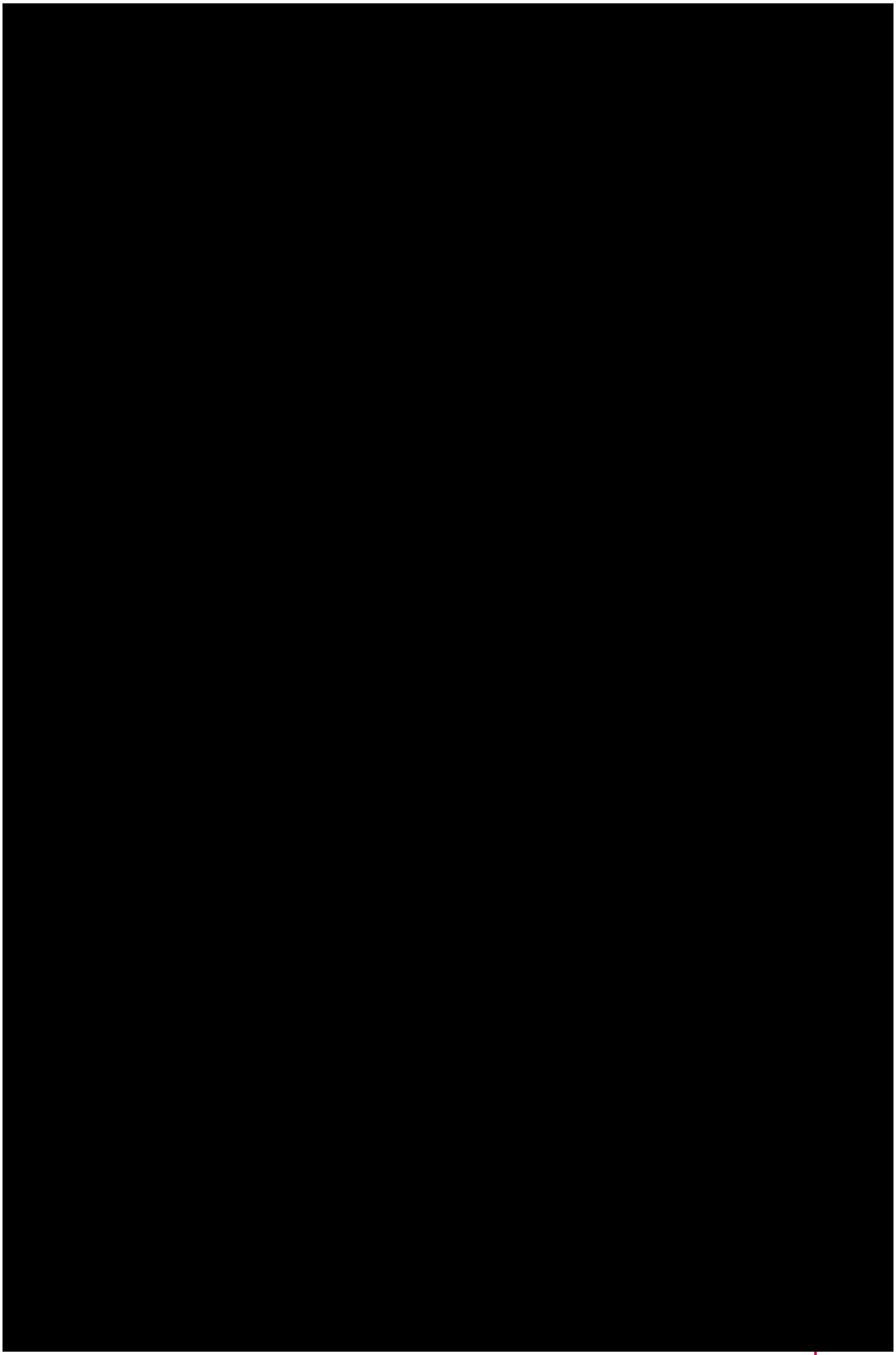


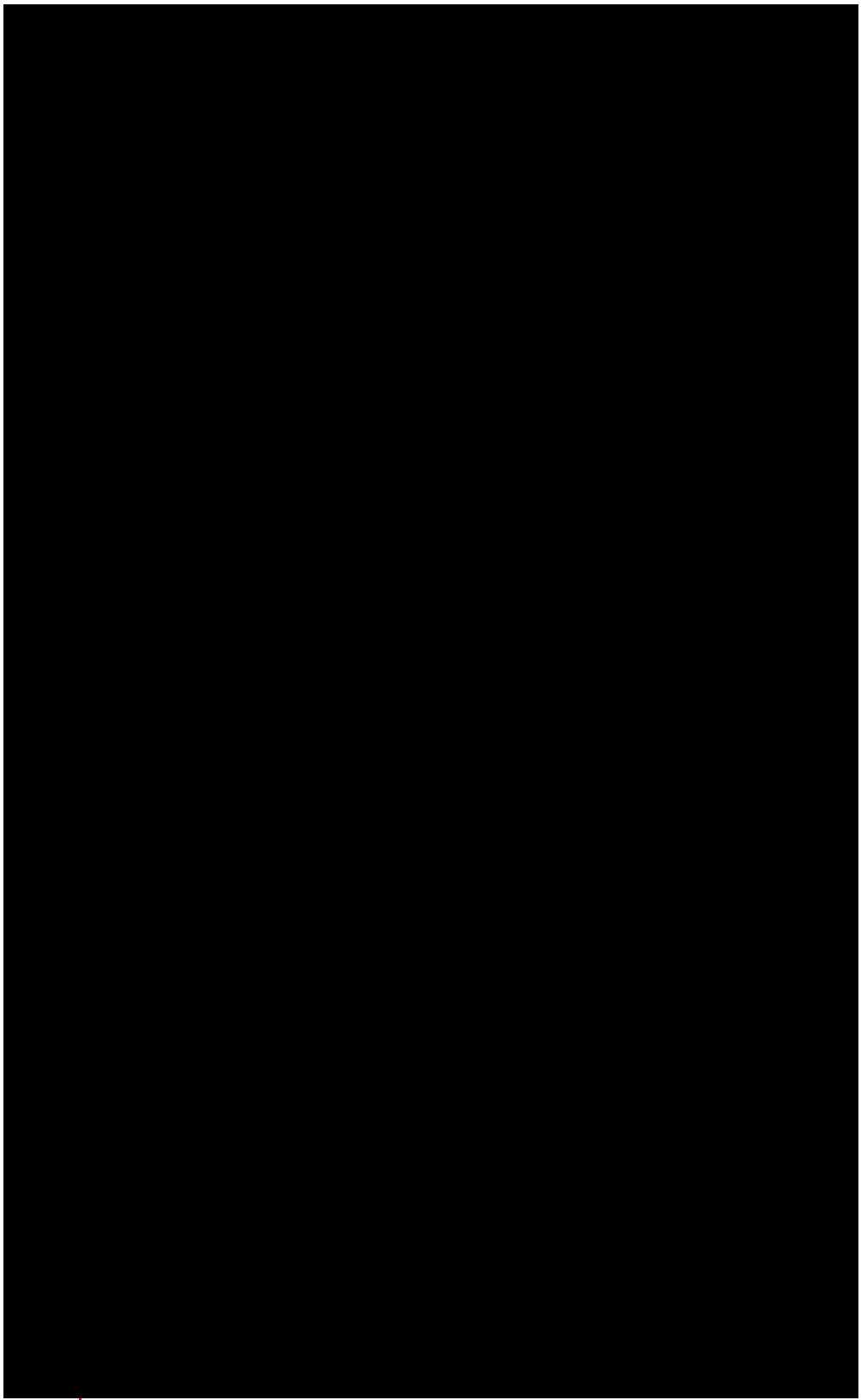




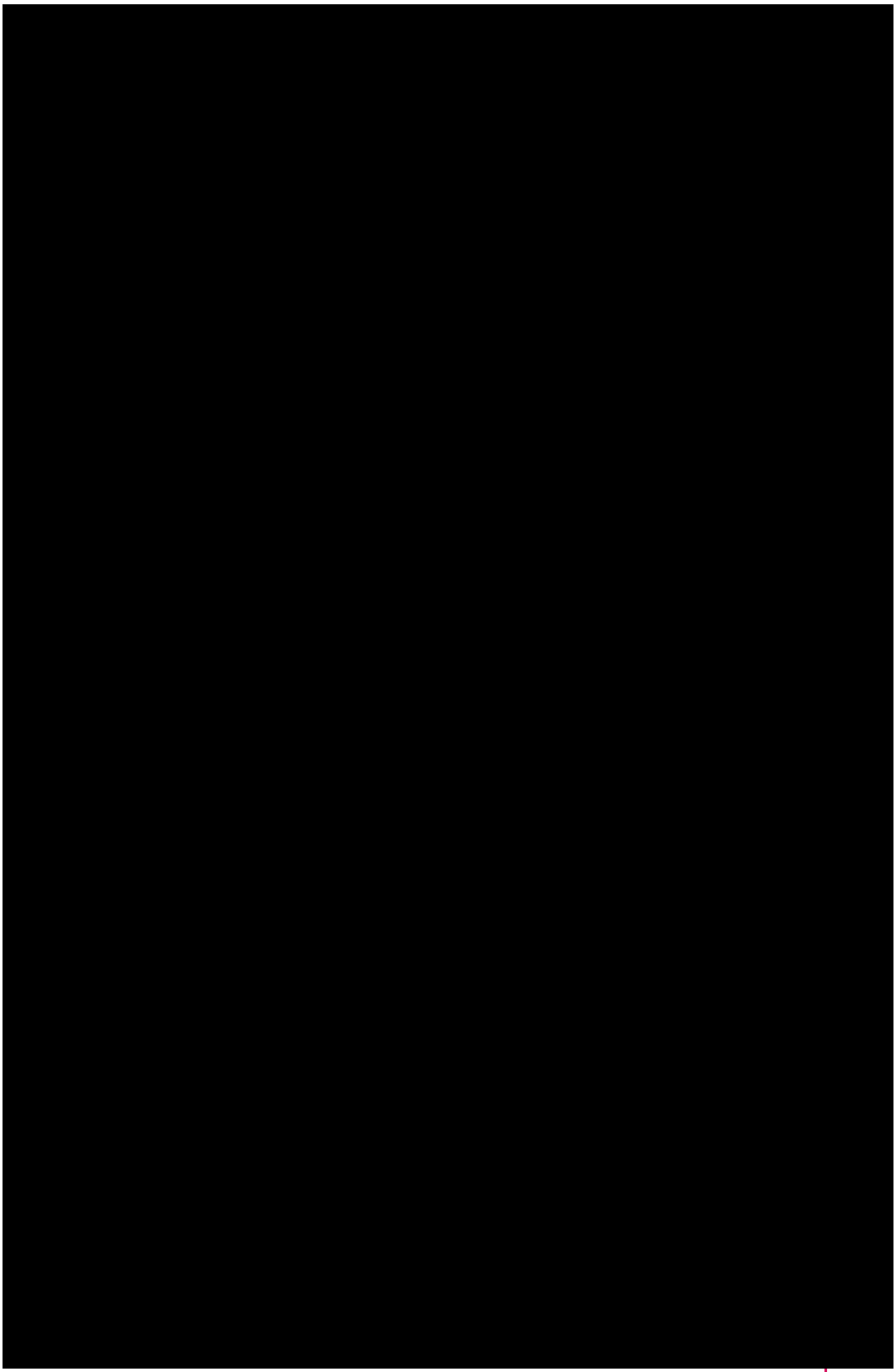


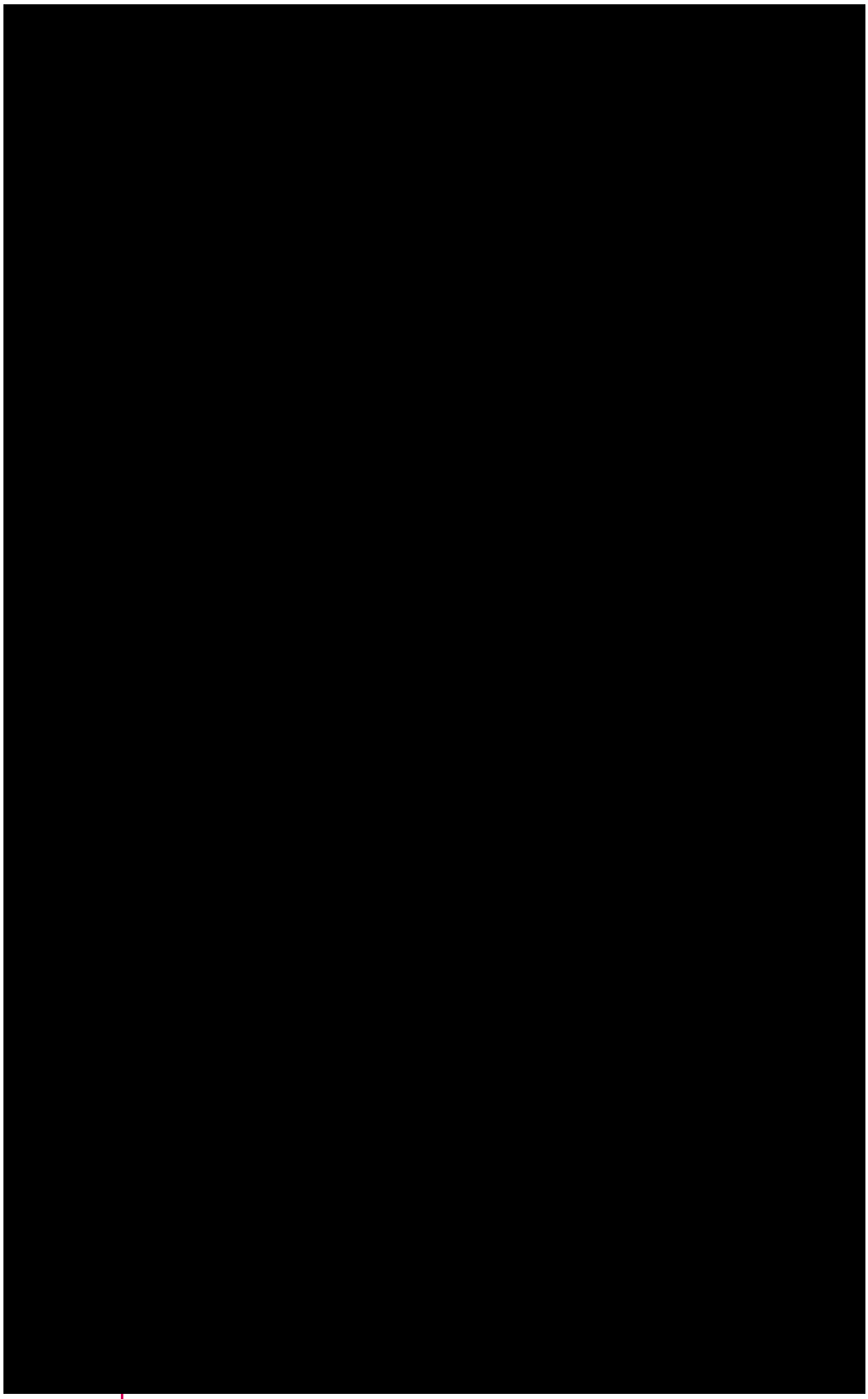


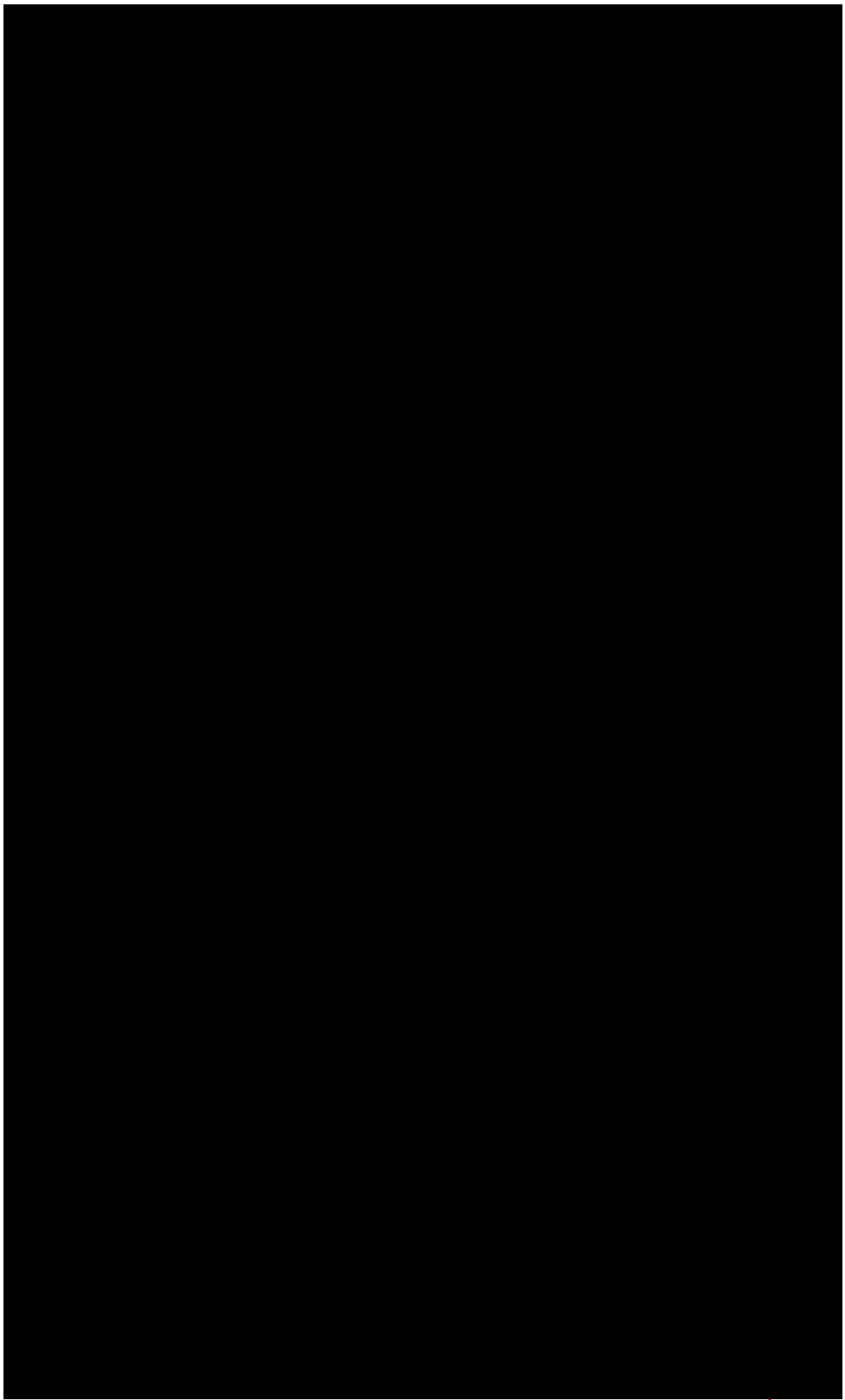


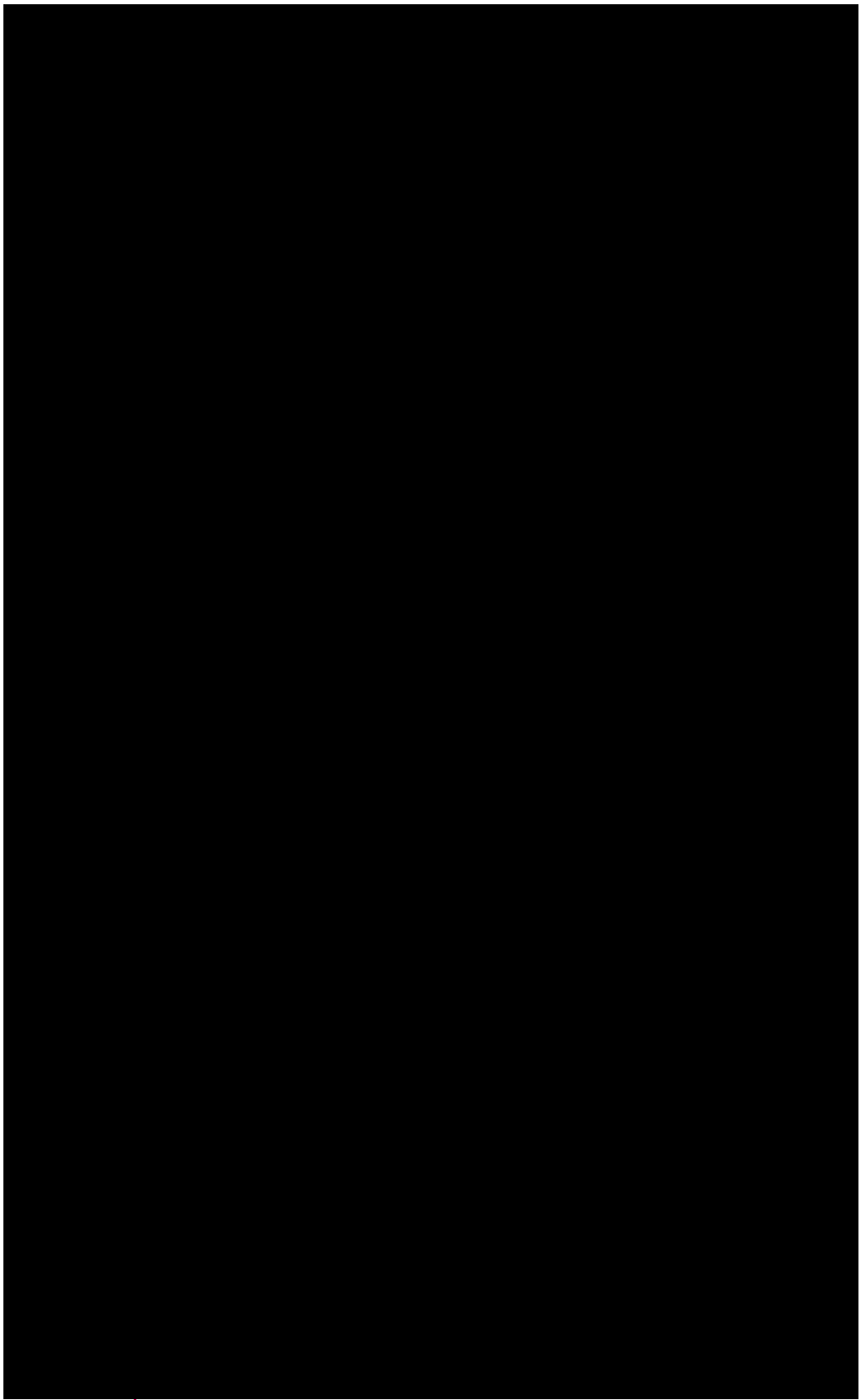


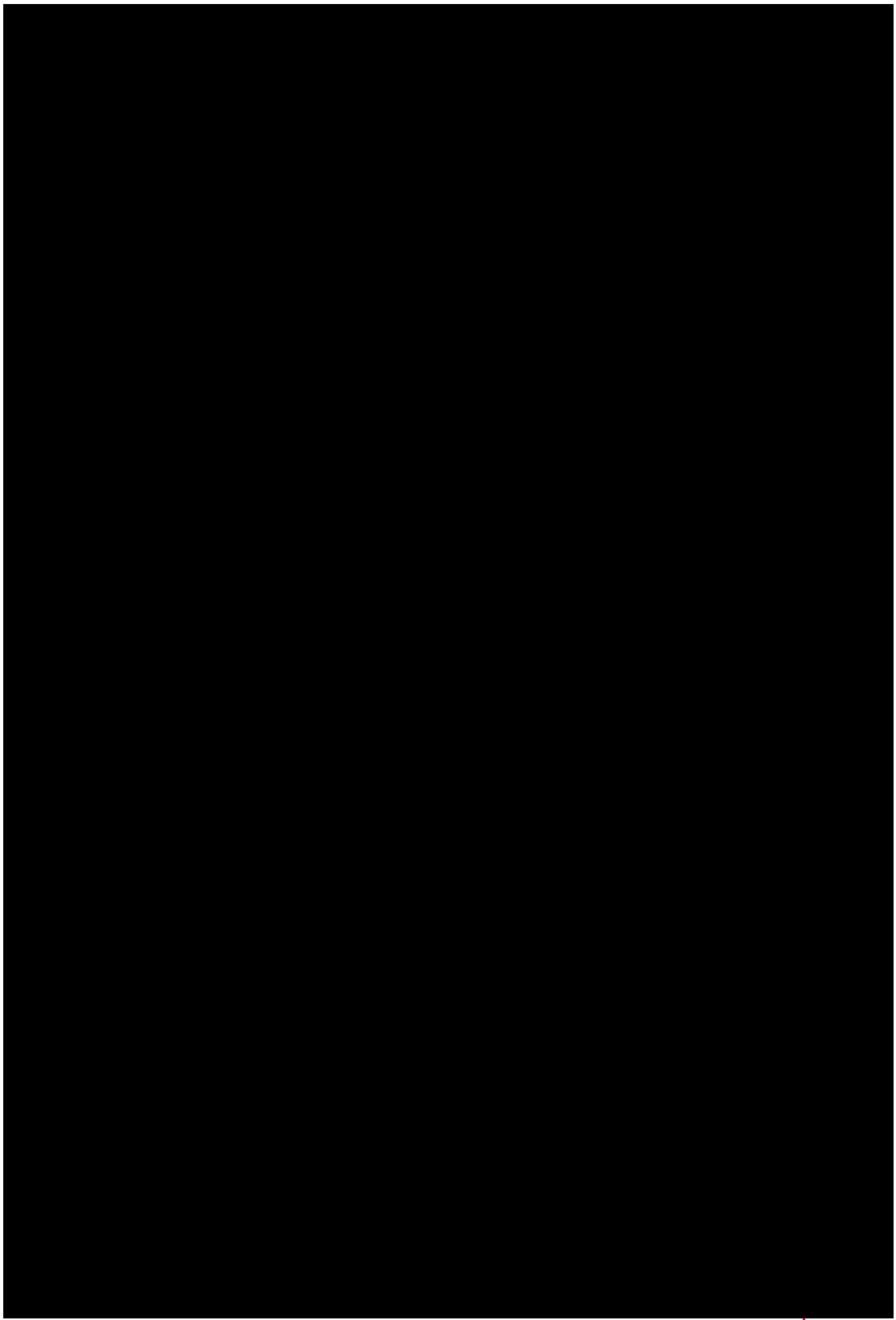
Fi

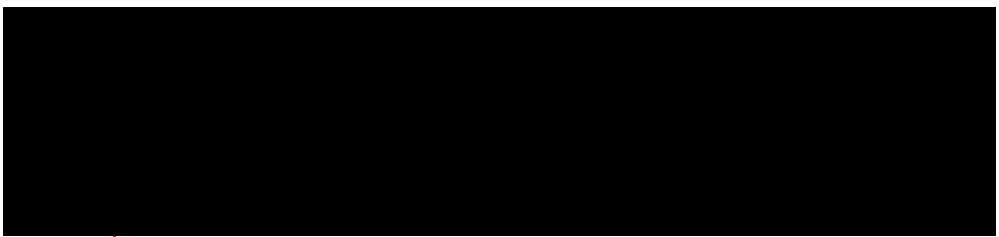












4

Single-molecule prediction of poly(A) tail length in Native RNA and cDNA with *tailfindr*

Summary: *A poly(A) tail is a stretch of adenosine that is added to the 3'-end of the mRNA during the final stages of RNA processing. Poly(A) tails have myriad roles — from facilitating nuclear export of RNA and influencing its stability to helping in translation initiation by forming a pseudo-circular loop with the 5'-cap. Existing methods for investigating poly(A) tail length rely on short-read sequencing. Assigning the estimated poly(A) tail length to a particular isoform can be difficult when the isoforms have the same alternative polyadenylation sites. Furthermore, these methods require converting the mRNA into cDNA which makes it impossible to study poly(A) tails in combination with other RNA modifications at the isoform level. Together with developing the Nanopore sequencing protocols, we also developed a computational method — tailfindr — that allows us to study poly(A) tail length at a single-molecule resolution in not only Native RNA, but also in direct and amplified cDNA as well. The poly(A) tail length prediction in Native RNA opens new avenues for studying them in tandem with cap methylations and other RNA modifications.*

The chapter begins with a background on poly(A) tails and their biological role followed by the state-of-the-art and their limitations. We then discuss how we address these limitations with our Nanopore sequencing-based approach using tailfindr. We then append our published work that goes into more details about our approach for poly(A)-tail profiling in RNA and unamplified DNA. Lastly, we explain some of the follow-up work that we did for poly(A)-tail profiling in amplified cDNA as well.

4.1 Poly(A) tails and their biological role

Poly(A)-tailing is the non-templated addition of adenosines to the 3'-end of the RNA during the last stages of RNA processing. Before polyadenylation, the transcripts undergo an endonucleolytic cleavage step that removes some of the bases from their 3'-end [87]. Transcripts from the same gene may be cleaved at different locations,

which gives rise to alternative polyadenylation sites that can be in the 3'-UTR, exon, and even in an intron [88]. Next, a poly(A) polymerase adds a stretch of the newly-formed RNA ends. The nuclear poly(A) tail length falls in a tightly-controlled species-specific range, e.g., 70–90 nt for yeast [89] and 200–250 nt for mammals [90, 91].

Once polyadenylation and splicing are complete, poly(A)-binding proteins (PABs) bind to the poly(A) tails and export the transcripts out of the nucleus. Once in the cytoplasm, the poly(A) tails may undergo shortening which gives rise to heterogeneous steady-state poly(A) tail length distribution [90].

The poly(A) tail plays an important role in two of the many mRNA decay pathways: 1) in deadenylation-dependent cap-hydrolysis, and 2) in 3'–5' degradation by exonucleases. Both these decay pathways require the poly(A) tail to be degraded first. The rate of deadenylation is a major determinant of the mRNA half-life and depends on the sequence elements in each individual mRNA transcript [92].

In the cytoplasm, poly(A) tails help initiate translation by forming a closed loop which the the physical bridging between the 5'–cap and 3'–poly(A) tails is mediated by translation initiation factors eIF4E/4G and poly(A)-binding proteins [93]. In this the secondary structure of the mRNA is resolved which facilitates ribosome recruitment onto the mRNA [94]. Furthermore, once a ribosome has traversed from 5' to 3', due the closed-loop structure, the same ribosome can be recycled to do another round of translation [95]. Short poly(A) tails have recently been found to be a conserved feature of highly expressed genes [96] indicating that poly(A) tail length influences how efficiently an mRNA is translated.

4.2 Poly(A)-tail profiling

As we have seen, poly(A) tails play a crucial role during the life-cycle of an mRNA. Their length encodes information about the age of mRNA and how efficiently it is translated to protein. The process of finding the poly(A) tail length of these mRNA transcripts is referred to as poly(A)-tail profiling and is a crucial tool in understanding the role of poly(A) tails in a biological system.

4.3 State-of-the-art for poly(A)-tail profiling

In this section, we describe some of the existing methods for investigating poly(A) tail length.

4.3.1 extension PolyA-tail test (ePAT)

This is a gel-based method for determining the size of poly(A) tails in a chosen isoform or gene [97]. First, a DNA oligonucleotide with a 5'-poly(T) stretch is hybridized to the 3'-end of the RNA at 25 °C (Fig. 4.1). A Klenow polymerase extends the 3'-recessed end of mRNA using dNTPs. Next, the temperature is increased to 55 °C which releases internally primed DNA oligos, followed by reverse transcription which extends correctly primed DNA oligos. A gene-specific primer and a universal primer are then used to amplify the cDNA. The cDNA is used in a gel to find the length of the cDNA. Because the cDNA contains both the gene-specific primer and universal primer sequence, to find the true length of the poly(A) tails, a poly(A) standard with 12 nt poly(A) tail and the same gene-specific primer and universal primer is also used on the gel. When compared against a gel ladder, the difference between the length of the standard and the gene cDNA bands yields the length of the poly(A) tail in the gene.

4.3.2 Poly(A) profiling by sequencing (PAL-seq)

Poly(A) tail profiling by sequencing is a short-read method for profiling poly(A) tails [98]. A DNA oligo with 3'-biotin and a splint oligo with 5'-poly(T) stretch are incubated with total RNA in the presence of T4 DNA ligase (Fig. 4.2). The products are then digested with RNase T1 and size selected (104-750 nt) on a gel. Splint-ligation products are bound to streptavidin beads, and while in this bead-bound state, the 5'-end of then RNAs are phosphorylated and a 5' sequencing primer is ligated. The RNA is then reverse transcribed into cDNA, biotin cleaved off, and the RNA digested away. The cDNA was then clonally amplified on the flow cell which leaves clusters of forward strands on the flow cell. Each cDNA cluster on the flow cell is from one original RNA strand. Next, a primer is hybridized immediately 3'-of the poly(A) and extended in the presence of dTTPs and biotin-conjugated dUTPs — the amount of dUTPs incorporated in the poly(A) tail are proportional to the length of the poly(A) tail. Next 36 nucleotides of the transcript proximal to the poly(A) tail are sequenced by synthesis. This is followed by flooding the

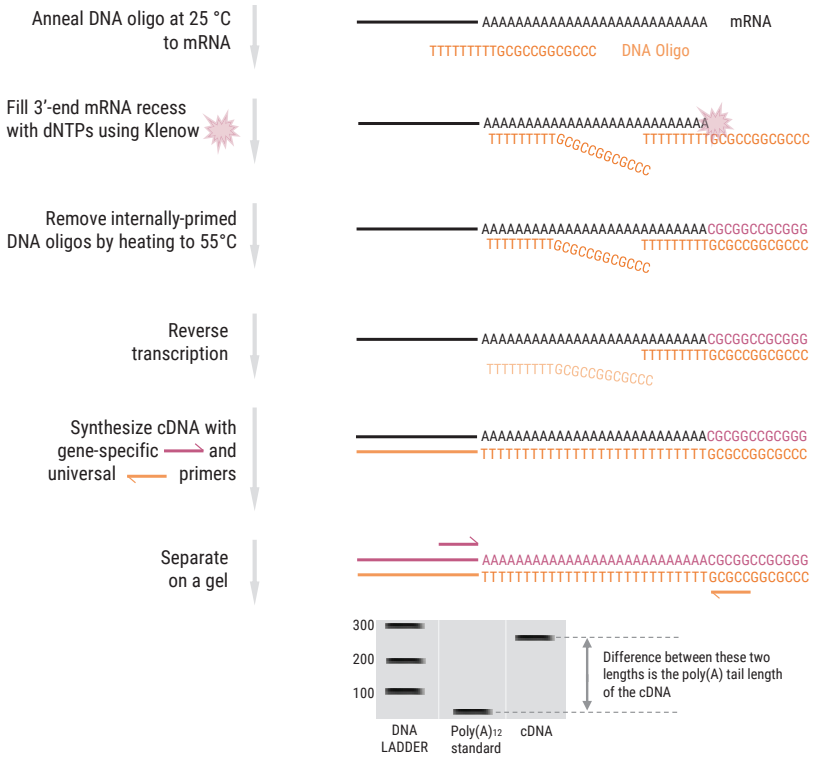


Fig. 4.1. The extension polyA-tail test (ePAT) for poly(A) tail profiling.

flow cell with fluorophore-tagged streptavidin which binds to the biotin-conjugated UTPs incorporated in the poly(A) in the earlier steps. Thus each cluster glows in proportion to the amount of biotin incorporated into the poly(A) tails which is proportional to the poly(A) tail length. Armed with the 36nt sequence proximal to the poly(A) tail, each measured poly(A) tail can be attributed to the gene from which it originated.

4.3.3 TAIL-seq

TAIL-seq is also a short-read sequencing method for poly(A) tail profiling [99]. Unlike PAL-seq, which uses incorporation of fluorophore-tagged streptavidin to indirectly gauge the poly(A) length, TAIL-seq directly sequences the poly(A) on the Illumina sequencer. In this method, the total RNA is depleted of rRNA and then the mRNA is ligated to biotinylated 3' adaptors, followed by RNases T1 fragmentation

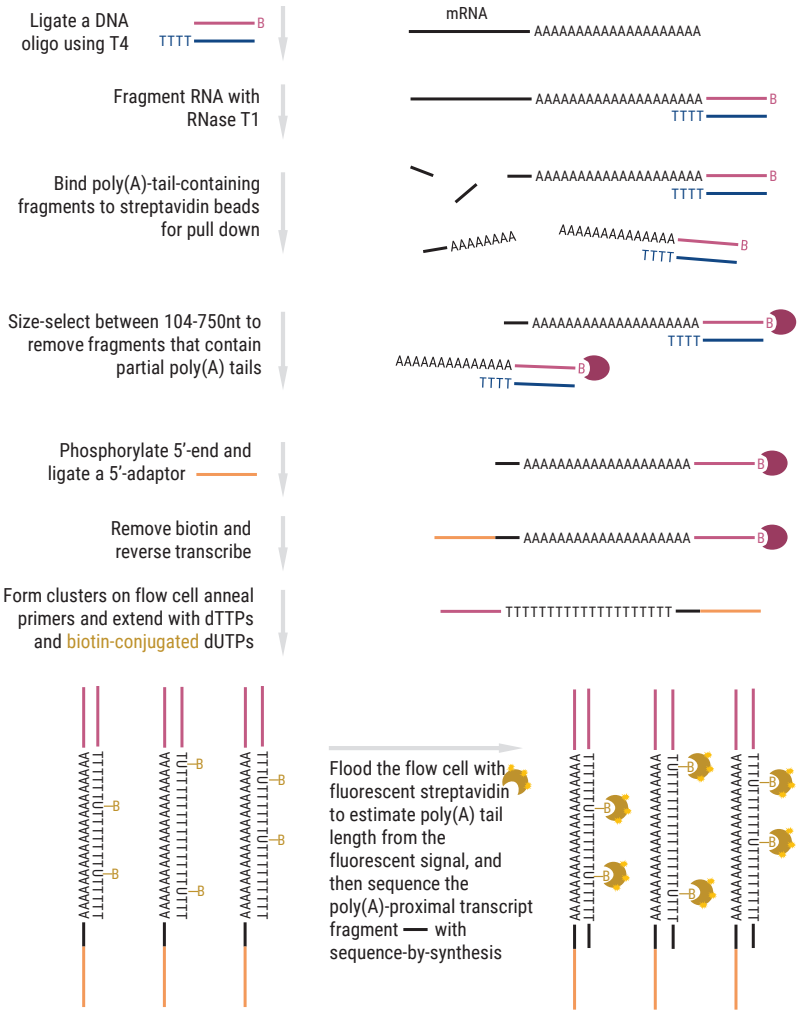


Fig. 4.2. Poly(A) profiling by sequencing (PAL-seq)

(Fig. 4.3). The fragments containing poly(A) tails are then pulled down using streptavidin beads and size selected to 500-1000nt. The 5' ends are phosphorylated, ligated to 5' adapters, reverse-transcribed, PCR-amplified, and finally paired-end sequenced. Read1 sequences 52 nt from the 5' end of the mRNA (which is used to identify the gene) while read2 sequence 251 nt from the 3' end (which is used for poly(A) tail length determination).

In TAIL-seq, standard basecalling routines of the Illumina sequencer cannot be used to basecall read 2. This is because, in long homopolymer T stretches, the phasing errors are very large, and the basecaller can output a T even when the current sequence cycle is sequencing a non-poly(T) part of the read next to the poly(T) tail. This causes an overestimate of the poly(A) tail length if Illumina's own basecalling routines are used. The authors of the study, therefore, used a machine learning approach to analyze sequencer images for cycles and correctly estimate the poly(A) tail length.

Tail-seq can also sequence poly(A) tails with uridine and guanine content at the 3'-end which makes it useful for studying non-homogenous poly(A) tails. If sequencing of only homogeneous poly(A) tails is required then a variation of TAIL-seq called mTAIL-seq (short for mRNA TAIL-seq) can be used [100].

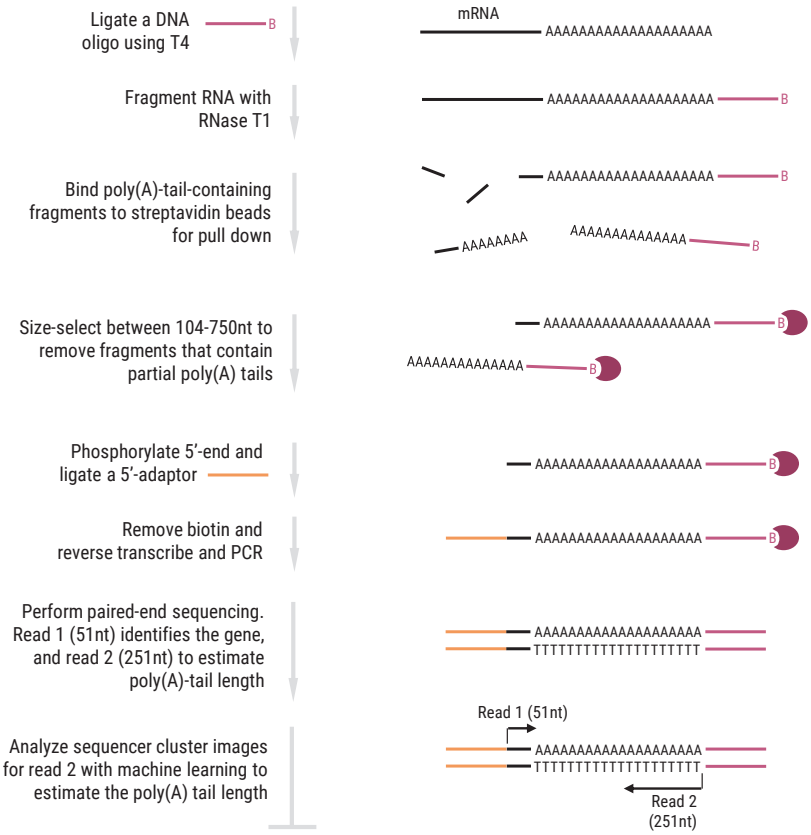


Fig. 4.3. TAIL-seq method for poly(A)-tail profiling

4.3.4 Poly(A) inclusive RNA isoform sequencing (PAIso-seq)

This method is based on PacBio sequencing and can sequence full-length transcript isoforms along with their poly(A) tails in cDNA [101]. In this method, the 3'-end of the RNA is extended with dNTPs by a Klenow fragment using a DNA oligo template with two Us incorporated in it (Fig. 4.4). The user enzymes then cleaves off the DNA oligo at the U site and the cleaved fragments are then removed. Next, reverse transcription is performed which leaves CCC in the reverse transcribed strand when the reverse transcriptase encounters the 5'-cap. Two DNA primers complementary to each other but one having a GGG overhang hybridize to the 5'-end of the RNA-DNA duplex and perform template switching. This is followed by PCR amplifications and SMRT bell adapter ligation at both ends. Sequencing yields a long read with tandem repeats of the forward and reverse reads and poly(A) and poly(T) segments. These copies in the read are analyzed to yield a consensus sequence and poly(A) tail length.

4.4 Limitations of existing poly(A)-tail profiling methods

The ePAT requires the use of a gene-specific primer, which makes this method intractable if the number of genes to be profiled is large. Furthermore, the resolution of gel is poor and hence the method does not yield sharp poly(A) tail length estimates.

PAL-seq is a technically complicated protocol to pull off successfully and specifically requires a now-defunct sequencer — the Illumina Genome Analyzer II. No attempt has been made to adapt this protocol to modern-day Illumina sequencers due to a number of low-level changes needed in the sequencer to make this protocol work.

The more recent TAIL-seq protocol captures only 51 nt of the poly(A)-proximal transcript region. When the different isoforms have the same poly(A) cleavage site then these different isoforms will have the same poly(A)-proximal sequence. Due to a lack of difference in the poly(A)-proximal sequence for these different isoforms, the measured poly(A) could not be uniquely assigned to one or the other isoform as shown in Fig 4.5. Thus isoform-specific poly(A)-tail assignment cannot be made in these cases.

Although full-length isoform and associated poly(A) tails can be studied with the PacBio-based PAIso-seq, conversion of RNA to cDNA is required, which makes it

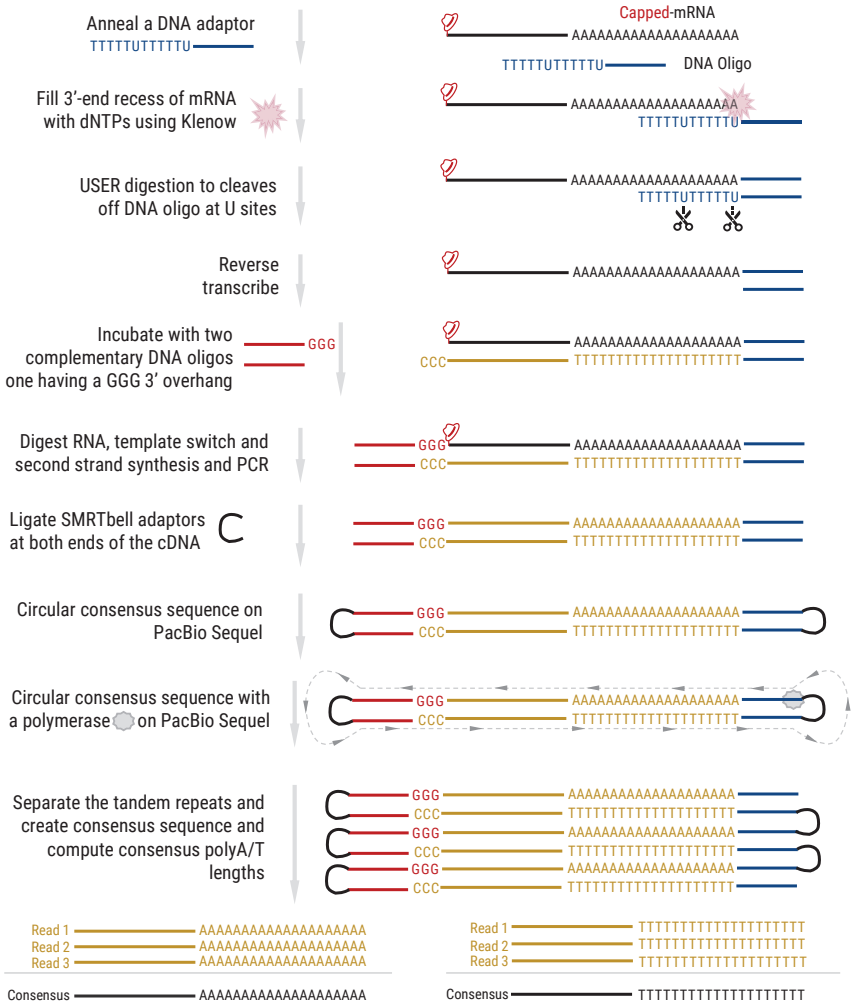


Fig. 4.4. PacBio-sequencing based PAIso-seq method which can be used to study poly(A) tails alongwith their full-length transcript isoforms in cDNA (not in native RNA)

impossible to study the poly(A) tails in conjunction with other RNA modifications such as the caps.

In summary, current methods do not allow us to study poly(A)-tails transcriptome-wide along with other important RNA features such as RNA modifications and cap structures. We aim to develop a Nanopore-based method that can be used to estimate poly(A) tail length potentially in combination with capable to study cap

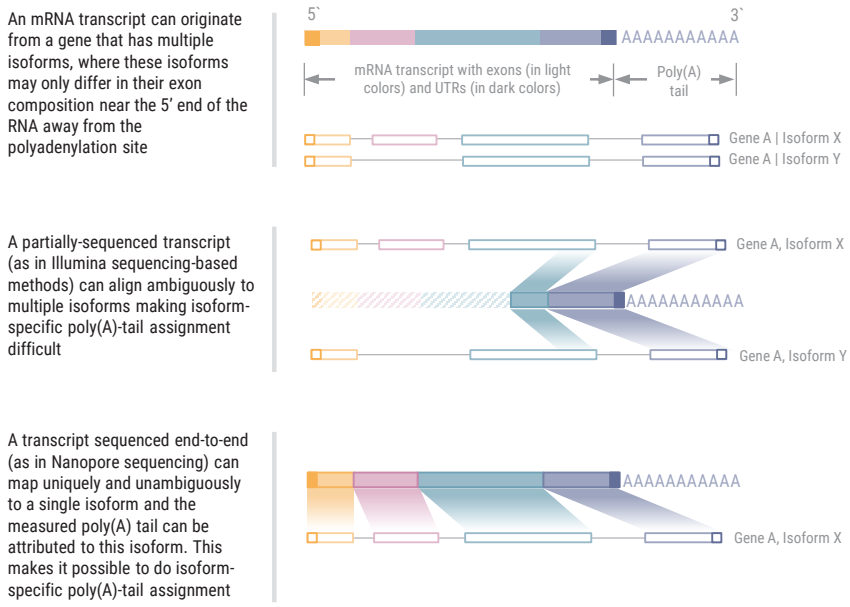


Fig. 4.5. Limitation of short-read sequencing methods for poly(A)-tail profiling. Poly(A) tail length estimates obtained from short-read sequencing-based methods make it difficult to assign the estimated poly(A) tail uniquely to a particular transcript isoform if the different isoforms have the same 3'-end sequence.

modifications — and possibly other RNA modifications in the future — which will prove instrumental in our understanding of the complex world of RNA.

4.5 Nanopore sequencing of poly(A) tails and the challenges involved

In nanopore sequencing, the native RNA molecule is sequenced end-to-end. Any nucleotide that goes through the pore creates a distinct current signature which, theoretically, makes it possible to decode its identity. This enables us to sequence full-length RNA transcript isoforms including their poly(A) tails, caps, and other modifications.

As a poly(A) tail is a homogenous stretch of adenosines, it creates a constant partial blockage of the pore current when going through the Nanopore. This results in a monotonous stretch of signal in the pore current — a tell-tale sign of a homopolymer

going through the pore. The longer the poly(A) tail, the longer this stretch of monotonous signal.

When the squiggle for the entire read including the poly(A) tail is basecalled, the basecaller predicts which base corresponds to which current samples in the squiggle. However, when the basecaller encounters the monotonous signal for the poly(A) tail, it predicts fewer A's compared to the actual number of A's in the poly(A) tail. This is because the monotonous poly(A) signal does not have any detectable transition when going from one adenosine base to the next. Thus a poly(A) tail — or any other homopolymer for that matter — is compressed in the basecalled output (Fig. 4.6a). If one were to estimate the poly(A) tail length from basecalled Nanopore data, the estimated poly(A) tail length will mostly be an underestimate of the true poly(A) tail length. The error between estimated and true poly(A) length is greater in longer poly(A) tails compared to shorter poly(A) tails.

4.6 Our method – *tailfindr* – in brief

To perform poly(A)-tail profiling in Nanopore data, I have created an R package called *tailfindr* (`https://github.com/adnaniazi/tailfindr`). In brief, *tailfindr* accurately estimates the true poly(A) tail length in a Nanopore read by first finding how long the monotonous stretch of poly(A) signal is time (samples), and then normalizing it by the average time that nucleotides of that read spent in the pore (Fig. 4.6b). In this way, *tailfindr* estimates how many poly(A)-tail bases are encoded in the homopolymer stretch of the read's squiggle.

The quality of the estimate is dependent on how accurately the borders of the homopolymer stretch of the signal are found in the noisy and messy Nanopore signal, and then how accurately the normalizer (average dwell time of a nucleotide in every read) is computed.

Appended next is my published work that mentions all the relevant details of the *tailfindr* algorithm and the results obtained. This work shows that *tailfindr* can not only correctly predict poly(A) tail lengths in RNA, but also in unamplified DNA as well. After this publication, we worked some more to make tailfindr work on amplified cDNA as well (more on that in the later sections).

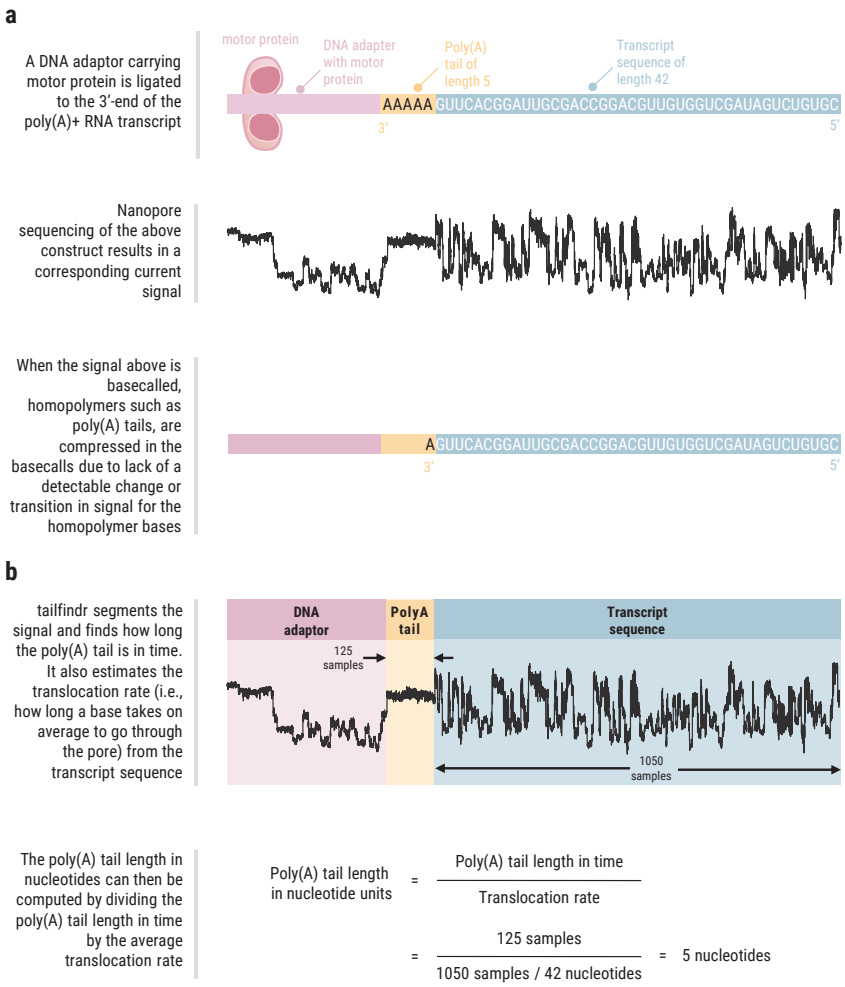


Fig. 4.6. **Nanopore sequencing of poly(A) tails. a)** Homopolymer compression in Nanopore basecalls prevents us from finding the precise poly(A) tail length. **b)** *tailfindr* uses the information encoded in the length of the poly(A) tail signal to infer the poly(A) tail length in nucleotide units.

4.7 Our published work on *tailfindr* with more details

BIOINFORMATICS

tailfindr: alignment-free poly(A) length measurement for Oxford Nanopore RNA and DNA sequencing

MAXIMILIAN KRAUSE,[1,2,3] ADNAN M. NIAZI,[1,3] KORNEL LABUN,[1] YAMILA N. TORRES CLEUREN,[1] FLORIAN S. MÜLLER,[1] and EIVIND VALEN[1,2]

[1]Computational Biology Unit, Department of Informatics, University of Bergen, 5008 Bergen, Norway
[2]Sars International Centre for Marine Molecular Biology, University of Bergen, 5008 Bergen, Norway

ABSTRACT

Polyadenylation at the 3′-end is a major regulator of messenger RNA and its length is known to affect nuclear export, stability, and translation, among others. Only recently have strategies emerged that allow for genome-wide poly(A) length assessment. These methods identify genes connected to poly(A) tail measurements indirectly by short-read alignment to genetic 3′-ends. Concurrently, Oxford Nanopore Technologies (ONT) established full-length isoform-specific RNA sequencing containing the entire poly(A) tail. However, assessing poly(A) length through base-calling has so far not been possible due to the inability to resolve long homopolymeric stretches in ONT sequencing. Here we present *tailfindr*, an R package to estimate poly(A) tail length on ONT long-read sequencing data. *tailfindr* operates on unaligned, base-called 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 assess *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.

Keywords: poly(A) tail; nanopore sequencing; cDNA; R package

INTRODUCTION

The poly(A) tail is a homopolymeric stretch of adenosines at the 3′-end of the majority of eukaryotic mRNAs. These tails are necessary for the nuclear export of mature mRNAs (Hector et al. 2002; Bear et al. 2003; Fuke and Ohno 2008) and influence mRNA stability and translation (Eckmann et al. 2011).

The poly(A) tail is generated directly after transcription by the nontemplated addition of adenosines to the mRNA 3′-end, a process catalyzed by nuclear Poly(A)-polymerases (for review, see Millevoi and Vagner 2010). The initial length of poly(A) tails generated by this process has been estimated to be around 250 nt in vitro (Darnell et al. 1971; Edmonds et al. 1971; Raabe et al. 1991, 1994). After nuclear export, poly(A) length is dynamically regulated by the interplay of 3′-to-5′ degradation through exoribonucleases, poly(A) tail stabilization via poly(A) tail binding proteins, and elongation by cytoplasmic Poly(A)-polymerases (Diez and Brawerman 1974; Clegg and Pikó 1982; Hake and Richter 1994; Mendez et al. 2000; Read et al. 2002). While it has been shown that the poly(A) tail has a regulatory role, it is still not fully understood whether a specific length allows for specific regulatory outcomes (Jalkanen et al. 2014). A minimal poly(A) tail is needed to prevent quick 3′-to-5′ exonuclease degradation (Ford et al. 1997), yet hyperadenylated RNAs are marked for fast RNA degradation in the nucleus (Bresson and Conrad 2013; Jalkanen et al. 2014). Besides regulating RNA degradation, poly(A) tail length has been shown to correlate with translation efficiency during embryonic development (Beilharz and Preiss 2007; Subtelny et al. 2014), possibly by favoring a closed-loop structure of the mRNA. However, recent studies using *C. elegans* have proposed that shorter poly(A) tails are more actively translated, while longer tails are refractory to translation (Lima et al. 2017).

To understand the regulatory role of poly(A) tails, it is crucial to be able to measure poly(A) tail length genome-wide with transcript isoform resolution. Up until recently, estimating poly(A) tail lengths was restricted to transcript-specific measurements that relied on PCR and/or on laborious northern blotting techniques (Nilsen 2015). These techniques suffer from low throughput, high

[3]These authors contributed equally to this work.
Corresponding author: eivind.valen@uib.no
Article is online at http://www.majournal.org/cgi/doi/10.1261/rna.071332.119. Freely available online through the *RNA* Open Access option.

workload and possible technical artifacts due to amplification (Hite et al. 1996; Murray and Schoenberg 2008; Hommelsheim et al. 2014). Only recently a set of studies implemented short-read sequencing strategies to study poly(A) tail length in a transcriptome-wide manner (Chang et al. 2014; Subtelny et al. 2014; Lim et al. 2016; Balagopal et al. 2017; Lima et al. 2017; Woo et al. 2018). While these studies allowed a thorough understanding of poly(A) tail lengths throughout the transcriptome for the first time, they are technically restricted to a specific size of poly(A) tails depending on sample enrichment and sequencing strategy. Additionally, most of these techniques rely on PCR amplification of the poly(A) tail region, which might lead to amplification artifacts that affect poly(A) length measurements as well as quantitative comparisons between long and short poly(A) tails (Hite et al. 1996; Murray and Schoenberg 2008; Hommelsheim et al. 2014). Finally, and more importantly, these techniques can only indirectly identify the transcript linked to the poly(A) by alignment of short sequences representing the RNA 3′-ends. Thus it is challenging and in many cases virtually impossible to assign poly(A) tail measurements to specific transcript isoforms.

Oxford Nanopore Technologies' (ONT) native RNA sequencing strategy allows for the sequencing of full-length mRNA molecules without amplification artifacts (Jain et al. 2016). The standard library preparation protocol retains the full poly(A) tail in the molecule to be sequenced, making it possible to obtain isoform-specific poly(A) tail length estimates in a transcriptome-wide manner (Garalde et al. 2018). However, current base-callers do not perform well on long homopolymer RNA and DNA stretches, resulting in the length of poly(A) tails not being accurately reported (Rang et al. 2018).

Here we present *tailfindr*, an R tool that estimates poly(A) tail length from individual reads directly from ONT FAST5 raw data. *tailfindr* is able to estimate poly(A) tails from both RNA and DNA reads, including DNA reverse-complement reads containing poly(T) stretches. *tailfindr* uses the raw data without prior alignment as input, and estimates the length based on normalization with the read-specific nucleotide translocation rate. We validate the performance of *tailfindr* on a set of RNA and DNA molecules with defined poly(A) tail lengths. *tailfindr* operates on the output of widely used as well as the most recent ONT base-calling applications (flip-flop model).

RESULTS

tailfindr estimates poly(A) tail length from base-called ONT native RNA sequencing

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 (Fig. 1A; Garalde et al. 2018). Indeed, long stretches of monotonous low-variance raw signal corresponding to poly(A) tails can be observed at the beginning of most reads (Fig. 1B). However, since base-calling relies on fluctuations of the raw signal, these low-variance sections are poorly decoded into the correct nucleobase sequence (Rang et al. 2018).

To identify the region corresponding to the expected poly(A) tail, we apply thresholding to normalized raw data, refine the boundaries of possible poly(A) stretches based on raw signal slope, and normalize by the read-specific nucleotide translocation rate (Fig. 1C, for details see Materials and Methods). *tailfindr* provides the user with a tabular output containing the unique read-ID, the estimated poly(A) tail length and all factors extracted from the raw data that are needed to calculate the poly(A) tail estimate (Supplemental Fig. S4A). This allows for custom filtering of the acquired poly(A) measurements by the user. Optionally, *tailfindr* allows the user to generate read-specific plots displaying the raw data and all signal derivatives generated in the process to estimate poly(A) tail length (Supplemental Fig. S4B). To test the performance of our algorithm, we pooled six barcoded in vitro transcribed eGFP RNA samples with different poly(A) tail lengths (10, 30, 40, 60, 100, and 150 nt) and sequenced the pooled samples with ONT's native RNA sequencing kit in two replicates. Because the barcodes that define molecules with specific poly(A) length are located at the 5′-end of the eGFP RNA, only reads that cover the full RNA molecule from 5′-end to 3′-end were considered for the analysis. After barcode demultiplexing, the estimated poly(A) tail lengths for each length group overall match the expected poly(A) tail length, with the exception of eGFP with a poly(A) tail of 10 nt (Fig. 1D). While the molecules with an expected poly(A) length of 10 nt were measured with a mode of 21, the mode of poly(A) measurements of all other barcoded RNA molecules matches well with the expected poly(A) lengths (30 nt: 33; 40 nt: 41; 60 nt: 59; 100 nt: 91; 150 nt: 136). However, even though the majority of sequences show the expected poly(A) tail length, the standard deviation of poly(A) tail measurements is relatively high (coefficient of variation between 45% and 79%, see Table 1). This is not a result of poor poly(A) tail boundary assignment, as poly(A) tail end coordinates defined by *tailfindr* match with coordinates from alignment of the expected adjacent eGFP sequence with a precision of around 2 nt (Supplemental Fig. S6; Supplemental Discussion). Both the high accuracy of poly(A) length estimation as well as the variation around the average is consistent across replicates (Supplemental Fig. S1A). Furthermore, the estimates are robust across different sequencing conditions, as a third replicate performed with the new Library preparation kit (SQK-RNA002) and omitting the optional Reverse Transcription reaction resulted in similar poly(A)

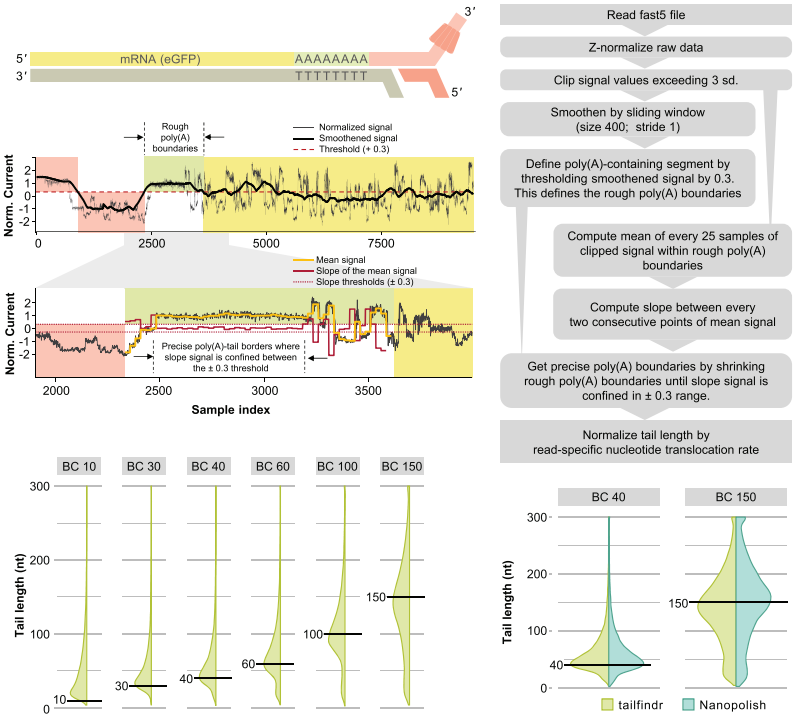


FIGURE 1. Workflow and performance of *tailfindr* on ONT RNA data. (*A*) Schematic representation of Oxford Nanopore RNA sequencing. The motor protein (red) is attached to the native RNA molecule (yellow) at the 3′-end by T4 DNA ligation via a double-stranded adapter (light red) with oligo-T overhang. The motor protein thus feeds the RNA strand to the pore from 3′ to 5′. (*B*) Representative signal tracks from eGFP-RNA sequencing. *Upper* panel shows normalized signal data calculated by *z*-normalization through *tailfindr* (gray, workflow box 3) with smoothened signal track (black, workflow box 4). Red background indicates ONT adapter signal and green background represents rough borders of poly(A) signal as identified by thresholding (workflow box 5), whereas yellow background highlights signal corresponding to potential RNA sequence. *Lower* panel shows zoom on potential poly(A) region with signal track for the mean of clipped, normalized raw data (yellow, workflow box 6) and slope of the mean signal track (red, workflow box 7), which are used to refine poly(A) boundaries (dashed *vertical* lines, workflow box 8). (*C*) Schematic workflow of data processing by the *tailfindr* algorithm for ONT native RNA sequencing data leading to signal tracks shown in *B* and ultimately poly(A) estimation. (*D*) Vertical density plots of poly(A) length estimation on in vitro transcribed eGFP-RNA molecules with known poly(A) tail length (from *left* to *right*: 10, 30, 40, 60, 100, and 150 nt labeled as BC10, BC30, BC40, BC60, BC100, and BC150, respectively). *Horizontal* black lines demarcate expected poly(A) length for individual barcodes. Poly(A) estimates exceeding 300 nt were set to 300 prior to plotting. (*E*) Vertical density plots of poly(A) length estimation from *tailfindr* (light green) and Nanopolish (turquoise) on in vitro transcribed eGFP-RNA with poly(A) length of 40 or 150 nt (labeled as BC40 and BC150, respectively). Poly(A) estimates exceeding 300 nt were set to 300 prior to plotting. Comparison of all known poly(A) lengths can be found in Supplemental Figure S2B.

length measurements (Supplemental Fig. S1B). Thus, while the poly(A) estimation suffers from significant variation, the length of most barcoded molecules can be successfully estimated by the use of *tailfindr* on ONT RNA sequencing.

While this study was in progress, another tool estimating poly(A) tail lengths from ONT RNA data was developed (Workman et al. 2018). Instead of estimating poly(A) tails from base-called data directly, this tool requires read

TABLE 1. Summary statistics for poly(A) estimates on direct RNA sequencing experiments

Barcode	Read count	Mean	Median	Mode	Std dev	CoV
10	47,036	53.84	40.47	21	42.47	0.79
30	45,637	56.44	44.96	33	37.57	0.67
40	26,317	63.33	52.72	41	38.66	0.61
60	59,591	79.03	69.49	59	43.97	0.56
100	36,390	108.53	102.38	91	49.57	0.46
150	29,267	138.29	139.56	136	62.83	0.45

alignment information for the definition of the poly(A) tail segment. To compare whether our algorithm results in similar performance, we measured poly(A) tail lengths from Nanopolish and *tailfindr* on different barcoded eGFP molecules. Our analysis showed that both tools matched well in their estimated poly(A) tail lengths, as exemplified in Figure 1E for 40 and 100 nt poly(A) tail length (full comparison including analyses on published data set by Workman et al. 2018 in Supplemental Fig. S2). However, while both tools agreed in the majority of cases on the definition of poly(A) segments, we routinely observed slightly higher estimates from Nanopolish which can be attributed to differences in normalization (Supplemental Fig. S3A,B). In conclusion, *tailfindr* accurately defines poly(A) tail segments in ONT native RNA sequencing data and provides similar estimates to Nanopolish while only using basecalled data files as input.

Poly(A) and poly(T) tail length can be estimated from ONT DNA sequencing data

ONT native RNA sequencing is lower in both quantity and quality compared to cDNA sequencing approaches and relies on large amounts of starting material [500 ng of poly(A)-selected RNA, Oxford Nanopore Technologies 2018a, 2019]. Therefore, cDNA sequencing approaches that retain the full-length poly(A) tail would enable studies where material is scarce as well as increase statistical power of poly(A) tail estimates. We thus aimed to expand *tailfindr* to operate on ONT DNA sequencing approaches as well. Since standard cDNA approaches result in double-stranded DNA, both poly(A) as well as poly(T) stretches are present in ONT sequencing reads. During cDNA sequencing both of these strands are threaded through the pore separately from 5′ to 3′ (Fig. 2A). Indeed we observe homogenous stretches of raw signal both at the beginning [poly(T) tail] as well as at the end [poly(A) tail] of individual raw read sequences [example for poly(T)-containing read in Supplemental Fig. S5B].

We extended our algorithm to accommodate ONT DNA sequencing data output (Fig. 2B). Running the algorithm provides the user with a tabular output of tail length measurements as well as optional raw data plots (Supplemental Fig. S5). We account for the double-stranded nature of DNA and define the read type [poly(A)- or poly(T)-containing] by making use of known sequence motifs in Nanopore adapters (details in Materials and Methods). We tested the performance of the DNA-specific *tailfindr* algorithm on PCR products of eGFP coding sequence with known poly(A)/(T) length in two replicates, similar to the spike-ins generated for native RNA sequencing. As shown in Figure 2C, the DNA-specific *tailfindr* approach resulted in estimated poly(A) and poly(T) lengths close to the expected length for barcoded molecules [mode of distribution for 10 nt: 10; 30 nt: 29; 40 nt: 39; 60 nt: 59; 100 nt: 97 for poly(A) and 110 for poly(T); 150 nt: 148 for poly(A) and 155 for poly(T)]. These estimates were consistent across replicates from different Library preparation kits (Supplemental Fig. S7) and the poly(A)/(T) end coordinates matched with coordinates of the alignment of adjacent eGFP sequence (Supplemental Fig. S6). For all poly(A)/(T) tail lengths bigger than 10 nt, a small subpopulation of reads with shorter estimated tails could be observed, possibly due to amplification artifacts that connect barcoded eGFP sequence with wrong poly(A) tail lengths (see Supplemental Discussion; Supplemental Fig. S8).

Next we compared poly(A)-length estimates from DNA and native RNA sequencing. We observed that DNA sequencing results in significantly more precise estimation of poly(A) tail length, mainly due to fewer outliers toward longer poly(A) tail lengths (Fig. 2D). Especially the shortest poly(A) tail length (10 nt) could be estimated more correctly with DNA sequencing [mode of poly(A) length estimation 10 in DNA vs. 22 in RNA sequencing]. On other poly(A) lengths, the mode of poly(A) estimation does not differ dramatically, but the precision is significantly higher for DNA sequencing (coefficient of variation between 33% and 50% in DNA sequencing, Table 2). In summary, *tailfindr* is able to estimate poly(A) and poly(T) tail size from ONT DNA sequencing with significantly higher precision compared to ONT RNA sequencing estimates.

tailfindr is compatible with flip-flop model base-calling

While this manuscript was in preparation, ONT released a new DNA base-calling strategy based on flip-flop models. Flip-flop model base-calling screens the raw signal by comparing probabilities to either stay in the same nucleotide state or change to a new state. Additionally, the raw data is read by averaging over two sample points only, as opposed to averaging over five sample points in standard model base-calling. These improvements have been shown to result in higher quality base-calling, and more importantly to increase the base-call fidelity over homopolymer sequences (Oxford Nanopore Technologies

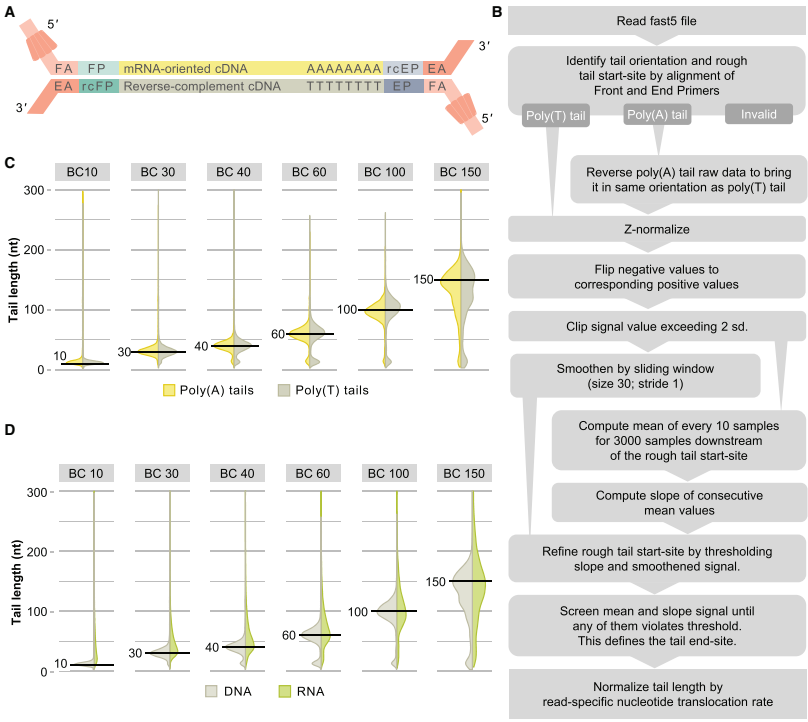


FIGURE 2. Workflow and performance of *tailfindr* on ONT DNA sequencing data. (*A*) Schematic representation of Oxford Nanopore DNA sequencing. In cDNA approaches, amplification is ensured by oligo-dT-aided anchoring of the end primer (EP, blue) and addition of front primer sequence (FP, light green) by template switching during reverse transcription. The motor protein (red) is attached to the double-stranded DNA molecules at both ends by T4 DNA ligation. The front adapter (FA) bears the motor protein, while the end adapter (EA) is a short complementary oligo that will ultimately appear at the 3′-end of resulting sequences. Both DNA strands are sequenced from 5′ to 3′. Thus, oligo-dT stretches will be present at the beginning of raw data, while oligo-dA stretches appear at the end. (*B*) Schematic workflow for ONT DNA sequencing data processing by the *tailfindr* algorithm. (*C*) Vertical density plot of poly(A) (yellow) and poly(T) (gray) length estimates on PCR-amplified eGFP coding sequence with known poly(A) length. *Horizontal* black lines demarcate expected poly(A) length for individual barcodes (from *left* to *right*: 10, 30, 40, 60, 100, and 150 nt labeled as BC10, BC30, BC40, BC60, BC100, BC150, respectively). (*D*) Vertical density plot of poly(A)/(T) length estimates on DNA sequences (gray) and poly(A) length estimates on RNA (light green) (from *left* to *right*: 10, 30, 40, 60, 100, and 150 nt labeled as BC10, BC30, BC40, BC60, BC100, BC150, respectively).

2018b). So far, flip-flop model base-calling is only available for ONT DNA sequencing data.

We implemented changes in *tailfindr* to account for the updates in flip-flop model raw data output. As expected, flip-flop model base-calling detects more nucleotide translocations (called "moves") over poly(A) stretches when compared to standard model base-calling (Fig. 3A, yellow highlights). To test whether the detected moves agree with expected poly(A)/(T) length, we plotted the moves from either standard model base-calling (Fig. 3B) or flip-flop model base-calling (Fig. 3C) on eGFP-PCR products with 30 or 100 nt poly(A)/(T) tail length. While flip-flop model base-calling resulted in significantly more detected moves over poly(A)/(T) tail sections compared to standard model

TABLE 2. Summary statistics for poly(A)/(T) estimates on DNA sequencing experiments

Barcode	Read type	Read count	Mean	Median	Mode	Std dev	CoV
10	poly(A)	5462	21.27	13.06	10	25.56	1.20
	poly(T)	11,072	16.23	12.12	10	19.81	1.22
30	poly(A)	13,063	34.44	31.21	29	16.90	0.49
	poly(T)	17,087	31.65	29.98	29	15.15	0.48
40	poly(A)	6946	42.10	40.29	39	17.45	0.41
	poly(T)	13,811	39.03	39.48	39	16.64	0.43
60	poly(A)	8261	57.69	59.14	59	18.90	0.33
	poly(T)	10,072	53.27	59.11	59	24.56	0.46
100	poly(A)	3015	93.59	96.82	97	23.11	0.25
	poly(T)	3166	91.70	101.18	110	34.41	0.38
150	poly(A)	1767	126.09	138.46	148	41.76	0.33
	poly(T)	2535	138.29	130.15	155	50.31	0.42

base-calling, the number of moves still severely underestimates existing poly(A)/(T) lengths. Thus even with improved homopolymer base-call fidelity, external tools are needed to correctly measure poly(A) tail lengths. We used *tailfindr* to compare poly(A) and poly(T) tail measurements from the same sequencing reads base-called either with flip-flop or standard models, and could show that the estimated poly(A)/(T) tail length is highly correlated between the two base-calling approaches [$R = 0.93$ for poly(A); $R = 0.97$ for poly(T); Fig. 3D,E]. We thus conclude that *tailfindr* operates on both standard and the most recent flip-flop model base-calling, and provides accurate poly(A)/(T) length estimates for ONT DNA sequencing approaches.

DISCUSSION

Polyadenylation at the 3′-end is understood to be a major regulator of mRNA (Hector et al. 2002; Bear et al. 2003; Fuke and Ohno 2008; Eckmann et al. 2011). While the poly(A) length of mRNAs has been under investigation since the 1970s (Brawerman 1973; Morrison et al. 1973; Groner et al. 1974; Merkel et al. 1976), transcriptome-wide analysis of poly(A) tail lengths have only recently emerged. The advent of Oxford Nanopore Technologies (ONT) native RNA sequencing technology now allows direct sequencing of full-length mRNA molecules, which intrinsically contain their full poly(A) tail, unbiased by potential amplification artifacts (Jain et al. 2016). However, even the most recent updates in base-calling tools do not perform well over long homopolymeric sequence stretches (Oxford Nanopore Technologies 2018b; Rang et al. 2018).

In this work we present *tailfindr*, a versatile R tool that allows estimation of poly(A) tail lengths from base-called ONT long-read sequencing data from both native RNA and DNA sequencing approaches. *tailfindr* operates on data from all current and previous ONT base-calling strategies that produce an events/move table in the resulting FAST5 files. We show that *tailfindr* is able to detect the poly(A) tail boundaries of in vitro transcribed eGFP RNA molecules and estimate their lengths based on read-specific raw data normalization. For molecules with known poly(A) tails from 30 nt up to 150 nt the estimates match well with the expected lengths (Fig. 1D), however the shortest poly(A) tail (10 nt) was estimated to have longer tails than expected. We believe that this bias can be explained by sample contamination of this RNA molecule during preparations, or by inefficient oligo-dT sequencing adapter ligation to poly(A) tail stretches at or below 10 nt. Consistent with the latter explanation we observed that the barcoded 10 nt RNA molecule was underrepresented in the RNA sequencing libraries compared to input quantities (Table 3). Overall, *tailfindr* correctly estimates poly(A) tail lengths of in vitro transcribed RNA over a wide range of lengths.

We further show that *tailfindr* poly(A) tail estimates agree closely with a recently developed tool that relies on the prior mapping of the data (Workman et al. 2018). While poly(A) tail boundaries in the raw signal are found to be essentially the same with the two different approaches (Supplemental Fig. S2C,D), the final calculated poly(A) tail lengths differ slightly (Supplemental Fig. S2A). Specifically, *tailfindr* estimates short poly(A) stretches slightly longer than Nanopolish, while long poly(A) stretches result in shorter estimates in *tailfindr*. These differences can be explained by a different calculation of the average nucleotide translocation rate (Supplemental Fig. S2B) which is used to normalize raw poly(A) tail measurements. Nanopolish normalizes by calculating the read-specific median of the samples per nucleotide after removing 5% of the translocation rate outliers. We observed that this normalization is resulting in correct poly(A) estimation in RNA, but not DNA sequencing approaches (further discussed in Supplemental Discussion). Instead, we normalize by the read-specific geometric mean of samples per nucleotide without a specific arbitrary outlier threshold. Another difference between the tools is that *tailfindr* does not need any sequence

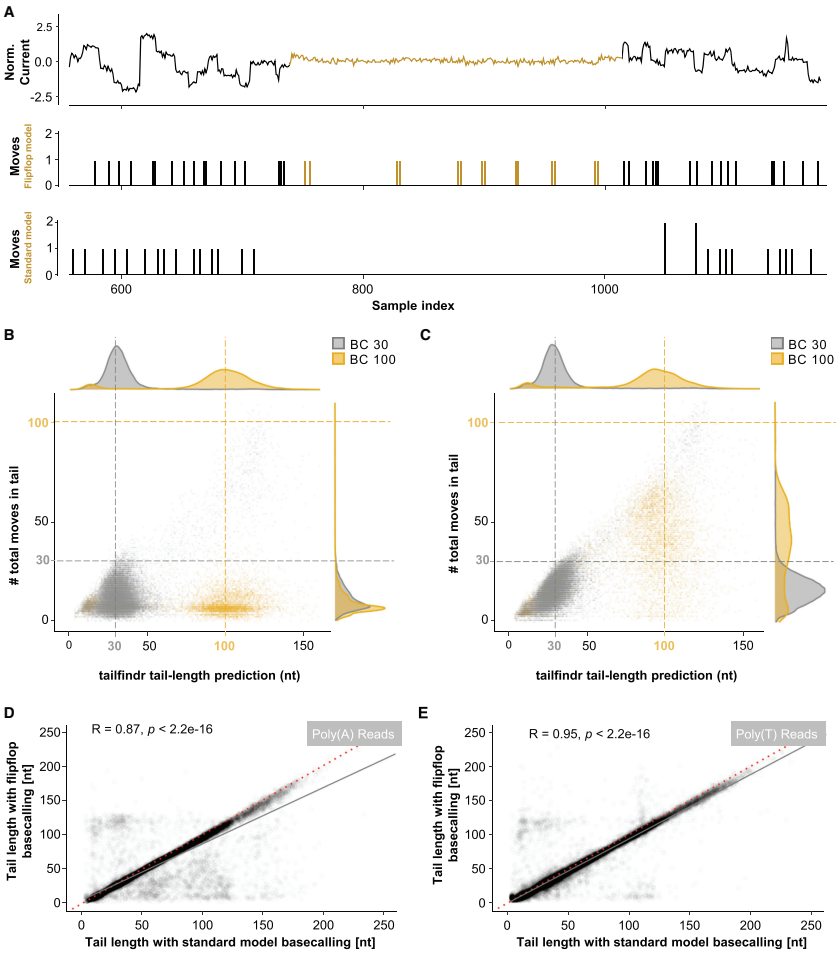


FIGURE 3. Differences in poly(A) tail estimation for standard and flip-flop model base-calling. (*A*) Representative raw data squiggle of PCR-amplified eGFP coding sequence over the identified poly(A) tail region (colored yellow) with associated moves (shifts in raw data representing possible nucleotide translocations) in both flip-flop (*middle* panel) and standard model base-calling (*bottom* panel). Flip-flop model base-calling detects moves with higher resolution, and calls more moves, especially in the poly(A) tail region (yellow). (*B*,*C*) Scatter plot of estimated poly(A)/(T) tail length (*x*-axis) and moves detected with standard (*B*) or flip-flop model base-calling (*C*) on PCR-amplified eGFP coding sequence with poly(A) length of 30 nt (gray) and 100 nt (yellow). Colored dashed lines indicate expected poly(A) length. (*D*,*E*) Scatter plot of poly(A) (*D*) or poly(T) (*E*) tail length estimated from PCR-amplified eGFP coding sequence with different poly(A) tail lengths that were base-called either with standard (*x*-axis) or flip-flop models (*y*-axis). (R, p by Pearson correlation). Red dashed line indicates $x = y$; gray line indicates linear fit.

TABLE 3. Overview of read counts of sequencing experiment replicates

	DNA Replicate 1 (SQK-LSK108)		DNA Replicate 2 (SQK-LSK109)		RNA Replicate 1 (SQK-RNA001 +RT)	RNA Replicate 2 (SQK-RNA001 +RT)	RNA Replicate 3 (SQK-RNA002 −RT)
Type	Poly(A)	Poly(T)	Poly(A)	Poly(T)	Poly(A)	Poly(A)	Poly(A)
BC 10	4884	8572	578	2500	3148	40,425	3463
BC30	11,953	13,898	1110	3189	7724	28,557	9356
BC40	6375	11,294	571	2517	12,044	279	13,994
BC60	7293	7733	968	2339	11,679	33,222	14,690
BC100	2619	2357	396	809	5439	21,120	9831
BC150	1502	1929	265	606	5219	16,777	7271

Data can be found in the ENA archive, study no. PRJEB31806.

data preprocessing, as it only requires base-called FAST5 files with an events table as input. This allows for poly(A) tail studies independent of any other tool than the essential base-caller, which would allow for an integration of the *tailfindr* algorithm into the base-calling procedure. This in turn makes it possible to assign poly(A) tail lengths to individual reads in parallel to the sequencing procedure, making live poly(A) tail analysis feasible.

In comparison to recent short-read sequencing-based strategies to measure poly(A) tails, methods using ONT sequencing are currently less precise. Short-read sequencing approaches promise poly(A) measurements with just a few bases of deviation due to cyclic incorporation of nucleotides and integration of the fluorescence signal of multiple molecules toward one single base-call (Chang et al. 2014; Lim et al. 2016; Balagopal et al. 2017; Lima et al. 2017; Woo et al. 2018). In contrast, ONT long-read sequencing measures individual single-stranded molecules, and single nucleotide changes are detected based on subtle changes in measured current levels. More importantly, the raw signal for ONT sequencing does not change over a homopolymeric region, making single-event detection almost impossible. Thus, ONT poly(A) length estimation relies on normalization of variable data taken from single-molecule measurements. Most of the variation observed in *tailfindr* poly(A) estimation thus comes from the sequencing process. However, the sequencing chemistry as well as the properties of the motor protein is under constant development. It is thus conceivable that in the near future an increase in speed and robustness of translocation rates can be observed, which will have a positive impact on poly(A) tail estimation (Oxford Nanopore Technologies 2018b). Updates in both sequencing chemistry as well as base-calling strategies can dramatically change the appearance and data obtained by base-calling. This is exemplified by the differences observed for base-calling the same data with standard and flip-flop models (Fig. 3; Supplemental Discussion). While these changes could in theory render the described algorithms imprecise or worst nonfunctional, future changes are more likely to reduce the variability and increase the precision of the nucleotide translocation rate. This would address the methods' current weakness and result in increased accuracy for poly(A) tail estimation using our *tailfindr* algorithm.

While currently not as precise in measuring poly(A) tails, ONT long-read sequencing approaches have unique advantages over short-read sequencing approaches. First, ONT sequencing is intrinsically a single-molecule technique. Second, RNA sequencing approaches are amplification-free, avoiding the emergence of possible amplification artifacts. Third, since the native molecule is sequenced as it comes from the specimen, additional features of the RNA can be measured directly, as was shown for RNA modifications (Viehweger et al. 2018; Workman et al. 2018). Fourth, and most importantly, long-read sequencing allows direct assignment of transcript isoforms to single molecules without bioinformatics post-processing, making truly isoform-specific measurements of poly(A) tail lengths possible. Additionally, ONT sequencing allows to study features of 5′-end and 3′-end events of the same molecule in conjunction with the poly(A) tail length. Together, ONT sequencing in conjunction with *tailfindr* poly(A) estimation offers great potential to combine the study of poly(A) tail length and other RNA features with transcript-isoform specificity in one assay.

Beyond ONT RNA sequencing applications, *tailfindr* is the first tool to show that poly(A) tails can be measured in ONT DNA sequencing. For DNA sequencing approaches *tailfindr* handles the most up-to-date base-calling strategy using flip-flop model base-calling (Fig. 3), making *tailfindr* compatible with all recently produced data sets. Interestingly, poly(A) estimation from ONT DNA sequencing is far more precise compared to measurements of similar RNA molecules (Fig. 2D). This is likely explained by a faster and more robust translocation rate with less likelihood for stochastic stalling during sequencing.

tailfindr makes it possible to design specific cDNA library preparation protocols that retain the full poly(A) tail in ONT sequencing approaches. This strategy has recently been shown to allow further insights into poly(A) tail regulation based on PacBio long-read sequencing (Legnini

TABLE 4. DNA oligos for the design of poly(A)-tailed eGFP constructs

Name	Sequence
BC10-eGFP	ATTTAGGTGACACTATAGCGCTCCATGCAAACCTGTCTGCAGATCTCTTGCCGTCGCC
BC30-eGFP	ATTTAGGTGACACTATAGCGCTCCATGCAAACCTGTCCTCGAAGCATTGTAAGTCGCC
BC40-eGFP	ATTTAGGTGACACTATAGCGCTCCATGCAAACCTGTCAACGGTAGCCACCAAGTCGCC
BC60-eGFP	ATTTAGGTGACACTATAGCGCTCCATGCAAACCTGTCTGCACGAGATTGATGGTCGCC
BC100-eGFP	ATTTAGGTGACACTATAGCGCTCCATGCAAACCTGTCGACACATAGTCATGGGTCGCC
BC150-eGFP	ATTTAGGTGACACTATAGCGCTCCATGCAAACCTGTCCATGAGTGCTGAGCTGTCGCC
poly(A) Bfo1 rev	GAGTCCGGGCGGCGCTTTTTTTTTT
SP6 Bfo1 fw	ATTTAGGTGACACTATAGCGATCCATGC
eGFP_pA_10_rev	GCGGCCGCTTTTTTTTTTCTACTTGTACAGCTCGTCCATGC
eGFP_pA_30_rev	GCGGCCGCT(x30)CTACTTGTACAGCTCGTCCATGC
eGFP_pA-40_rev	GCGGCCGCT(x40)CTACTTGTACAGCTCGTCCATGC
eGFP_pA_60_rev	GCGGCCGCT(x60)CTACTTGTACAGCTCGTCCATGC
eGFP_pA_100_rev	GCGGCCGCT(x100)CTACTTGTACAGCTCGTCCATGC
eGFP_pA_150_rev	GCGGCCGCT(x150)CTACTTGTACAGCTCGTCCATGC

et al. 2019). ONT cDNA sequencing has the advantage to yield approximately 10× more data per library preparation compared to native RNA sequencing, and due to amplification would allow sequencing experiments starting with minute RNA amounts as input (Oxford Nanopore Technologies 2018c, 2019). Additionally, we envision that future cDNA applications using Unique Molecular Identifiers (UMI) will make it possible to acquire multiple poly(A) tail measurements from each molecule, which would increase the fidelity of isoform-specific poly(A) tail measurements. Thus, using *tailfindr* with specific ONT cDNA applications offers new approaches to study the role of poly(A) tail lengths from scarce biological samples.

In conclusion, ONT RNA sequencing offers a new possibility to study poly(A) tail biology by directly associating poly(A) tail length with other RNA features in a transcript isoform-specific manner. *tailfindr* has proven successful in measuring the poly(A) tail of both RNA and DNA sequencing solely from base-called raw data, an approach that allows real-time analysis during ONT long-read sequencing. With the application of *tailfindr* for ONT DNA sequencing we allow future development of poly(A)-retaining cDNA sequencing assays that further increase the ability to study poly(A) tail lengths from limited material.

MATERIALS AND METHODS

Spike-in generation

To generate RNA with known poly(A) tail lengths, we used eGFP as a carrier RNA as it fulfills basic criteria for successful ONT RNA sequencing (especially minimal length requirement). The coding sequence of eGFP was amplified from pCS2+ –eGFP vector using High Fidelity Phusion MasterMix (ThermoFisher, #F-531L). The primers for the PCR included the SP6 promoter sequence and a barcode in the forward primer, as well as a homopolymer T stretch in the reverse primer (see Table 4). After gel purification of the desired PCR product, a second PCR was performed with a reverse primer that introduces a Bfo1 restriction site before the homopolymer T stretch (polyA Bfo1 rev, together with SP6 Bfo1 fw, Table 4). After gel purification and Phenol–chloroform extraction, the resulting PCR products were used for Nanopore DNA ligation sequencing (see below). For preparation of RNA spike-ins, the PCR products were digested with FastDigest Bfo1 (ThermoFisher, #FD2184) for 2 h and purified by Phenol–chloroform extraction. An amount of 100–300 ng of purified DNA was used for RNA in vitro transcription by the SP6 mMessage mMachine kit (ThermoFisher, #AM1340) following the manufacturer's procedures. The resulting RNA was purified using Zymo RNA Clean & Concentrator-5 columns (Zymo Research, #R1013).

ONT long-read sequencing

Native RNA sequencing was performed on two replicates using the ONT kit SQK-RNA001 following the manufacturer's protocol. One additional replicate was performed using the kit SQK-RNA002 omitting the reverse transcription reaction described in the manufacturer's protocol. In brief, 500 ng of poly(A)-selected RNA was mixed with 100 ng of poly(A) spike-in RNA, or 500 ng poly(A) spike-in RNA was used alone. The RNA was ligated to ONT RT adapter (RTA) and used for reverse transcription with SuperScript II (ThermoFisher, #18064022; omitted for third replicate). Next, the proprietary sequencing adapter was ligated using T4 DNA ligase (NEB, #M0202M) and loaded onto ONT Sequencing Flow Cells (FLO-MIN106 R9.4.1). Sequencing was performed for 16–24 h using MinKNOW 2 software. All RNA purification steps were performed with RNAClean XP beads (Beckham Coulter, #A63987) with 15 min incubation intervals.

DNA sequencing was performed using the DNA Ligation Kits SQK-LSK108 and SQK-LSK109 on poly(A)-containing PCR products. In brief, 500 ng of pooled barcoded PCR products were end-prepped using the NEBNext Ultra II dA tailing module (NEB, #E7546S) and ligated to proprietary sequencing adapters using T4 DNA ligase (NEB, #M0202M). Purified libraries were

sequenced on flow cells (FLO-MIN106 R9.4.1) for 24 h using MinKNOW 2.

Sequencing data processing

RNA and DNA raw reads were base-called using Albacore v2.3.3. DNA raw reads were additionally base-called with Guppy v2.3.1 using the flip-flop model. Sequencing quality and general metrics were assessed using NanoPlot (v1.19.0, De Coster et al. 2018). Reads that passed the default albacore quality filter were mapped against the eGFP sequence using minimap2 (v2.14-r883) with default settings for ONT data mapping (-ax splice -uf -14 for RNA; -ax splice for DNA, Li 2018).

Demultiplexing barcoded spike-ins

All alignments discussed in this manuscript, unless mentioned otherwise, were performed using Smith–Waterman local alignments with Biostrings (Pages et al. 2019) (match score 1; mismatch score -1; gap opening penalty 0; and gap extension penalty 1). The normalized alignment score was calculated by dividing the local alignment score by the length of the query sequence. If not otherwise mentioned, alignments with a normalized alignment score below 0.6 were discarded as unspecific.

Barcoded eGFP RNA reads with known poly(A) length were demultiplexed by locating the first 29 bases of eGFP sequence (see Table 5) within the first 250 bases of FASTA strings extracted from every FAST5 file. Next, the barcode was assigned by aligning the expected barcode sequences against the extracted read sequence preceding the eGFP alignment (see Table 5). The barcode with highest normalized alignment score (and above threshold of 0.6) was assigned to the read.

To analyze barcoded eGFP DNA reads, the orientation of reads was investigated by aligning the first 29 bases of eGFP and its reverse-complement (Table 5) to the first 250 bases of FASTA strings extracted from each FAST5 file. A read was considered a poly(A)-containing read if the normalized alignment score of eGFP sequence was greater than both the normalized alignment score of the reverse-complement of eGFP and the threshold value of 0.5. Reads where the normalized alignment score of the reverse-complement of eGFP was higher than the forward eGFP sequence and passed the threshold value of 0.5 were considered to be poly(T)-containing reads. For Barcode demultiplexing, first the sequence preceding the identified eGFP start was queried for the presence of the experiment-specific PCR front primer in the case of poly(A) reads, or its reverse-complement for poly(T) reads (sequences in Table 5). Next, the sequence between front primer and eGFP locations were used for barcode identification as described above.

Comparing poly(A) and poly(A)/(T) end coordinates to eGFP sequence alignments

All alignments were performed using Smith–Waterman local alignments with Biostrings (Pages et al. 2019) (match score 1; mismatch score −1; gap opening penalty 0; and gap extension penalty 1). The normalized alignment score was calculated by dividing the local alignment score by the length of the query sequence. If not otherwise mentioned, alignments with a normalized alignment score below 0.6 were discarded as unspecific.

RNA reads with known poly(A) length were screened for the presence of the eGFP end sequence by querying the reverse FASTA string against the reverse eGFP end sequence (see Table 5). Reversing the FASTA sequence is necessary to achieve similar orientation between the raw signal in events tables (3′ to 5′) and FASTA string (initially 5′ to 3′). If the first three bases of the alignment are a perfect match, the sample index corresponding to the first alignment base is extracted by matching the number of the alignment character in the reversed FASTA string with the cumulative move count in the corresponding FAST5 file. This

TABLE 5. Sequences used in *tailfindr* alignments

Name	Sequence
Barcode 10 nt	TGCAGATCTCTTGCC
Barcode 30 nt	TCGAAGCATTGTAA
Barcode 40 nt	AACGGTAGCCACCAA
Barcode 60 nt	TGCACGAGATTGATG
Barcode 100 nt	GACACATAGTCATGG
Barcode 150 nt	CATGAGTGCTGAGCT
eGFP start sequence	CCACCATGGTGAGCAAGGGCGAGGAGCTG
eGFP start sequence (reverse-complement)	CAGCTCCTCGCCCTTGCTCACCATGGTGG
reverse eGFP end sequence	GATGAACATGTCGAGCAGGTACGGCTCTCACTA
reverse eGFP end (reverse-complement)	TAGTGAGAGCCGTACCTGCTCGACATGTTCATC
PCR front primer	ATTTAGGTGACACTATAGCGCTCCATGCAAACCTGTC
PCR front primer (reverse-complement)	GACAGGTTTGCATGGAGCGCTATAGTGTCACCTAAAT
Nanopore front primer	TTTCTGTTGGTGCTGATATTGCTGCCATTACGGCCGGG
Nanopore end primer	GAGTCCGGGCGGCGC
Nanopore end primer (reverse-complement)	GCGCCGCCCGGACTC

sample index is further used to compare with the sample index defined by *tailfindr* as representing the poly(A) end.

DNA reads were split based on the read type [poly(A)- or poly(T)-containing, see above]. Poly(A)-containing reads were treated similar to RNA reads (see above). Poly(T)-containing reads were screened for the presence of the eGFP end sequence by querying the original FASTA string against the reverse-complement of the reverse eGFP end sequence (see Table 5). The corresponding sample index is extracted as described for RNA reads (see above).

tailfindr RNA poly(A) length estimation algorithm

To identify the signal corresponding to the poly(A) tails in RNA reads, the raw signal from ONT native RNA sequencing is extracted from the FAST5 files and *z*-normalized. Next, signal values above +3 and below −3 are truncated. The resulting processed raw signal is smoothened by a moving average filter (window size 400 samples; stride 1) to produce two smoothened signals: one by calculating the moving average from start to end, and one from end to start of the signal corresponding to the sequencing direction. Both smoothened signal vectors are then merged by point-by-point maximum calculation. Next, the calculated smoothened signal is segmented into regions being above or below 0.3. The expected signal of the ONT adapter consists of one segment above and one segment below 0.3 in the smoothened signal. The poly(A) tail immediately follows the Nanopore Adapter, thus the next segment in which the smoothened signal is above 0.3 is considered the poly(A) region, and the boundaries of this segment are considered the rough start and end of poly(A) tail (Fig. 1B). The threshold was chosen as the expected normalized signal for regions of homopolymer A on average is 0.89 and even a raw signal with two standard deviations below would result in a normalized signal of 0.55 (calculations in Supplemental Rmarkdown file).

The rough start and end are refined by first calculating a mean signal of the processed raw data contained between these boundaries through a moving average filter (window size 25; stride 25). Next, the slope of this mean signal is calculated between each two consecutive points. The boundaries of the longest continuous stretch of low-slope values (confined within bounds of +0.3 and −0.3 of slope signal) between the rough poly(A) start and end boundaries are considered the precise boundaries (Fig. 1B). The resulting poly(A) tail measurement in sample points is then normalized by the read-specific nucleotide translocation rate. To calculate the nucleotide translocation rate, the number of sample points per move is extracted from the FAST5 events table of each individual read (a "move" in raw data describes a single-nucleotide translocation through the pore as detected by base-calling). If a move of two is detected, two entries with each half the number of sample points are recorded; a move of two corresponds to a nucleotide translocation not identified by the base-caller. From the resulting vector of sample points per single move, the geometric mean is computed and used for normalization of poly(A) tail length.

tailfindr DNA poly(A)/(T) estimation algorithm

Unlike RNA, DNA is double-stranded. Thus, both poly(A) and poly(T) homopolymer stretches can occur. To determine the read orientation, the Nanopore-specific front and end primer sequences (sequences in Table 5) are aligned against the first 100 bases extracted from FAST5 files. A read is considered poly(T)-containing if the normalized alignment score of end primer sequence is greater than that of the front primer sequence, and above the threshold of 0.6. Conversely, a read is considered poly(A)-containing if the normalized alignment score of front primer sequence is greater than that of the end primer sequence, and above the threshold of 0.6. To ensure that the full poly(A) tail is present in raw data, the last 50 bases of poly(A)-containing reads are queried for the presence of the reverse-complement end primer sequence. Reads where the normalized alignment score of the reverse-complement end primer is below 0.6 are considered truncated poly(A) reads and not analyzed further.

To identify borders of poly(A) or poly(T) stretches by similar procedures, the raw data of poly(A)-containing reads is reversed. Thus, both the poly(A) and poly(T) stretches are expected to be at the beginning of the raw signal. The alignment of end primer is considered the approximate start of the poly(A) or poly(T) stretch. Next, the raw data is *z*-normalized and converted to absolute values. To reduce computational workload, calculations to identify precise borders of poly(A)/(T) stretches are restricted to 3000 raw samples downstream from the rough poly(A)/(T) start site. This 3000-samples wide search window is wide enough to accommodate poly(A)/(T) tails of ~350 nt length. The mean signal is generated by applying a sliding window (window size 10; stride 10) to the processed raw signal. Next, the slope of this mean signal is calculated between every two consecutive points. The precise start of the respective tail is considered to be the first location after the rough start site where the calculated slope is between −0.2 and 0.2, and the mean signal is between 0 and 0.3 for poly(T) reads and 0 to 0.6 for poly(A) reads. These thresholds contain all signal with two standard deviations away from the expected signal from homopolymer poly(T) or poly(A) stretches (calculation in Supplemental Rmarkdown file). To identify the precise tail end, the slope and the mean signals downstream from the precise tail start site are tested for violating their respective thresholds (see above). Since short non-tail-like signal spikes can randomly occur, we test the signal downstream from this tentative tail end for tail-like signal within thresholds until we either reach the end of the search window of 3000 sample points, or find another stretch of tail-like signal of at least 60 sample points in length. In the latter case, the tentative tail end is updated to the downstream tail end to account for the spike signal. The maximum allowable signal length exceeding the threshold that is located between two tail-like signals has been set to 120 nt (e.g., 120× read-specific nucleotide translocation rate).

The difference of the precise boundaries define the raw length of poly(A)/(T) stretches in sample points. This value is normalized by the read-specific nucleotide translocation rate calculated dependent on the respective base-calling strategy. For DNA reads base-called with standard models, the nucleotide translocation rate is defined as the geometric mean of the sample points per single move, as described for RNA poly(A) estimation. For flip-flop model base-calling, the raw signal is likely over-segmented resulting in too many nucleotide translocations (see Supplemental Discussion). To account for this, the average translocation rate is defined as the arithmetic mean of sample points per detected move after discarding the 5% highest outliers.

SUPPLEMENTAL MATERIAL

Supplemental material is available for this article.

DATA DEPOSITION

The code repository can be found at https://github.com/adnaniazi/tailfindr. The data repository can be found at https://www.ebi.ac.uk/ena/data/view/PRJEB31806.

ACKNOWLEDGMENTS

M.K. would like to acknowledge the constant scientific exchange and troubleshooting with, as well as critical assessment of the manuscript by, Kirill Yefimov and Teshome Bizuayehu. M.K. further wants to thank his family (small and big) for constant moral support during his career. A.N. wants to thank his wife and his current family expansion for making his life joyful every day. The project was supported by the Bergen Research Foundation (E.V.), the Sars International Centre for Marine Molecular Biology core funding (M.K.), University of Bergen core funding (A.M.N.; K.L.) and the Norwegian Research Council (#250049) (Y.T.-C.).

Received March 22, 2019; accepted June 25, 2019.

REFERENCES

Balagopal V, Bolisetty M, Al Husaini N, Coller J. 2017. Ccr4 and Pop2 control poly(A) tail length in *Saccharomyces cerevisiae*. bioRxiv doi:10.1101/140202

Bear DG, Fomproix N, Soop T, Björkroth B, Masich S, Daneholt B. 2003. Nuclear poly(A)-binding protein PABPN1 is associated with RNA polymerase II during transcription and accompanies the released transcript to the nuclear pore. *Exp Cell Res* **286:** 332–344. doi:10.1016/S0014-4827(03)00123-X

Beilharz TH, Preiss T. 2007. Widespread use of poly(A) tail length control to accentuate expression of the yeast transcriptome. *RNA* **13:** 982–997. doi:10.1261/rna.569407

Brawerman G. 1973. Alterations in the size of the poly(A) segment in newly-synthesized messenger RNA of mouse sarcoma 180 ascites cells. *Mol Biol Rep* **1:** 7–13. doi:10.1007/BF00357399

Bresson SM, Conrad NK. 2013. The human nuclear poly(A)-binding protein promotes RNA hyperadenylation and decay. *PLoS Genet* **9:** e1003893. doi:10.1371/journal.pgen.1003893

Chang H, Lim J, Ha M, Kim VN. 2014. TAIL-seq: genome-wide determination of poly(A) tail length and 3′ end modifications. *Mol Cell* **53:** 1044–1052. doi:10.1016/j.molcel.2014.02.007

Clegg KB, Pikó L. 1982. RNA synthesis and cytoplasmic polyadenylation in the one-cell mouse embryo. *Nature* **295:** 343–344. doi:10.1038/295342a0

Darnell JE, Philipson L, Wall R, Adesnik M. 1971. Polyadenylic acid sequences: role in conversion of nuclear RNA into messenger RNA. *Science* **174:** 507–510. doi:10.1126/science.174.4008.507

De Coster W, D'Hert S, Schultz DT, Cruts M, Van Broeckhoven C. 2018. NanoPack: visualizing and processing long-read sequencing data. *Bioinformatics* **34:** 2666–2669. doi:10.1093/bioinformatics/bty149

Diez J, Brawerman G. 1974. Elongation of the polyadenylate segment of messenger RNA in the cytoplasm of mammalian cells. *Proc Natl Acad Sci* **71:** 4091–4095. doi:10.1073/pnas.71.10.4091

Eckmann CR, Rammelt C, Wahle E. 2011. Control of poly(A) tail length. *Wiley Interdiscip Rev RNA* **2:** 348–361. doi:10.1002/wrna.56

Edmonds M, Vaughan MH, Nakazato H. 1971. Polyadenylic acid sequences in the heterogeneous nuclear RNA and rapidly-labeled polyribosomal RNA of HeLa cells: possible evidence for a precursor relationship. *Proc Natl Acad Sci* **68:** 1336–1340. doi:10.1073/pnas.68.6.1336

Ford LP, Bagga PS, Wilusz J. 1997. The poly(A) tail inhibits the assembly of a 3′-to-5′ exonuclease in an in vitro RNA stability system. *Mol Cell Biol* **17:** 398–406. doi:10.1128/mcb.17.1.398

Fuke H, Ohno M. 2008. Role of poly(A) tail as an identity element for mRNA nuclear export. *Nucleic Acids Res* **36:** 1037–1049. doi:10.1093/nar/gkm1120

Garalde DR, Snell EA, Jachimowicz D, Sipos B, Lloyd JH, Bruce M, Pantic N, Admassu T, James P, Warland A, et al. 2018. Highly parallel direct RNA sequencing on an array of nanopores. *Nat Methods* **15:** 201–206. doi:10.1038/nmeth.4577

Groner B, Hynes N, Phillips S. 1974. Length heterogeneity in the poly (adenylic acid) region of yeast messenger ribonucleic acid. *Biochemistry* **13:** 5378–5383. doi:10.1021/bi00723a020

Hake LE, Richter JD. 1994. CPEB is a specificity factor that mediates cytoplasmic polyadenylation during *Xenopus* oocyte maturation. *Cell* **79:** 617–627. doi:10.1016/0092-8674(94)90547-9

Hector RE, Nykamp KR, Dheur S, Anderson JT, Non PJ, Urbinati CR, Wilson SM, Minvielle-Sebastia L, Swanson MS. 2002. Dual requirement for yeast hnRNP Nab2p in mRNA poly(A) tail length control and nuclear export. *EMBO J* **21:** 1800–1810. doi:10.1093/emboj/21.7.1800

Hite JM, Eckert KA, Cheng KC. 1996. Factors affecting fidelity of DNA synthesis during PCR amplification of d(C-A)n•d(G-T)n microsatellite repeats. *Nucleic Acids Res* **24:** 2429–2434. doi:10.1093/nar/24.12.2429

Hommelsheim CM, Frantzeskakis L, Huang M, Ülker B. 2014. PCR amplification of repetitive DNA: a limitation to genome editing technologies and many other applications. *Sci Rep* **4:** 5052. doi:10.1038/srep05052

Jain M, Olsen HE, Paten B, Akeson M. 2016. The Oxford Nanopore MinION: delivery of nanopore sequencing to the genomics community. *Genome Biol* **17:** 239. doi:10.1186/s13059-016-1103-0

Jalkanen AL, Coleman SJ, Wilusz J. 2014. Determinants and implications of mRNA poly(A) tail size—Does this protein make my tail look big? *Semin Cell Dev Biol* **34:** 24–32. doi:10.1016/j.semcdb.2014.05.018

Legnini I, Alles J, Karaiskos N, Ayoub S, Rajewsky N. 2019. Full-length mRNA sequencing reveals principles of poly(A) tail length control. bioRxiv doi:10.1101/547034

Li H. 2018. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics* **34:** 3094–3100. doi:10.1093/bioinformatics/bty191

Lim J, Lee M, Son A, Chang H, Kim VN. 2016. mTAIL-seq reveals dynamic poly(A) tail regulation in oocyte-to-embryo development. *Genes Dev* **30:** 1671–1682. doi:10.1101/gad.284802.116

Lima SA, Chipman LB, Nicholson AL, Chen Y-H, Yee BA, Yeo GW, Coller J, Pasquinelli AE. 2017. Short poly(A) tails are a conserved feature of highly expressed genes. *Nat Struct Mol Biol* **24:** 1057–1063. doi:10.1038/nsmb.3499

Mendez R, Murthy KG, Ryan K, Manley JL, Richter JD. 2000. Phosphorylation of CPEB by Eg2 mediates the recruitment of CPSF into an active cytoplasmic polyadenylation complex. *Mol Cell* **6:** 1253–1259. doi:10.1016/S1097-2765(00)00121-0

Merkel CG, Wood TG, Lingrel JB. 1976. Shortening of the poly(A) region of mouse globin messenger RNA. *J Biol Chem* **251:** 5512–5515.

Millevoi S, Vagner S. 2010. Molecular mechanisms of eukaryotic pre-mRNA 3′ end processing regulation. *Nucleic Acids Res* **38:** 2757–2774. doi:10.1093/nar/gkp1176

Morrison MR, Merkel CG, Lingrel JB. 1973. Size of the poly(A) region in mouse globin messenger RNA. *Mol Biol Rep* **1:** 55–60. doi:10.1007/BF00357406

Murray EL, Schoenberg DR. 2008. Chapter 24 assays for determining poly(A) tail length and the polarity of mRNA decay in mammalian cells. *Methods Enzymol* **448:** 483–504. doi:10.1016/S0076-6879(08)02624-4

Nilsen TW. 2015. Measuring the length of poly(A) tails. *Cold Spring Harb Protoc* **2015:** 413–418. doi:10.1101/pdb.prot081034

Oxford Nanopore Technologies. 2018a. *Nanopore sequencing—the value of full-length transcripts without bias.* https://nanoporetech.com/resource-centre/rna-sequencing-white-paper-value-full-length-transcripts-without-bias

Oxford Nanopore Technologies. 2018b. *Clive G. Brown: Nanopore community meeting 2018 talk.* https://nanoporetech.com/about-us/news/clive-g-brown-nanopore-community-meeting-2018-talk

Oxford Nanopore Technologies. 2018c. *Low bias RNA-seq: PCR-cDNA, PCR-free direct cDNA and direct RNA sequencing.* https://nanoporetech.com/resource-centre/low-bias-rna-seq-pcr-cdna-pcr-free-direct-cdna-and-direct-rna-sequencing

Oxford Nanopore Technologies. 2019. *cDNA sequencing with Oxford Nanopore—getting started.* https://nanoporetech.com/sites/default/files/s3/literature/Nanopore-cDNA-Guide.pdf

Pagés H, Aboyoun P, Gentleman R, DebRoy S. 2019. *Biostrings: Efficient manipulation of biological strings.* R package version 2.52.0. https://bioconductor.org/packages/release/bioc/html/Biostrings.html. doi: 10.18129/B9.bioc.Biostrings.

Raabe T, Bollum FJ, Manley JL. 1991. Primary structure and expression of bovine poly(A) polymerase. *Nature* **353:** 229–234. doi:10.1038/353229a0

Raabe T, Murthy KG, Manley JL. 1994. Poly(A) polymerase contains multiple functional domains. *Mol Cell Biol* **14:** 2946–2957. doi:10.1128/MCB.14.5.2946

Rang FJ, Kloosterman WP, de Ridder J. 2018. From squiggle to basepair: computational approaches for improving nanopore sequencing read accuracy. *Genome Biol* **19:** 90. doi:10.1186/s13059-018-1462-9

Read RL, Martinho RG, Wang S-W, Carr AM, Norbury CJ. 2002. Cytoplasmic poly(A) polymerases mediate cellular responses to *S* phase arrest. *Proc Natl Acad Sci* **99:** 12079–12084. doi:10.1073/pnas.192467799

Subtelny AO, Eichhorn SW, Chen GR, Sive H, Bartel DP. 2014. Poly(A)-tail profiling reveals an embryonic switch in translational control. *Nature* **508:** 66–71. doi:10.1038/nature13007

Viehweger A, Krautwurst S, Lamkiewicz K, Madhugiri R, Ziebuhr J, Hölzer M, Marz M. 2018. Nanopore direct RNA sequencing reveals modification in full-length coronavirus genomes. bioRxiv doi:10.1101/483693

Woo YM, Kwak Y, Namkoong S, Kristjánsdóttir K, Lee SH, Lee JH, Kwak H. 2018. TED-seq identifies the dynamics of poly(A) length during ER stress. *Cell Rep* **24:** 3630–3641.e7. doi:10.1016/j.celrep.2018.08.084

Workman RE, Tang A, Tang PS, Jain M, Tyson JR, Zuzarte PC, Gilpatrick T, Razaghi R, Quick J, Sadowski N, et al. 2018. Nanopore native RNA sequencing of a human poly(A) transcriptome. bioRxiv doi:10.1101/459529

4.8 Poly(A)-tail profiling in PCR-amplified cDNA (in-house protocol)

In our *tailfindr* paper (above), we tested our algorithm on RNA and on unamplified DNA constructs carrying poly(A) tails. We showed that it is possible to estimate poly(A) tail length in not only RNA but also in unamplified DNA as well. We also showed that the poly(A) tail estimates in DNA have less variance compared to poly(A) tails measured using RNA sequencing. This is because the ONT's DNA sequencing chemistry is more advanced and more developed than RNA chemistry: The basecalling accuracy for DNA is higher than RNA, the DNA basecaller produces a basecall prediction for every 5 current samples compared to 10 for RNA basecaller, and the DNA feeds through the pore with fewer stalls compared to RNA due to lack of secondary structures and inter-molecule interactions on both *cis* and *trans* sides of the Nanopore membrane. All of these factors make the estimation of poly(A) tail boundaries and normalizer more accurate which in turn leads to DNA estimates of poly(A) tails being more accurate than the poly(A) estimates of RNA and having less variance.

However, unamplified cDNA provides limited advantages. In some scenarios for poly(A)-tail profiling, the starting RNA is too little. In these cases, the only way to get enough material for sequencing is to create cDNA through reverse transcription and amplify it. We developed an in-house protocol (Fig. 4.7) for making cDNA using custom splint adapters that would bind to the very end of the RNA poly(A) tail allowing us to reverse transcribe the RNA along with its complete poly(A) tail. This is followed by strand switching and PCR for amplification. We had to develop these custom splint adapters because the adapters in the then available PCR-cDNA kit, SQK-PCS109, were not designed to bind to the very end of the poly(A) tail but could bind anywhere in the poly(A) tail stretch which made it impossible to estimate the true length of poly(A)-tails in cDNA.

Using our custom adapters that would bind to the very end of the poly(A) tail, we did multiple sequencing runs on PCR-amplified cDNA created from RNA poly(A) standards (10, 30, 60, 100, 150 nt). We expected to see poly(A) tail peaks at 10, 30, 60, 100, 150 nt (see Fig 4.8a). But what actually observed is that the poly(A) tails in cDNA for 60, 100, and 150 nt standards were shortened to around 30 nt, and elongated to 25 nt and 35 nt for 10 nt and 30 nt poly(A) standards, respectively (see Fig 4.8b). This trend remained relatively unchanged even by changing the choice of polymerase used.

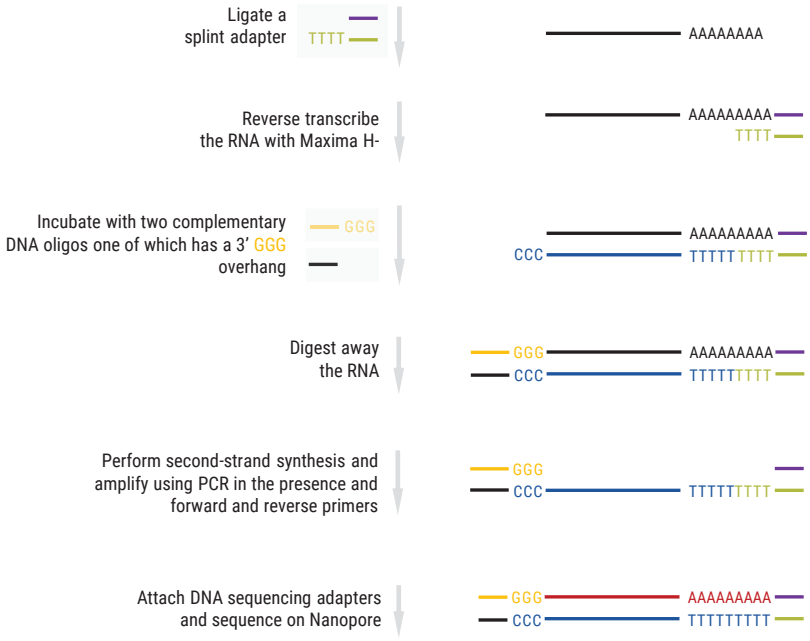


Fig. 4.7. Our approach for amplifying poly(A)+ RNA into cDNA

We argued that this change in length of the poly(A) tail standards during PCR is because of the priming of incomplete PCR products with poly(A) tails during the cycles of the PCR as shown in Fig. 4.9. Mispriming may cause the resulting poly(A) tails to become shorter or longer, depending on where the incomplete PCR product binds in the poly(A) tail.

To avoid the above-mentioned errors during PCR cycles and to preserve the poly(A) tail lengths in cDNA, we next tried using rolling-circle amplification of RNA into cDNA

4.9 Poly(A)-tail profiling of rolling circle-amplified cDNA (in-house protocol)

We argued that if we can somehow avoid using partially-extended primers from mispriming during PCR cycles, then we can faithfully copy the poly(A) tails from RNA into cDNA. We posited that rolling-circle amplification can be a possible way

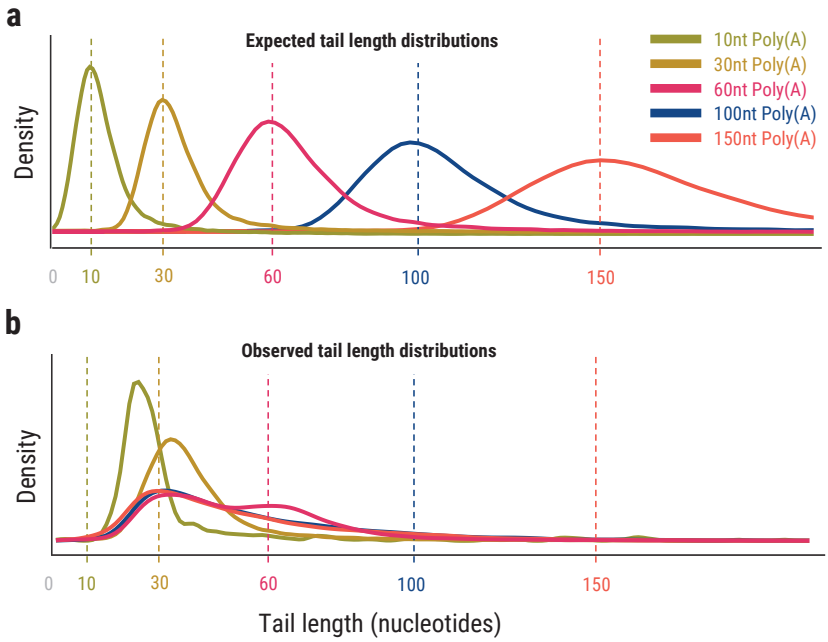


Fig. 4.8. **Poly(A)-tail profiling in cDNA poly(A) standards with poly(A) tail lengths 10, 30, 60, 100 and 150 nt. a)** We expected to see nice peaks for each of the five different standards we sequence. **b)** What we observe is the longer tails (60, 100, and 150nt) were shortened to around 30 nt, and shorter tails such as 10 nt and 30 nt poly(A) were elongated to 25 nt and 35 nt, respectively.

to successfully amplify the poly(A)-tails in cDNA. Rolling-circle amplification (RCA) [102], also known as multiple displacement amplification, uses a circular DNA template and random primers. A strand displacing polymerase extends these random primers by traversing the circular template and in doing so produces long single-stranded concatemers of the sequence in the original circular template. Using rolling circle amplification, we can obtain long reads that contain concatemers of poly(A) tails and poly(T) tails (see Fig. 4.10). The advantage of this approach is that we can obtain multiple measurements of the same transcript and its poly(A). Thus we can create a more accurate consensus transcript sequence and also a more robust poly(A) tail estimate by averaging the poly(A) tail length of individual repeats.

We modified *tailfindr* to work with concatemers obtained from RCA. Briefly, *tailfindr* identifies the individual repeats using the Repeated Match Alignment algorithm [103] to find the adapter sequences flanking the individual repeats. Once the repeats boundaries are found in the sequence space, these boundaries are then translated

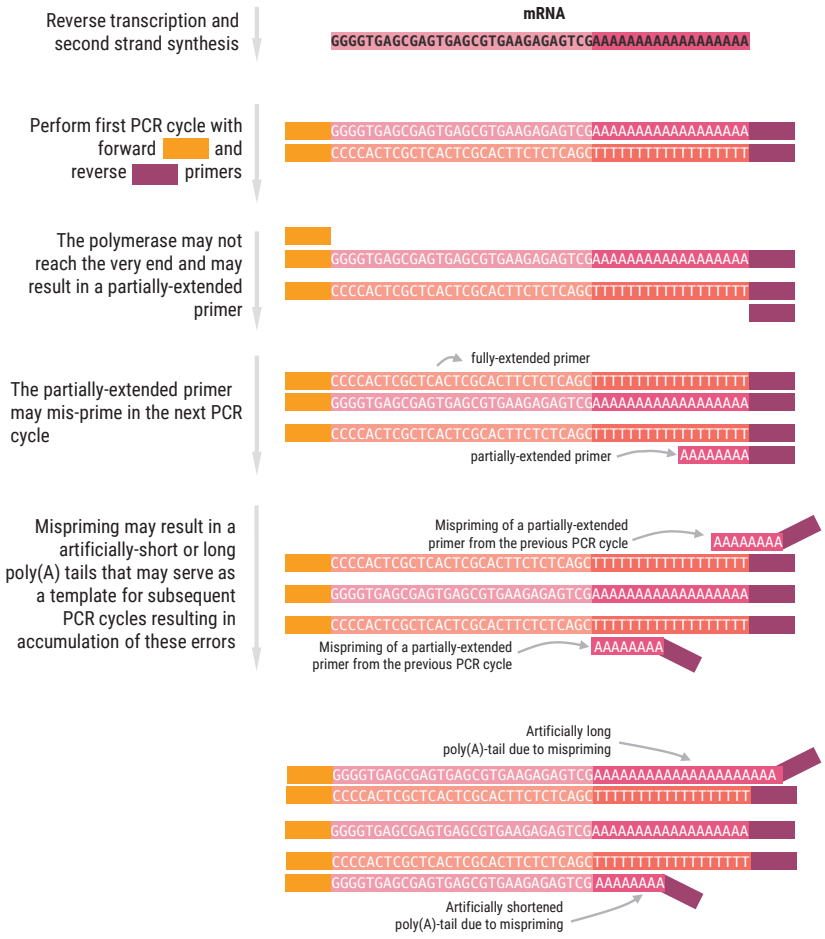


Fig. 4.9. Mispriming of partially-extended PCR primers can cause the poly(A) tails in amplified cDNA to be longer or smaller than the poly(A) tail in the original template. The poly(A) tail lengths in this amplified cDNA, therefore, do not faithfully represent the true poly(A) tail lengths in the starting RNA sample.

to signal space to delineate the signal boundaries for each repeat. The signal for individual repeat is then dispatched to the *tailfindr* algorithm [104] which finds the poly(A)/(T) stretches and estimates their length. A consensus transcript sequence is also created from individual repeats of the transcript in a concatemer using the DECIPHER R package [105], and the poly(A) tail lengths are averaged across the multiple transcript copies in a concatemer to yield an average poly(A) tail length (Fig. 4.11).

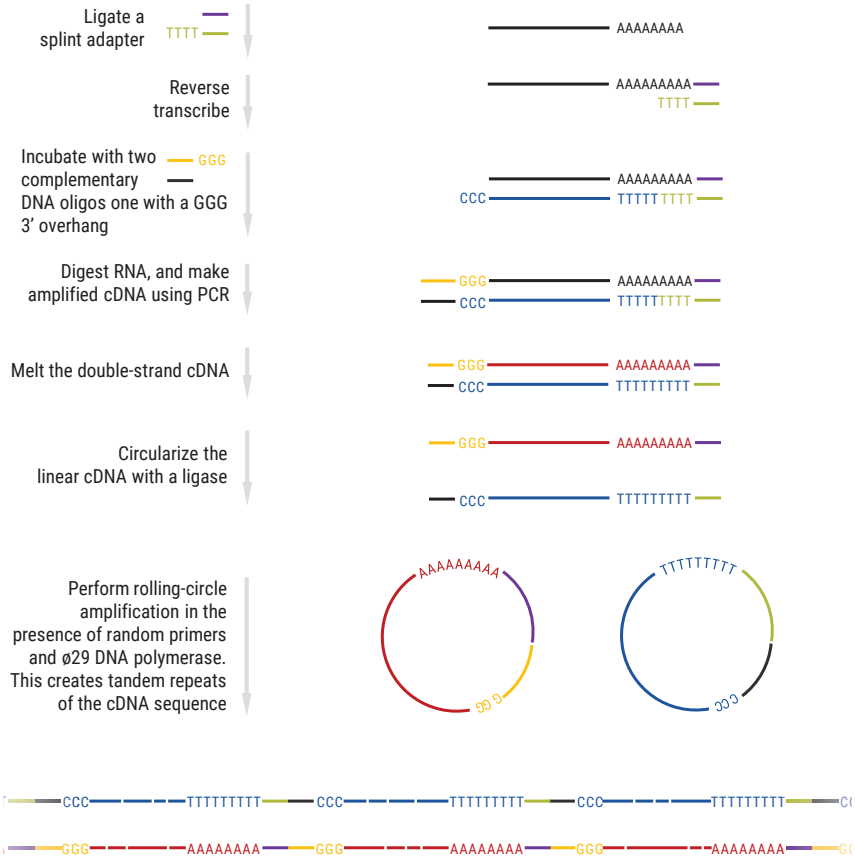


Fig. 4.10. **Protocol for producing amplified cDNA from RNA using rolling-circle amplification.** The resulting reads have concatemers of original linear cDNA sequences containing poly(A) and poly(T) stretches.

Using the RCA protocol, we sequenced two RNA standards with 30 and 100 nt long poly(A) tails on a 750 nt long transcript sequence. We expected to see peaks at 30 and 100 nt but observed only one peak around 18 nt (Fig. 4.12a). We also observed that the number of repeats of the transcript is low in the majority of the concatemers (Fig. 4.12b). Furthermore, the transcript sequences recovered from the repeats in concatemers are much shorter than the expected 750 nt (Fig. 4.12c).

We did many more attempts to optimize the protocol and obtain better poly(A) tail estimates but did not have much success.

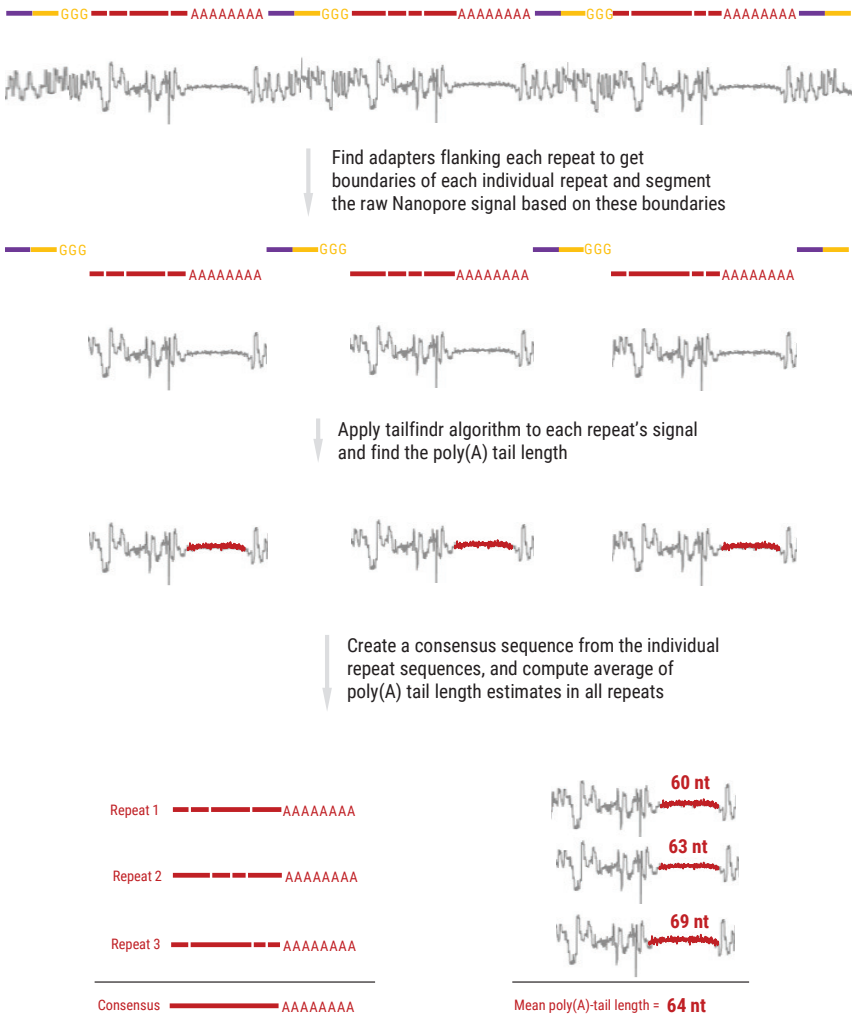


Fig. 4.11. *tailfindr's* approach to estimate poly(A) tail lengths in concatemers produced from rolling-circle amplification

4.10 Poly(A)-tail profiling cDNA (ONT protocol)

While we were busy with our attempts in debugging the RCA protocol, Oxford Nanopore Technologies contacted us about their new PCR-cDNA kit — SQK-PCS111 (`store.nanoporetech.com/cdna-pcr-sequencing-kit111.html`) — which could successfully amplify poly(A) tail in cDNA. ONT wanted us to make appropriate

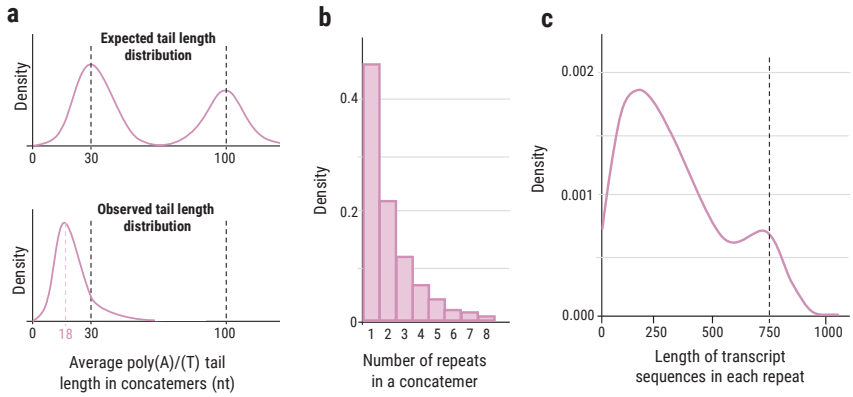


Fig. 4.12. **Results of poly(A)-tail profiling on rolling-circle amplified cDNA. a)** We expected to see two peaks around 30 and 100 nt (top panel) corresponding to two different poly(A) standards used in the experiments, but we observed a single peak around 18 nt (bottom panel). **b)** Majority of the concatemers had only one repeat in them. **c)** We used transcript sequences of length 750 nt in the experiment, but the majority of the transcripts recovered from the individual repeats were not much shorter than 750 nt.

changes in *tailfindr* to work with this kit, which we did, and now *tailfindr* works out-of-the-box for this kit.

What this kit does differently compared to our approach is digesting away the part of the splint adapter that has the poly(T) stretch before reverse transcription of the RNA (Fig. 4.13). It is accomplished by using a mixture of Exonuclease I and USER. Exonuclease I degrades a DNA strand from 5'–to–3' when it is double-stranded. It can also degrade 3'–to–5' but in a much slower way. The U in the splint adapter ensures that the degradation of the top adapter ligated to the RNA poly(A) tail stops at the abasic site left because of the excision of U by the USER enzyme. Next, a shorter primer, which is complementary to the DNA adapter left in the RNA strand, starts the reverse transcription reaction. We think that these additional steps help in faithfully copying RNA polyA tails in cDNA.

Using RNA standards of poly(A) tail length 10, 30, 50, 70, and 100 nt and amplifying them with PCS111 kit into amplified cDNA, we were able to successfully validate that the *tailfindr* can estimate correct tail lengths in both RNA and cDNA (Fig. 4.14 a and b).

If needed, the PCR step in this PCS111 can be omitted during library prep and the sequencing adapters can be ligated immediately after the second-strand synthesis. This direct cDNA sequencing (i.e. without PCR amplification) and *tailfindr* can also be used on such data for poly(A)-tail profiling.

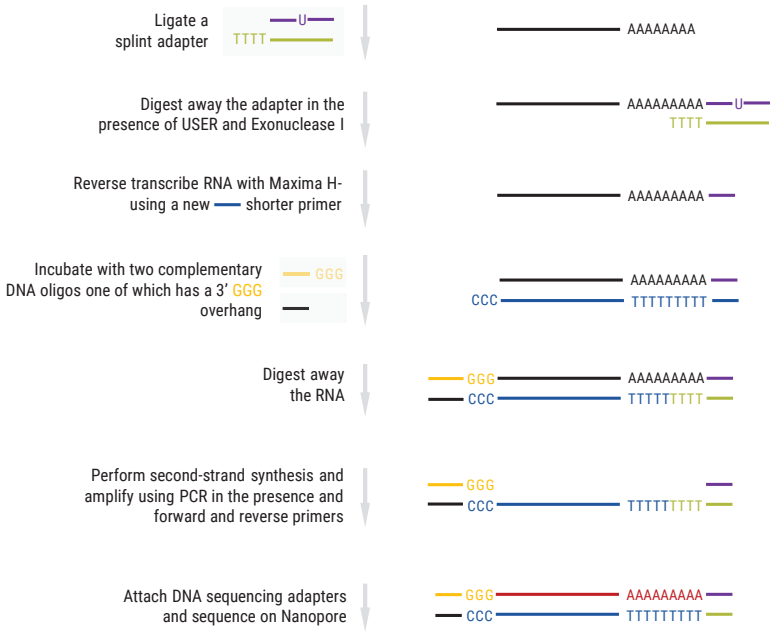


Fig. 4.13. **PCR-cDNA sequencing protocol of SQK-PCS111 kit.** The kit digests away part of the splint adapter before reverse transcription which helps prevent mispriming during PCR cycles and consequently leads to faithful reproduction of poly(A) tails in amplified cDNA.

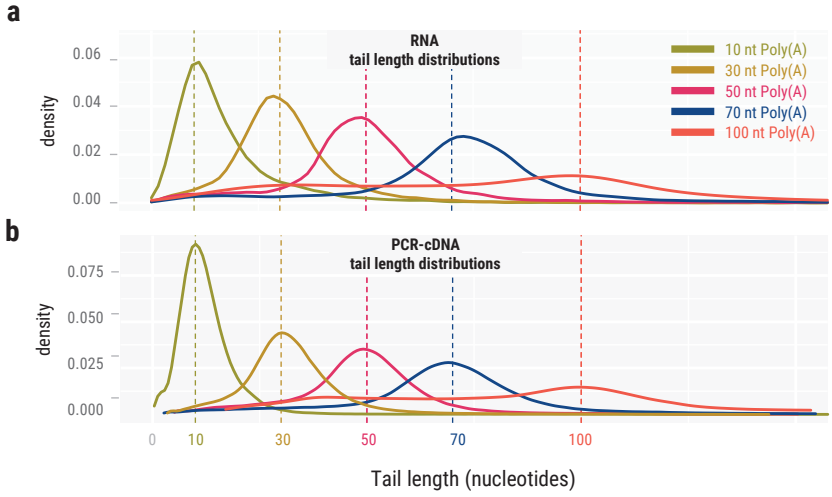


Fig. 4.14. **Poly(A)-tail profiling results. a)** on RNA tail standards **b)** on cDNA tail standards obtained by PCR amplification of RNA standards in (a) into cDNA with the newly-released PCR-cDNA kit PCS111.

With this new kit out and working as expected, we abandoned our RCA approach for amplifying RNA into cDNA.

4.11 Discussion and future perspectives

With *tailfindr*, we have developed a method that can perform poly(A)-tail profiling in native RNA, direct cDNA (unamplified), and PCR-cDNA transcriptome-wide at a single-molecule resolution. *tailfindr* also provides the start and end coordinates of a poly(A) tail which then enable researchers to also home in to the polyadenylation sites and investigate alternative polyadenylation.

The default ONT protocols for native RNA, direct cDNA (unamplified), and PCR-cDNA sequencing enrich poly(A)+ RNA and can only study this small sub-population of RNA. However, total RNA is composed of poly(A)- transcript species that could be of potential interest to many who may want to study polyA+ and poly(A)- transcripts simultaneously while needing to have poly(A) tail estimates for poly(A)+ transcripts. To cater to these needs, a new protocol — Nano3P-seq [106] — was developed and we worked together with the developers of this method to adapt *tailfindr* such that it now works with both poly(A)+ and poly(A)- transcripts sequenced with this protocol. This protocol allows researchers to do poly(A)-tail profiling on unamplified cDNA made from poly(A)+/- RNA. Instead of using a splint-adapter with poly(T) overhang to the RNA, Nano3P-seq uses a splint-adaptor with a single N overhang. In doing so Nano3P-seq can sequence all the different RNA species including those with non-homogenous poly(A) tails such as transcripts with poly(A) tails that are poly-uridinated at their 3'-ends.

In eukaryotes, poly(A) tails can also be trailed by a stretch of uridine bases [107]. Furthermore, non-polyadenylated transcripts can be tailed with poly(U) stretches that are thought to be essential for their degradation [108]. Currently, *tailfindr* only works on poly(A) stretches in RNA. But the algorithm could be modified and extended to estimate the length of poly(U) tails both when it appears alone or in combination with a poly(A) tail. We are currently collaborating with University of Syracuse in developing *tailfindr* for such use cases.

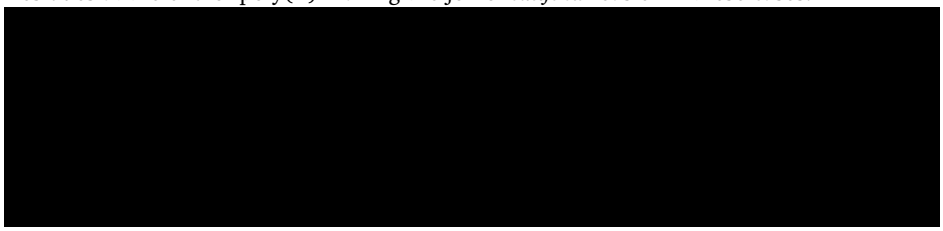
Conclusion and future outlook

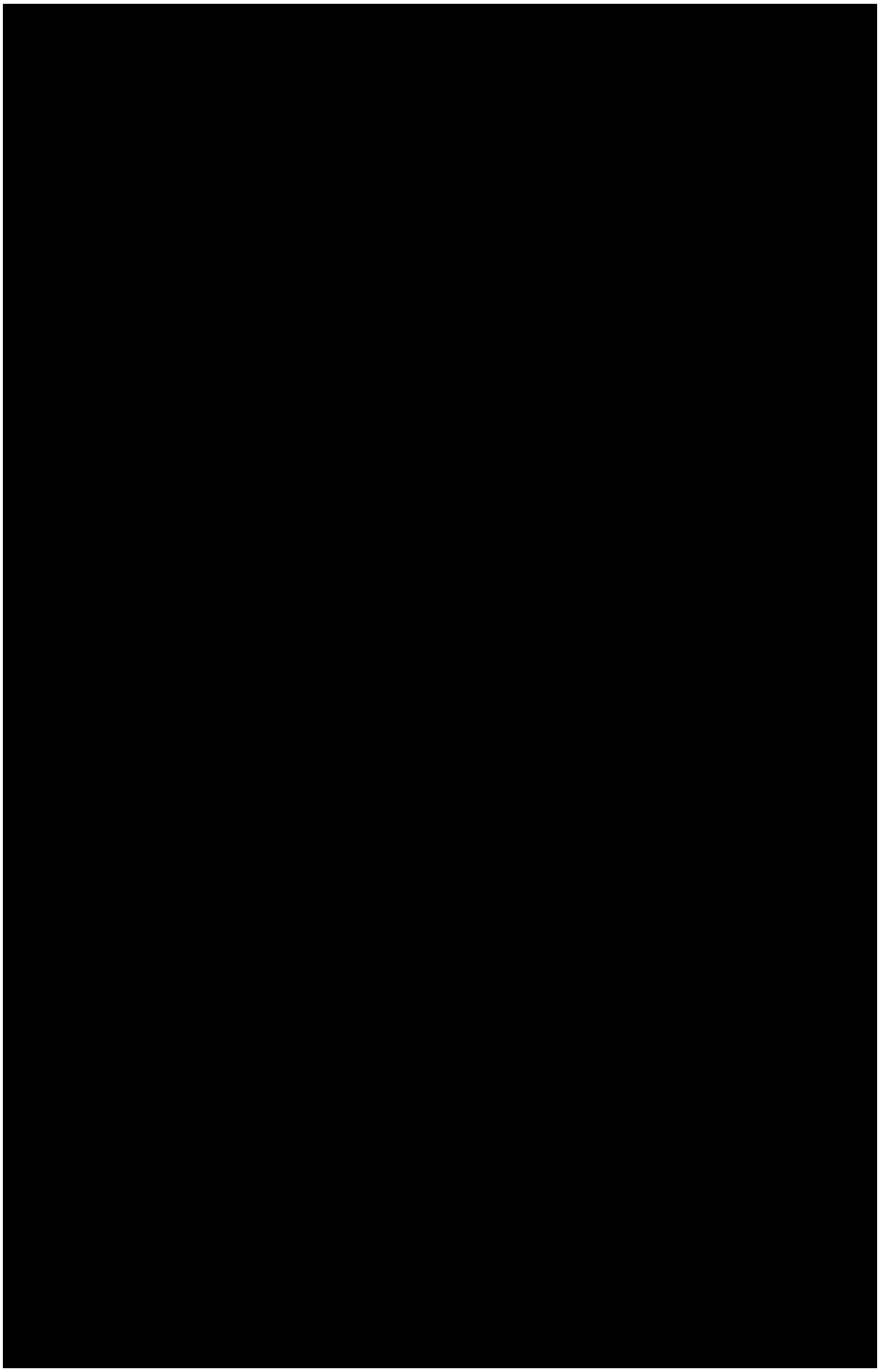
5

By developing the *capable* and *tailfindr* frameworks, I am providing the scientific community with tools to predict cap modifications and poly(A) tail at a single-molecule resolution. Both these tools can be applied on the same reads yielding for each read a prediction of cap type at its 5'-end and an estimate of the poly(A) tail length at its 3'-end. Currently, there exists no method that can simultaneously probe the cap type and tail length of an RNA in a single assay. This approach has the potential to shed new light on the role that RNA caps and poly(A) tails play in the lifecycle of an RNA molecule.

Since its publication in 2020, the *tailfindr* tool has been cited 30 times and has been directly used in research involving pathogen surveillance and the study of gene expression and tail dynamics in humans, zebrafish, mouse, and SARS CoV-2 [109, 106, 110, 111]. *tailfindr* has also been integrated into the Master of Pores NextFlow pipeline [112] allowing its deployment on clusters and other runtime environments. Based on our work on *tailfindr*, we have already established two fruitful ongoing collaborations with research groups in Spain and US for extending *tailfindr* with new functionality for studying poly(A)-tails in total RNA libraries, and for investigating tails with non-adenosine residues.

Currently, *tailfindr* only predicts the poly(A) tail length in individual Nanopore reads, and the downstream analysis of these tail predictions is then left for the end-users to work out. In the future, we hope to include in *tailfindr* the functionality for studying alternative polyadenylation sites, isoform- and gene-specific poly(A) tail lengths, differential poly(A)-tailing and alternative polyadenylation site usage. Furthermore, with ONT's recent announcement (Clive Brown's technology update; March 30, 2022) of discontinuation of the R9 pore by the end of 2023, we will also have to adapt *tailfindr* for the newer dual-head R10 pore. Keeping up with the chemistry and pore updates will be crucial for the long-term survival of this tool. We hope that the R10 pore will be able to more accurately basecall shorter tails and non-adenosine residues at the end of poly(A) making the job of *tailfindr* easier in these cases.





Bibliography

[1] Remigiusz Worch, Anna Niedzwiecka, Janusz Stepinski, et al. “Specificity of recognition of mRNA 5’ cap by human nuclear cap-binding complex”. en. In: *RNA* 11.9 (Sept. 2005), pp. 1355–1363 (cit. on p. 1).

[2] Anna Niedzwiecka, Joseph Marcotrigiano, Janusz Stepinski, et al. “Biophysical Studies of eIF4E Cap-binding Protein: Recognition of mRNA 5’ Cap Structure and Synthetic Fragments of eIF4G and 4E-BP1 Proteins”. In: *J. Mol. Biol.* 319.3 (June 2002), pp. 615–635 (cit. on p. 1).

[3] Christine Schuberth-Wagner, Janos Ludwig, Ann Kristin Bruder, et al. “A Conserved Histidine in the RNA Sensor RIG-I Controls Immune Tolerance to N1-2’O-Methylated Self RNA”. en. In: *Immunity* 43.1 (July 2015), pp. 41–51 (cit. on p. 1).

[4] Maria Werner, Elzbieta Purta, Katarzyna H Kaminska, et al. “2’-O-ribose methylation of cap2 in human: function and evolution in a horizontally mobile family”. en. In: *Nucleic Acids Res.* 39.11 (June 2011), pp. 4756–4768 (cit. on pp. 1, 23, 26).

[5] Y Furuichi, M Morgan, A J Shatkin, et al. “Methylated, blocked 5 termini in HeLa cell mRNA”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 72.5 (May 1975), pp. 1904–1908 (cit. on p. 1).

[6] Karolina Drazkowska, Natalia Baran, Marcin Warminski, et al. “2’-O-methylation of the second transcribed nucleotide within mRNA 5’ cap impacts protein production level in a cell specific manner and contributes to RNA immune evasion”. In: *bioRxiv* (Feb. 2022) (cit. on pp. 1, 26).

[7] Ewa Grudzien-Nogalska, Jeremy G Bird, Bryce E Nickels, and Megerditch Kiledjian. “‘NAD-capQ’ detection and quantitation of NAD caps”. en. In: *RNA* 24.10 (Oct. 2018), pp. 1418–1425 (cit. on p. 1).

[8] Victoria H Cowling. “CAPAM: The mRNA Cap Adenosine N6-Methyltransferase”. en. In: *Trends Biochem. Sci.* 44.3 (Mar. 2019), pp. 183–185 (cit. on pp. 1, 23).

[9] Andreas J Gruber and Mihaela Zavolan. “Alternative cleavage and polyadenylation in health and disease”. en. In: *Nat. Rev. Genet.* 20.10 (Oct. 2019), pp. 599–614 (cit. on p. 2).

[10] Agnieszka Tudek, Marta Lloret-Llinares, and Torben Heick Jensen. “The multitasking polyA tail: nuclear RNA maturation, degradation and export”. en. In: *Philos. Trans. R. Soc. Lond. B Biol. Sci.* 373.1762 (Nov. 2018) (cit. on p. 2).

[11] Harvey N Rubin and Mostafa N Halim. “Why, when and how does the poly(A) tail shorten during mRNA translation?” In: *International Journal of Biochemistry* 25.3 (1993), pp. 287–295 (cit. on p. 2).

[12]S Z Tarun Jr and A B Sachs. “Association of the yeast poly(A) tail binding protein with translation initiation factor eIF-4G”. en. In: *EMBO J.* 15.24 (Dec. 1996), pp. 7168–7177 (cit. on p. 2).

[13]S E Wells, P E Hillner, R D Vale, and A B Sachs. “Circularization of mRNA by eukaryotic translation initiation factors”. en. In: *Mol. Cell* 2.1 (July 1998), pp. 135–140 (cit. on p. 2).

[14]Ki Young Paek, Ka Young Hong, Incheol Ryu, et al. “Translation initiation mediated by RNA looping”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 112.4 (Jan. 2015), pp. 1041–1046 (cit. on p. 2).

[15]Erika Check Hayden. “Data from pocket-sized genome sequencer unveiled”. en. In: *Nature* (Feb. 2014) (cit. on pp. 4, 7, 10).

[16]Daniel R Garalde, Elizabeth A Snell, Daniel Jachimowicz, et al. “Highly parallel direct RNA sequencing on an array of nanopores”. en. In: *Nat. Methods* 15.3 (Mar. 2018), pp. 201–206 (cit. on pp. 4, 7, 10).

[17]Henry Brinkerhoff, Albert S W Kang, Jingqian Liu, Aleksei Aksimentiev, and Cees Dekker. “Multiple rereads of single proteins at single-amino acid resolution using nanopores”. en. In: *Science* 374.6574 (Dec. 2021), pp. 1509–1513 (cit. on pp. 4, 7, 10).

[18]Nicole Stéphanie Galenkamp, Misha Soskine, Jos Hermans, Carsten Wloka, and Giovanni Maglia. “Direct electrical quantification of glucose and asparagine from bodily fluids using nanopores”. en. In: *Nat. Commun.* 9.1 (Oct. 2018), p. 4085 (cit. on p. 4).

[19]Sarah Zernia, Nieck Jordy van der Heide, Nicole Stéphanie Galenkamp, Giorgos Gouridis, and Giovanni Maglia. “Current Blockades of Proteins inside Nanopores for Real-Time Metabolome Analysis”. en. In: *ACS Nano* 14.2 (Feb. 2020), pp. 2296–2307 (cit. on p. 4).

[20]Pehr Edman, Erik Högfeldt, Lars Gunnar Sillén, and Per-Olof Kinell. “Method for determination of the amino acid sequence in peptides”. In: *Acta Chem. Scand.* 4 (1950), pp. 283–293 (cit. on p. 7).

[21]Dan Dominissini, Sharon Moshitch-Moshkovitz, Schraga Schwartz, et al. “Topology of the human and mouse m6A RNA methylomes revealed by m6A-seq”. In: *Nature* 485.7397 (2012), pp. 201–206 (cit. on p. 8).

[22]Qing Dai, Sharon Moshitch-Moshkovitz, Dali Han, et al. “Nm-seq maps 2’-O-methylation sites in human mRNA with base precision”. en. In: *Nat. Methods* 14.7 (July 2017), pp. 695–698 (cit. on p. 8).

[23]Pietro Boccaletto, Magdalena A Machnicka, Elzbieta Purta, et al. “MODOMICS: a database of RNA modification pathways. 2017 update”. en. In: *Nucleic Acids Res.* 46.D1 (Jan. 2018), pp. D303–D307 (cit. on p. 9).

[24]David W Deamer and Daniel Branton. “Characterization of nucleic acids by nanopore analysis”. en. In: *Acc. Chem. Res.* 35.10 (Oct. 2002), pp. 817–825 (cit. on pp. 9, 11).

[25] Nava Whiteford, Tom Skelly, Christina Curtis, et al. “Swift: primary data analysis for the Illumina Solexa sequencing platform”. en. In: *Bioinformatics* 25.17 (Sept. 2009), pp. 2194–2199 (cit. on p. 10).

[26] Matthew D Macmanes. “On the optimal trimming of high-throughput mRNA sequence data”. en. In: *Front. Genet.* 5 (Jan. 2014), p. 13 (cit. on p. 10).

[27] Tom Z Butler, Mikhail Pavlenok, Ian M Derrington, Michael Niederweis, and Jens H Gundlach. “Single-molecule DNA detection with an engineered MspA protein nanopore”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 105.52 (Dec. 2008), pp. 20647–20652 (cit. on p. 10).

[28] A C Ward and W Kim. “MinION™: New, Long Read, Portable Nucleic Acid Sequencing Device”. In: *J. Bacteriol. Virol.* (2015) (cit. on p. 12).

[29] James A Cracknell, Deanpen Japrung, and Hagan Bayley. “Translocating kilobase RNA through the Staphylococcal α-hemolysin nanopore”. en. In: *Nano Lett.* 13.6 (June 2013), pp. 2500–2505 (cit. on p. 12).

[30] Heng Li. “Minimap2: pairwise alignment for nucleotide sequences”. en. In: *Bioinformatics* 34.18 (Sept. 2018), pp. 3094–3100 (cit. on p. 20).

[31] A J Shatkin and J L Manley. “The ends of the affair: capping and polyadenylation”. en. In: *Nat. Struct. Biol.* 7.10 (Oct. 2000), pp. 838–842 (cit. on p. 21).

[32] Stewart Shuman. “RNA capping: progress and prospects”. en. In: *RNA* 21.4 (Apr. 2015), pp. 735–737 (cit. on p. 21).

[33] Anand Ramanathan, G Brett Robb, and Siu-Hong Chan. “mRNA capping: biological functions and applications”. en. In: *Nucleic Acids Res.* 44.16 (Sept. 2016), pp. 7511–7526 (cit. on p. 21).

[34] Yasuhiro Furuichi. “Discovery of m7G-cap in eukaryotic mRNAs”. In: *Proc. Jpn. Acad. Ser. B Phys. Biol. Sci.* 91.8 (2015), pp. 394–409 (cit. on pp. 23, 29).

[35] Laurence Wurth, Anne-Sophie Gribling-Burrer, Céline Verheggen, et al. “Hypermethylated-capped selenoprotein mRNAs in mammals”. en. In: *Nucleic Acids Res.* 42.13 (July 2014), pp. 8663–8677 (cit. on p. 24).

[36] J Marcotrigiano, A C Gingras, N Sonenberg, and S K Burley. “X-ray studies of the messenger RNA 5’ cap-binding protein (eIF4E) bound to 7-methyl-GDP”. en. In: *Nucleic Acids Symp. Ser.* 36 (1997), pp. 8–11 (cit. on p. 24).

[37] Hao Hu, Nora Flynn, and Xuemei Chen. “Discovery, Processing, and Potential Role of Noncanonical Caps in RNA”. In: *Epitranscriptomics*. Ed. by Stefan Jurga and Jan Barciszewski. Cham: Springer International Publishing, 2021, pp. 435–469 (cit. on p. 25).

[38] Oldřich Hudeček, Roberto Benoni, Paul E Reyes-Gutierrez, et al. “Dinucleoside polyphosphates act as 5’-RNA caps in bacteria”. en. In: *Nat. Commun.* 11.1 (Feb. 2020), p. 1052 (cit. on p. 25).

[39] J D Lewis and E Izaurralde. “The role of the cap structure in RNA processing and nuclear export”. en. In: *Eur. J. Biochem.* 247.2 (July 1997), pp. 461–469 (cit. on p. 26).

[40]T Preiss and M W Hentze. “From factors to mechanisms: translation and translational control in eukaryotes”. en. In: *Curr. Opin. Genet. Dev.* 9.5 (Oct. 1999), pp. 515–521 (cit. on p. 26).

[41]Roy Parker and Ujwal Sheth. “P bodies and the control of mRNA translation and degradation”. en. In: *Mol. Cell* 25.5 (Mar. 2007), pp. 635–646 (cit. on p. 26).

[42]Swapnil C Devarkar, Chen Wang, Matthew T Miller, et al. “Structural basis for m7G recognition and 2’-O-methyl discrimination in capped RNAs by the innate immune receptor RIG-I”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 113.3 (Jan. 2016), pp. 596–601 (cit. on p. 26).

[43]K M Reinisch, M L Nibert, and S C Harrison. “Structure of the reovirus core at 3.6 A resolution”. en. In: *Nature* 404.6781 (Apr. 2000), pp. 960–967 (cit. on p. 26).

[44]A J Caton and J S Robertson. “Structure of the host-derived sequences present at the 5’ ends of influenza virus mRNA”. en. In: *Nucleic Acids Res.* 8.12 (June 1980), pp. 2591–2603 (cit. on p. 26).

[45]Josh M Whisenand, Krist T Azizian, Jordana M Henderson, et al. “Considerations for the Design and cGMP Manufacturing of mRNA Therapeutics”. In: *San Diego, CA: TriLink BioTechnologies* (2017) (cit. on p. 26).

[46]Radha Raman Pandey, Elena Delfino, David Homolka, et al. “The Mammalian Cap-Specific m6Am RNA Methyltransferase PCIF1 Regulates Transcript Levels in Mouse Tissues”. en. In: *Cell Rep.* 32.7 (Aug. 2020), p. 108038 (cit. on p. 26).

[47]Jan Mauer, Xiaobing Luo, Alexandre Blanjoie, et al. “Reversible methylation of m6Am in the 5’ cap controls mRNA stability”. en. In: *Nature* 541.7637 (Jan. 2017), pp. 371–375 (cit. on p. 27).

[48]J Huber, U Cronshagen, M Kadokura, et al. “Snurportin1, an m3G-cap-specific nuclear import receptor with a novel domain structure”. en. In: *EMBO J.* 17.14 (July 1998), pp. 4114–4126 (cit. on p. 27).

[49]Dong Jia, Lun Cai, Housheng He, et al. “Systematic identification of non-coding RNA 2,2,7-trimethylguanosine cap structures in Caenorhabditis elegans”. en. In: *BMC Mol. Biol.* 8 (Sept. 2007), p. 86 (cit. on p. 27).

[50]Hana Cahová, Marie-Luise Winz, Katharina Höfer, Gabriele Nübel, and Andres Jäschke. “NAD captureSeq indicates NAD as a bacterial cap for a subset of regulatory RNAs”. en. In: *Nature* 519.7543 (Mar. 2015), pp. 374–377 (cit. on p. 27).

[51]Robert W Walters, Tyler Matheny, Laura S Mizoue, et al. “Identification of NAD+ capped mRNAs in Saccharomyces cerevisiae”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 114.3 (Jan. 2017), pp. 480–485 (cit. on p. 27).

[52]Xinfu Jiao, Selom K Doamekpor, Jeremy G Bird, et al. “5’ End Nicotinamide Adenine Dinucleotide Cap in Human Cells Promotes RNA Decay through DXO-Mediated deNADding”. en. In: *Cell* 168.6 (Mar. 2017), 1015–1027.e10 (cit. on p. 27).

[53]Yuan Wang, Shaofang Li, Yonghui Zhao, et al. “NAD+-capped RNAs are widespread in the Arabidopsis transcriptome and can probably be translated”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 116.24 (June 2019), pp. 12094–12102 (cit. on p. 27).

[54] Jeremy G Bird, Urmimala Basu, David Kuster, et al. “Highly efficient 5’ capping of mitochondrial RNA with NAD+ and NADH by yeast and human mitochondrial RNA polymerase”. en. In: *Elife* 7 (Dec. 2018), e42179 (cit. on p. 27).

[55] Yaqing Zhang, David Kuster, Tobias Schmidt, et al. “Extensive 5’-surveillance guards against non-canonical NAD-caps of nuclear mRNAs in yeast”. en. In: *Nat. Commun.* 11.1 (Nov. 2020), p. 5508 (cit. on p. 27).

[56] Selom K Doamekpor, Ewa Grudzien-Nogalska, Agnieszka Mlynarska-Cieslak, et al. “DXO/Rai1 enzymes remove 5’-end FAD and dephospho-CoA caps on RNAs”. en. In: *Nucleic Acids Res.* 48.11 (June 2020), pp. 6136–6148 (cit. on p. 28).

[57] Sunny Sharma, Ewa Grudzien-Nogalska, Keith Hamilton, et al. “Mammalian Nudix proteins cleave nucleotide metabolite caps on RNAs”. en. In: *Nucleic Acids Res.* 48.12 (July 2020), pp. 6788–6798 (cit. on p. 28).

[58] Jin Wang, Bing Liang Alvin Chew, Yong Lai, et al. “Quantifying the RNA cap epitranscriptome reveals novel caps in cellular and viral RNA”. en. In: *Nucleic Acids Res.* 47.20 (Sept. 2019), e130–e130 (cit. on pp. 28, 31).

[59] S R Langberg and B Moss. “Post-transcriptional modifications of mRNA. Purification and characterization of cap I and cap II RNA (nucleoside-2’-)-methyltransferases from HeLa cells”. en. In: *J. Biol. Chem.* 256.19 (Oct. 1981), pp. 10054–10060 (cit. on p. 28).

[60] J E Cleaver and H J Burki. “Biological Damage from Intranuclear Carbon-14 Decays: DNA Single-strand Breaks and Repair in Mammalian Cells”. In: *Int. J. Radiat. Biol. Relat. Stud. Phys. Chem. Med.* 26.4 (Jan. 1974), pp. 399–403 (cit. on p. 29).

[61] Alison Galloway, Abdelmadjid Atrih, Renata Grzela, et al. “CAP-MAP: cap analysis protocol with minimal analyte processing, a rapid and sensitive approach to analysing mRNA cap structures”. en. In: *Open Biol.* 10.2 (Feb. 2020), p. 190306 (cit. on p. 30).

[62] Nils Muthmann, Petr Špaček, Dennis Reichert, Melissa van Dülmen, and Andrea Rentmeister. “Quantification of mRNA cap-modifications by means of LC-QqQ-MS”. en. In: *Methods* (May 2021) (cit. on p. 31).

[63] Irina O Vvedenskaya and Bryce E Nickels. “CapZyme-Seq: A 5’-RNA-Seq Method for Differential Detection and Quantitation of NAD-Capped and Uncapped 5’-Triphosphate RNA”. en. In: *STAR Protoc* 1.1 (June 2020) (cit. on p. 31).

[64] Hailei Zhang, Huan Zhong, Xufeng Wang, et al. “Use of NAD tagSeq II to identify growth phase-dependent alterations in E. coli RNA NAD+ capping”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 118.14 (Apr. 2021) (cit. on p. 33).

[65] David R Paquette, Jeffrey S Mugridge, David E Weinberg, and John D Gross. “Application of a Schizosaccharomyces pombe Edc1-fused Dcp1-Dcp2 decapping enzyme for transcription start site mapping”. en. In: *RNA* 24.2 (Feb. 2018), pp. 251–257 (cit. on p. 37).

[66] Marina Lizio, Jayson Harshbarger, Hisashi Shimoji, et al. “Gateways to the FANTOM5 promoter level mammalian expression atlas”. en. In: *Genome Biol.* 16 (Jan. 2015), p. 22 (cit. on p. 40).

[67]Olivia S Rissland, Andrea Mikulasova, and Chris J Norbury. “Efficient RNA polyuridylation by noncanonical poly(A) polymerases”. en. In: *Mol. Cell. Biol.* 27.10 (May 2007), pp. 3612–3624 (cit. on p. 41).

[68]Hiroki Ueda. “nanoDoc: RNA modification detection using Nanopore raw reads with Deep One-Class Classification”. en. In: *bioRxiv* (Sept. 2020), p. 2020.09.13.295089 (cit. on p. 47).

[69]Jonathan M Craig, Andrew H Laszlo, Ian C Nova, et al. “Determining the effects of DNA sequence on Hel308 helicase translocation along single-stranded DNA using nanopore tweezers”. en. In: *Nucleic Acids Res.* 47.5 (Mar. 2019), pp. 2506–2513 (cit. on p. 47).

[70]William Stephenson, Roham Razaghi, Steven Busan, et al. “Direct detection of RNA modifications and structure using single-molecule nanopore sequencing”. en. In: *Cell Genom* 2.2 (Feb. 2022) (cit. on pp. 47, 48).

[71]Adrien Leger, Paulo P Amaral, Luca Pandolfini, et al. “RNA modifications detection by comparative Nanopore direct RNA sequencing”. en. In: *Nat. Commun.* 12.1 (Dec. 2021), p. 7198 (cit. on p. 48).

[72]Franziska Horn, Robert Pack, and Michael Rieger. “The autofeat Python Library for Automated Feature Engineering and Selection”. In: *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, 2020, pp. 111–120 (cit. on p. 49).

[73]S Gopal Krishna Patro and Kishore Kumar Sahu. “Normalization: A Preprocessing Stage”. In: (Mar. 2015). arXiv: `1503.06462 [cs.OH]` (cit. on p. 53).

[74]Jerome H Friedman. “Greedy Function Approximation: A Gradient Boosting Machine”. In: *Ann. Stat.* 29.5 (2001), pp. 1189–1232 (cit. on p. 56).

[75]Tianqi Chen and Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’16. San Francisco, California, USA: Association for Computing Machinery, Aug. 2016, pp. 785–794 (cit. on p. 56).

[76]Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. “CatBoost: unbiased boosting with categorical features”. In: *Adv. Neural Inf. Process. Syst.* 31 (2018) (cit. on p. 56).

[77]Guolin Ke, Qi Meng, Thomas Finley, et al. “LightGBM: A highly efficient gradient boosting decision tree”. In: *Adv. Neural Inf. Process. Syst.* 30 (2017) (cit. on p. 56).

[78]Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. “Optuna: A Next-generation Hyperparameter Optimization Framework”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD ’19. Anchorage, AK, USA: Association for Computing Machinery, July 2019, pp. 2623–2631 (cit. on p. 56).

[79]Matthew T Parker, Katarzyna Knop, Anna V Sherwood, et al. “Nanopore direct RNA sequencing maps the complexity of Arabidopsis mRNA processing and m6A modification”. en. In: *Elife* 9 (Jan. 2020) (cit. on p. 56).

[80] Rupert G Fray and Gordon G Simpson. “The Arabidopsis epitranscriptome”. en. In: *Curr. Opin. Plant Biol.* 27 (Oct. 2015), pp. 17–21 (cit. on p. 58).

[81] Fadia Ibrahim, Jan Oppelt, Manolis Maragkakis, and Zissimos Mourelatos. “TERA-Seq: true end-to-end sequencing of native RNA molecules for transcriptome characterization”. en. In: *Nucleic Acids Res.* 49.20 (Nov. 2021), e115 (cit. on p. 58).

[82] Xian Adiconis, Adam L Haber, Sean K Simmons, et al. “Comprehensive comparative analysis of 5’-end RNA-sequencing methods”. en. In: *Nat. Methods* 15.7 (July 2018), pp. 505–511 (cit. on p. 58).

[83] Hazuki Takahashi, Sachi Kato, Mitsuyoshi Murata, and Piero Carninci. “CAGE (cap analysis of gene expression): a protocol for the detection of promoter and transcriptional networks”. en. In: *Methods Mol. Biol.* 786 (2012), pp. 181–200 (cit. on p. 64).

[84] Bo Yan, George Tzertzinis, Ira Schildkraut, and Laurence Ettwiller. “Comprehensive determination of transcription start sites derived from all RNA polymerases using ReCappable-seq”. en. In: *Genome Res.* (Nov. 2021) (cit. on p. 64).

[85] Jasmina Ponjavic, Boris Lenhard, Chikatoshi Kai, et al. “Transcriptional and structural impact of TATA-initiation site spacing in mammalian core promoters”. en. In: *Genome Biol.* 7.8 (Aug. 2006), R78 (cit. on p. 64).

[86] V A Efimov, O G Chakhmakhcheva, J Archdeacon, et al. “Detection of the 5’-cap structure of messenger RNAs with the use of the cap-jumping approach”. en. In: *Nucleic Acids Res.* 29.22 (Nov. 2001), pp. 4751–4759 (cit. on p. 65).

[87] Jonathan Neve, Radhika Patel, Zhiqiao Wang, Alastair Louey, and André Martin Furger. “Cleavage and polyadenylation: Ending the message expands gene regulation”. en. In: *RNA Biol.* 14.7 (July 2017), pp. 865–890 (cit. on p. 75).

[88] Bin Tian, Zhenhua Pan, and Ju Youn Lee. “Widespread mRNA polyadenylation events in introns indicate dynamic interplay between polyadenylation and splicing”. en. In: *Genome Res.* 17.2 (Feb. 2007), pp. 156–165 (cit. on p. 76).

[89] Brown Christine E. and Sachs Alan B. “Poly(A) Tail Length Control in Saccharomyces cerevisiae Occurs by Message-Specific Deadenylation”. In: *Mol. Cell. Biol.* 18.11 (Nov. 1998), pp. 6548–6559 (cit. on p. 76).

[90] Uwe Kühn, Miriam Gündel, Anne Knoth, et al. “Poly(A) Tail Length Is Controlled by the Nuclear Poly(A)-binding Protein Regulating the Interaction between Poly(A) Polymerase and the Cleavage and Polyadenylation Specificity Factor*”. In: *J. Biol. Chem.* 284.34 (Aug. 2009), pp. 22803–22814 (cit. on p. 76).

[91] Laure Weill, Eulàlia Belloc, Felice-Alessio Bava, and Raúl Méndez. “Translational control by changes in poly(A) tail length: recycling mRNAs”. In: *Nature Structural & Molecular Biology* 19.6 (2012), pp. 577–585 (cit. on p. 76).

[92] Sylke Meyer, Claudia Temme, and Elmar Wahle. “Messenger RNA turnover in eukaryotes: pathways and enzymes”. en. In: *Crit. Rev. Biochem. Mol. Biol.* 39.4 (July 2004), pp. 197–216 (cit. on p. 76).

[93]Stuart K Archer, Nikolay E Shirokikh, Claus V Hallwirth, Traude H Beilharz, and Thomas Preiss. “Probing the closed-loop model of mRNA translation in living cells”. en. In: *RNA Biol.* 12.3 (2015), pp. 248–254 (cit. on p. 76).

[94]Wolfgang Tomek and Karin Wollenhaupt. “The “closed loop model” in controlling mRNA translation during development”. en. In: *Anim. Reprod. Sci.* 134.1-2 (Sept. 2012), pp. 2–8 (cit. on p. 76).

[95]E Marshall, I Stansfield, and M C Romano. “Ribosome recycling induces optimal translation rate at low ribosomal availability”. en. In: *J. R. Soc. Interface* 11.98 (Sept. 2014), p. 20140589 (cit. on p. 76).

[96]Sarah Azoubel Lima, Laura B Chipman, Angela L Nicholson, et al. “Short poly(A) tails are a conserved feature of highly expressed genes”. en. In: *Nat. Struct. Mol. Biol.* 24.12 (Dec. 2017), pp. 1057–1063 (cit. on p. 76).

[97]Amrei Jänicke, John Vancuylenberg, Peter R Boag, Ana Traven, and Traude H Beilharz. “ePAT: a simple method to tag adenylated RNA to measure poly(A)-tail length and other 3’ RACE applications”. en. In: *RNA* 18.6 (June 2012), pp. 1289–1295 (cit. on p. 77).

[98]Alexander O Subtelny, Stephen W Eichhorn, Grace R Chen, Hazel Sive, and David P Bartel. “Poly(A)-tail profiling reveals an embryonic switch in translational control”. en. In: *Nature* 508.7494 (Apr. 2014), pp. 66–71 (cit. on p. 77).

[99]Hyeshik Chang, Jaechul Lim, Minju Ha, and V Narry Kim. “TAIL-seq: genome-wide determination of poly(A) tail length and 3’ end modifications”. en. In: *Mol. Cell* 53.6 (Mar. 2014), pp. 1044–1052 (cit. on p. 78).

[100]Jaechul Lim, Mihye Lee, Ahyeon Son, Hyeshik Chang, and V Narry Kim. “mTAIL-seq reveals dynamic poly(A) tail regulation in oocyte-to-embryo development”. en. In: *Genes Dev.* 30.14 (July 2016), pp. 1671–1682 (cit. on p. 80).

[101]Yusheng Liu, Hu Nie, Hongxiang Liu, and Falong Lu. “Poly(A) inclusive RNA isoform sequencing (PAIso-seq) reveals wide-spread non-adenosine residues within RNA poly(A) tails”. en. In: *Nat. Commun.* 10.1 (Nov. 2019), p. 5292 (cit. on p. 81).

[102]Michael G Mohsen and Eric T Kool. “The Discovery of Rolling Circle Amplification and Rolling Circle Transcription”. en. In: *Acc. Chem. Res.* 49.11 (Nov. 2016), pp. 2540–2550 (cit. on p. 101).

[103]Richard Durbin, Durbin Richard, Sean R Eddy, et al. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. en. Cambridge University Press, Apr. 1998 (cit. on p. 101).

[104]Maximilian Krause, Adnan M Niazi, Kornel Labun, et al. “tailfindr: Alignment-free poly(A) length measurement for Oxford Nanopore RNA and DNA sequencing”. en. In: *RNA* (July 2019) (cit. on p. 102).

[105]s Wright Erik. “Using DECIPHER v2.0 to analyze big biological sequence data in R”. en. In: *R J.* 8.1 (2016), p. 352 (cit. on p. 102).

[106] Oguzhan Begik, Huanle Liu, Anna Delgado-Tejedor, et al. “Nano3P-seq: transcriptome-wide analysis of gene expression and tail dynamics using end-capture nanopore sequencing”. en. In: *bioRxiv* (Oct. 2021), p. 2021.09.22.461331 (cit. on pp. 107, 109).

[107] Paola Munoz-Tello, Lional Rajappa, Sandrine Coquille, and Stéphane Thore. “Polyuridylation in Eukaryotes: A 3’-End Modification Regulating RNA Life”. en. In: *Biomed Res. Int.* 2015 (May 2015), p. 968127 (cit. on p. 107).

[108] Thomas E Mullen and William F Marzluff. “Degradation of histone mRNA requires oligouridylation followed by decapping and simultaneous degradation of the mRNA both 5’ to 3’ and 3’ to 5’”. en. In: *Genes Dev.* 22.1 (Jan. 2008), pp. 50–65 (cit. on p. 107).

[109] Yi Fang, Amogh Changavi, Manyun Yang, et al. “Nanopore whole transcriptome analysis and pathogen surveillance by a novel solid-phase catalysis approach”. en. In: *Adv. Sci.* 9.3 (Jan. 2022), e2103373 (cit. on p. 109).

[110] Satomi Mitsuhashi, So Nakagawa, Mitsuru Sasaki-Honda, et al. “Nanopore direct RNA sequencing detects DUX4-activated repeats and isoforms in human muscle cells”. en. In: *Hum. Mol. Genet.* 30.7 (May 2021), pp. 552–563 (cit. on p. 109).

[111] Jessie J-Y Chang, Josie Gleeson, Daniel Rawlinson, et al. “Long-read RNA sequencing identifies polyadenylation elongation and differential transcript usage of host transcripts during SARS-CoV-2 in vitro infection”. en. In: *bioRxiv* (Dec. 2021), p. 2021.12.14.472725 (cit. on p. 109).

[112] Luca Cozzuto, Huanle Liu, Leszek P Pryszcz, et al. “MasterOfPores: A Workflow for the Analysis of Oxford Nanopore Direct RNA Sequencing Datasets”. en. In: *Front. Genet.* 11 (Mar. 2020), p. 211 (cit. on p. 109).

List of Figures

2.1 **Nanopore sequencing of a native RNA molecule.** An engineered protein pore is suspended in a membrane which separates buffer-filled wells on *cis* and *trans* sides of a membrane. Across the membrane a voltage is applied. The translocation of the RNA in the DNA-RNA heteroduplex is controlled by the ratcheting action of a motor protein. A base — or any modification present on it — in the central constriction of the pore along with the two bases upstream and downstream of this central base (the 5-mer context) affects the ionic current signal that comes out of the pore. 11

2.2 **Nanopore FAST5 file structure information. a)** A Raw FAST5 file contains only the raw signal. **b)** A basecalled FAST5 file contains additional information such as the FASTQ, Move and Trace tables, and **c)** various attributes most important of which are the `block_stride` and the `first_sample_template`. **d)** The Move table in which each row is an event. Each event corresponds to a stretch of raw signal equal to the value of `block_stride`. A move of 1 represents the detection of a new base in that event, while a move of 0 represents that no new base has been detected and that the base from the previous event is still persisting in the current event. **e)** By using the Move table and information `block_stride` and `first_sample_template` attributes, it is possible to create a mapping between the raw signal samples and the basecalled sequence. **f)** Simplified form of the table in e. **g)** Raw data start and end indexes in table f can be used to annotate the raw Nanopore squiggle with base predictions. In this way, one-to-one mapping between the raw signal and basecalls can be obtained. 18

2.3 **A plot of the Trace table against the Nanopore signal and the base-called sequence.** The flip and flop probabilities switch (vertical arrows) whenever the basecaller thinks that one base within a homopolymer has passed and a new homopolymer base has started to pass through the pores constriction. If a modified base passes through the pore (for example ^{m}G), then the base probability gets split up between all the bases rather than being high only for one base. This is a tell-tale sign that the base passing through the pore is different (in other words modified) compared to the bases that the basecaller was trained on. . . 19

3.1 **RNA capping mechanism. a)** Step 1: RNGTT's TPase domain cleaves off γ-phosphate at the end of the nascent pre-mRNA. **b)** Step 2: RNGTT's GTP domain adds guanosine monophosphate to the end of RNA (produced in the first step 1). **c)** Step 3: RNMT methylates terminal guanine. 22

3.2 **Chemical structure of different canonical cap structures in mRNA.** The bond between terminal m^{7}G and the first transcribed nucleotide is formed due to an unusual 5'-to-5' linkage. This 5'-to-5' linkage only happens at the cap and nowhere else is the RNA as all the rest of the nucleotides in RNA are connected to each other with 5'-to-3' linkages. Some canonical are known to exist in biology (top table), while others are not yet known to exist (bottom table) 24

3.3 **Chemical structure of eukaryotic non-canonical initiating nucleotide (NCIN) caps.** 25

3.4 **Radio-isotope labeling-based method for quantifying different cap types in a sample** 29

3.5 **Mass-spectrometry based method for quantifying different cap types present in a sample** 30

3.6 **NGS-based method for quantifying different cap types in a sample** 32

3.7 **NAD tagSeq II protocol for studying NAD-capped RNA transcripts with Nanopore sequencing** 34

3.8 **Loss of processive control of 5'-end of RNA during Nanopore sequencing. a)** The motor protein ratchets RNA at a slow controlled speed until it reaches the 5'-end of the RNA. **b)** When the sequencing reaches the very end of the RNA molecule, the motor enzyme can no longer grab onto the molecule and falls off. With the processive control of the motor enzyme now gone, the ten nucleotides (shown in gray) that still need to be sequenced, go through the pore so fast that their current signature is undecipherable. **c)** IGV view of the alignment of basecalled reads with the reference shows that 10-20 bases are mostly missing from the 5'-end. 36

3.9 **Our method for decoding cap types in mRNA. a)** Classifier training. To create training data for the classifier, 67-nt long synthetic oligos are ligated to longer carrier GFP molecules. The synthetic oligos contain 44-nt long GeneRacer sequence followed by two bases (numbered 1 and 2) that may carry a 2'-O methylation depending on the cap type that the oligo is emulating (as shown in the accompanying table). Features are extracted from 26 positions (numbered in gray from –5 to 20) from the sequenced Nanopore data and are used to train the classifier. **b)** Cap type predictions in biological mRNA. The inverted m^7G is first removed by Cap-Clip enzyme, followed by oligocapping with GeneRacer sequence from the GeneRacer kit. Features are extracted from the sequenced Nanopore data and the trained classifier from step (a) is used to predict the cap types. 38

3.10 **Synthetic capped oligos for classifier training. a)** Relative abundance of different transcription start site dinucleotides determined from CAGE human and zebrafish datasets in FANTOM5 database. Nine out of sixteen possible cap dinucleotides (highlighted in green) have been prioritized for making training oligos for now. **b)** Uptil now, we have synthesized nine out of 48 possible cap flavors in four (cap0, cap1, cap2, and cap2,-1) out of five cap classes. The classifier is currently trained on these cap flavors while the remaining cap flavors are being sequenced. The number of reads for these classes show that we have a highly imbalanced dataset. 40

3.11 **Data analysis workflows in *capable*. a)** Workflow for processing Nanopore data from synthetic oligos used for training the classifier. **b)** Workflow for predicting cap types in Nanopore reads from biological mRNA using the pretrained classifier obtained in (a). 42

3.12 **Minimap2 alignments at 5'-end of mRNA transcript. a)** Minimap2 mostly fails to align the 5'-end of the mRNA transcripts due to its inherent limitations. The result is unaligned or soft-clipped bases (shown as colored letters in the reads) at the 5'-end. This presents a major hurdle in determining accurate transcription start site loci of these reads. **b)** Alignment after polishing the 5'-ends of alignments in (a). After performing a polishing step, the previously soft-clipped bases are now successfully aligned to the reference genome. The 5'-ends of the alignments now represent the transcription start sites. 45

3.13 **Dwell time distributions for three different 5-mers in a modification-free RNA dataset.** Some kmers (such as ACGCT in this case) have a very sharp dwell time distribution and lower median dwell time, whereas other kmers (such as GATAT and TGAAG) have a very wide distribution and higher median dwell times. 48

3.14 **Dwell time of cap and cap-proximal bases in the training oligos. a)** Raw dwell time for different synthetically-made cap oligos. Cap dinucleotides are present at positions 0, and 1. The methylations present in different cap types affect the dwell time of the cap bases and the flanking bases, and also the bases that are located 11 and 12 nucleotides downstream of the cap in the spacer. This raw dwell time is highly dependent on the sequence context. For example, the kmers in positions 5, 8, and 10 have a very small dwell time whereas kmers in positions -3, 4, and 6 have a high dwell time. Furthermore, different kmers have widely different value ranges. **b)** To remove kmer sequence-specific effects from the dwell time, and to capture only the 2'-O methylation-specific modulations in the dwell, this engineered feature represents how many percentiles points away the observed dwell time of a kmer is from a corresponding unmodified kmer's median. This feature confines the dwell time between –50 to +50 range, removes kmer-sequence specific effects, and amplifies the effect of cap modifications. Features at different positions were combined using mathematical operations to create new engineered features that help the classifier learn that the cap 2'-O methylation results in simultaneous changes in dwell at both the cap positions and also at the downstream positions. 50

3.15 **Mechanism of dwell time change due to cap 2'-O methylations. a)** When the cap is in the motor protein, the motor protein struggles to ratchet these methylated bases causing the bases 11 and 12 positions downstream of the cap which are currently being sampled in the constriction of the pore to spend an unusually long amount of time in the pore. Consequently, the bases in positions 11 and 12 have a higher dwell compared to if there were no 2'-O methylations on the cap bases. **b)** The second instance the cap 2'-O methylations result in a change in the dwell time is when the cap bases are passing through the constriction of the pore. The methylations in these cap bases cause prolonged interaction of these bases with the pores constriction site, thereby resulting in a longer dwell time of cap bases at position 0 and 1. 51

3.16 **Nanopore current for training oligos a)** Raw current level. Different kmers have different current levels which are further modulated by the cap modifications. Because different cap type oligos were sequenced in different runs and the biasing voltage is dynamically adjusted by MinKNOW in each run, this results in a run-specific shift in current as evident from the different overall current levels for the different cap types. A classifier trained on Nanopore current should only learn cap-specific modulations and not sequence-specific or run-specific effects in the current. **b)** Standardized current level obtained by first shifting (or normalizing) the observed current level towards the pore model using the Theil-Sen estimator, and then standardizing the resulting normalized current by subtracting the model kmer mean and dividing by the model kmer standard deviation. Engineered features were made by combining features at different positions with mathematical operations and used during model training. Only features from positions that are shown bold and underlined were used during model training. 52

3.17 **Standard deviation in current for different cap types.** Engineered features were made by combining features at different positions with mathematical operations and used during model training. Only features from positions that are shown bold and underlined were used during model training. 53

3.18 **Basecall quality for training oligos. a)** Raw basecall quality. **b)** Min–max normalized basecall quality. Engineered features were made by combining features at different positions with mathematical operations and used during model training. Only features from positions that are shown bold and underlined were used during model training. 54

3.19 **Log_2-transformed Guppy base probabilities in FAST5 Trace table.** Engineered features were made by combining features at different positions with mathematical operations and used during model training. Only features from positions that are shown bold and underlined were used during model training. 54

3.20 **Number of insertions in the alignment to a reference when 2'-O cap methylations are sequenced.** Only features from positions that are shown bold and underlined were used during model training. . . . 55

3.21 **Alignment CIGAR of different cap oligos used during classifier training.** Only features from positions that are shown bold and underlined were used during model training. 55

3.22 **Feature importance plot of all the features used during training.** . 57

3.23 **Assessing 5' RNA degradation in TERA-Seq dataset. a)** We define RNA degradation as the distance between observed TSS from Nanopore reads and the nearest CAGE-supported TSS in the FANTOM5 database. **b)** A plot of distance between the distance between observed TSS from Nanopore reads and the nearest CAGE-supported TSS in TERA-Seq dataset. Only 42% of the Nanopore reads have no degradation. 59

3.24 **IGV view of alignment** The coverage for the transcripts drops at the 5'-end showing rampant 5'-RNA degradation during library prep. . . . 60

3.25 **Assessing 5' RNA degradation in TERA-Seq dataset. a)** We define RNA degradation as the distance between observed TSS from Nanopore reads and the nearest CAGE-supported TSS in the FANTOM5 database. **b)** A plot of distance between the distance between observed TSS from Nanopore reads and the nearest CAGE-supported TSS in TERA-Seq dataset. Only 42% of the Nanopore reads have no degradation. 61

3.26 **Structure of capped mRNA transcripts.** A 2', 3'-*cis*, diol is present only on ribose sugar of 5' and 3' terminal nucleotides. Internal ribose sugars do not possess a 2', 3'-*cis*, diol. 66

3.27 **Cap-jumping protocol for sequencing RNA caps. a)** Steps involved in preparing the OTE for ligation. **b)** Step involved in preparing the capped RNA for ligation. **c)** Ligating the previously-prepared OTE and capped RNA in (a) and (b), and making the ligated product amenable for sequencing on a Nanopore. 67

3.28 **Feasibility of cap-jumping approach for non-canonical caps.** Cap-jumping approach can work with all eukaryotic non-canonical caps because all of them an exposed diol – OH groups highlighted in black. 68

3.29 **Duty time plots for sequencing runs with libraries prepared with the cap-jumping method. a)** Plot for the NAD library. The pores are all dead during the first hour of the sequencing run. **b)** Plot for m^7G library. The pores die more gradually compared to the NAD library. . . 69

3.30 **Nanopore current traces for transcripts sequenced with the cap-jumping approach.** The top current trace originated from sequencing a non-canonical NAD^+-capped transcript, whereas the bottom trace shows the current for the canonical m^7G-capped transcript. A large current spike in the signal is seen when the cap and linking chemistry containing the two morpholine rings are sequenced. 70

3.31 **IGV view of the alignment of three OTE–ligated reads in the m^7G cap-jumping library.** The OTE is the first 100 nucleotides of the reference followed by the cap (first and second transcribed nucleotides) and the rest of the transcript sequence. The ligation chemistry causes basecalling errors in 10-12 bases of the transcript sequence. The colored bases in the alignments are soft-clipped (unaligned) bases. 73

3.32 Close proximity of the two morpholine rings in the current cap-jumping protocol as shown in **(a)** can cause steric hindrance during ratcheting. This may be reduced by increasing the distance between the two morpholine rings with a longer carbon chain as shown in **(b)**. 73

4.1 The extension polyA-tail test (ePAT) for poly(A) tail profiling. 78

4.2 Poly(A) profiling by sequencing (PAL-seq) 79

4.3 TAIL-seq method for poly(A)-tail profiling 80

4.4 PacBio-sequencing based PAIso-seq method which can be used to study poly(A) tails alongwith their full-length transcript isoforms in cDNA (not in native RNA) 82

4.5 **Limitation of short-read sequencing methods for poly(A)-tail profiling.** Poly(A) tail length estimates obtained from short-read sequencing-based methods make it difficult to assign the estimated poly(A) tail uniquely to a particular transcript isoform if the different isoforms have the same 3'-end sequence. 83

4.6 **Nanopore sequencing of poly(A) tails. a)** Homopolymer compression in Nanopore basecalls prevents us from finding the precise poly(A) tail length. **b)** *tailfindr* uses the information encoded in the length of the poly(A) tail signal to infer the poly(A) tail length in nucleotide units. . 85

4.7 Our approach for amplifying poly(A)+ RNA into cDNA 100

4.8 **Poly(A)-tail profiling in cDNA poly(A) standards with poly(A) tail lengths 10, 30, 60, 100 and 150 nt. a)** We expected to see nice peaks for each of the five different standards we sequence. **b)** What we observe is the longer tails (60, 100, and 150nt) were shortened to around 30 nt, and shorter tails such as 10 nt and 30 nt poly(A) were elongated to 25 nt and 35 nt, respectively. 101
4.9 Mispriming of partially-extended PCR primers can cause the poly(A) tails in amplified cDNA to be longer or smaller than the poly(A) tail in the original template. The poly(A) tail lengths in this amplified cDNA, therefore, do not faithfully represent the true poly(A) tail lengths in the starting RNA sample. 102
4.10 **Protocol for producing amplified cDNA from RNA using rolling-circle amplification.** The resulting reads have concatemers of original linear cDNA sequences containing poly(A) and poly(T) stretches. . . . 103
4.11 *tailfindr's* approach to estimate poly(A) tail lengths in concatemers produced from rolling-circle amplification 104
4.12 **Results of poly(A)-tail profiling on rolling-circle amplified cDNA. a)** We expected to see two peaks around 30 and 100 nt (top panel) corresponding to two different poly(A) standards used in the experiments, but we observed a single peak around 18 nt (bottom panel). **b)** Majority of the concatemers had only one repeat in them. **c)** We used transcript sequences of length 750 nt in the experiment, but the majority of the transcripts recovered from the individual repeats were not much shorter than 750 nt. 105
4.13 **PCR-cDNA sequencing protocol of SQK-PCS111 kit.** The kit digests away part of the splint adapter before reverse transcription which helps prevent mispriming during PCR cycles and consequently leads to faithful reproduction of poly(A) tails in amplified cDNA. 106
4.14 **Poly(A)-tail profiling results. a)** on RNA tail standards **b)** on cDNA tail standards obtained by PCR amplification of RNA standards in (a) into cDNA with the newly-released PCR-cDNA kit PCS111. 106

List of Tables

2.1 **Comparison of different methods for sequencing RNA on Nanopore.** 15

3.1 **A snippet of the output of the sam2tsv tool.** A 1-to-1 map between bases in the reads and the references to which they are aligned to is created by *sam2tsv* tool by using alignment information in an alignment SAM file 46

3.2 **Summary of cap type predictions in different datasets.** 58

3.3 **Comparison of `end_reason` for all the reads in m^7G and NAD cap-jumping libraries.** Reads in the NAD cap-jumping library are blocking the pores more than normal. The reads in m^7G cap-jumping library are behaving like normal reads. 72

3.4 **Comparison of `end_reason` for only OTE–ligated reads in m^7G and NAD cap-jumping libraries.** Majority of the reads in the m^7G library went through the pore normally without triggering a read ejection event. Only one read in this library resulted in a pore blockage and subsequent ejection. 72

Appendix A

Chapter 30

Transcript Isoform-Specific Estimation of Poly(A) Tail Length by Nanopore Sequencing of Native RNA

Adnan M. Niazi, Maximilian Krause, and Eivind Valen

Abstract

The poly(A) tail is a homopolymeric stretch of adenosine at the 3′-end of mature RNA transcripts and its length plays an important role in nuclear export, stability, and translational regulation of mRNA. Existing techniques for genome-wide estimation of poly(A) tail length are based on short-read sequencing. These methods are limited because they sequence a synthetic DNA copy of mRNA instead of the native transcripts. Furthermore, they can identify only a short segment of the transcript proximal to the poly (A) tail which makes it difficult to assign the measured poly(A) length uniquely to a single transcript isoform. With the introduction of native RNA sequencing by Oxford Nanopore Technologies, it is now possible to sequence full-length native RNA. A single long read contains both the transcript and the associated poly(A) tail, thereby making transcriptome-wide isoform-specific poly(A) tail length assessment feasible. We developed *tailfindr*—an R-based package for estimating poly(A) tail length from Oxford Nanopore sequencing data. In this chapter, we describe in detail the pipeline for transcript isoform-specific poly(A) tail profiling based on native RNA Nanopore sequencing—from library preparation to downstream data analysis with *tailfindr*.

Key words Poly(A) tail, Nanopore sequencing, Native RNA, *tailfindr*, R, Transcriptomics

1 Introduction

A poly(A) tail is formed by the nontemplated addition of a stretch of adenosines to the 3′-end of messenger RNA (mRNA) during RNA processing in the nucleus [1]. It mediates the transfer of processed RNA from nucleus into the cytoplasm in eukaryotes [2]. Furthermore, it is known to stabilize or destabilize the mRNA depending on its length: relatively long poly(A) tails inhibit degradation of mRNA by 3′-exonucleases and 5′-cap hydrolysis, whereas short poly(A) tails mark the mRNA for degradation by the exosome [3]. Additionally, the length of the poly(A) tail can, under certain conditions, influence the translational efficiency of the

Adnan M. Niazi and Maximilian Krause contributed equally with all other contributors.

Ernesto Picardi (ed.), *RNA Bioinformatics*, Methods in Molecular Biology, vol. 2284, https://doi.org/10.1007/978-1-0716-1307-8_30,

mRNA [4–6]. Measuring isoform-specific poly(A) tail length over the whole transcriptome is therefore important in understanding its role in regulation of mRNA localization, mRNA half-life and translation regulation.

Existing methods for transcriptome-wide estimation of poly (A) tail length—which are primarily based on Illumina short-read sequencing technology [5, 7, 8]—have numerous limitations. First, RNA in its native form cannot be sequenced using Illumina sequencing: the RNA must first be reverse transcribed into cDNA, and subsequently amplified with PCR cycles to form clusters on the flow cell that are sequenced by synthesis. The conversion of RNA into cDNA results in loss of information; for example, the occurrence of native RNA modifications might be interesting to study along with the poly(A) tail length. Second, the repeated PCR cycles may introduce artefacts in the homopolymer regions that may cause errors in poly(A) tail length estimation [9–11]. Third, most of these methods estimate poly(A) tail length indirectly by inferring cDNA poly(A) or poly(T) segments using elaborate library preparation steps or custom-designed software for processing raw images of the sequencing clusters.

This renders these methods not only time-consuming but also technically challenging. Lastly, as Illumina sequencing is a short-read sequencing technology, a sequenced read from these methods contains only a small segment reflecting parts of the transcript proximal to the poly(A) tail. With such partial transcript fragments, transcript isoform-specific poly(A) tail assignment is hard, and in many instances impossible. This is because a read may align equally well to two or more transcript isoforms, making it impossible to decipher as to which transcript the read—and its associated poly (A) tail measurement—belongs to (*see* Fig. 1). Until recently it was therefore impossible to address whether different transcript isoforms have different poly(A) tail lengths.

With the advent of long read sequencing methods it recently became possible to sequence full length transcripts and their associated poly(A) tails [12–14]. In addition to offering long read sequencing only limited by the molecules integrity [15], Oxford Nanopore Technologies (ONT) novel sequencing approach also allows to sequence native RNA molecules without the conversion into cDNA [16]. This new technology has the potential to address isoform-specific poly(A) length measurements and RNA modification detection in a single assay [14, 17].

In this chapter, we will explain how ONT's sequencing approaches allow direct poly(A) measurement of native RNA (Subheading 2), describe the necessities for efficient Nanopore library preparation (Subheading 3), and how to process the data generated using *tailfindr* to perform transcriptome-wide isoform-specific poly(A) tail profiling (Subheading 4).

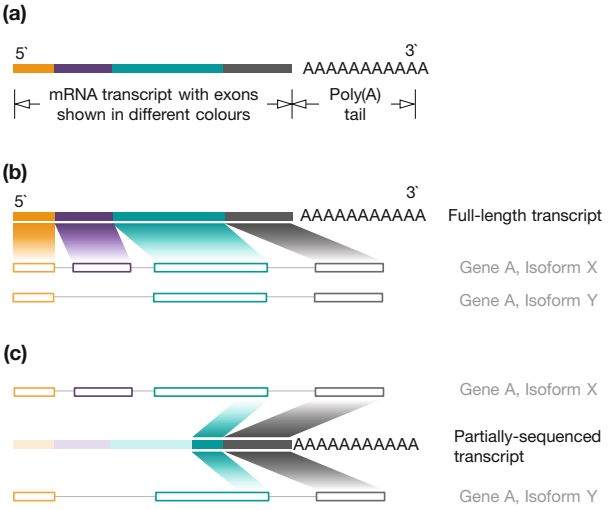


Fig. 1 (**a**) A poly(A)-tailed mRNA. (**b**) A full-length transcript uniquely and unambiguously maps to the isoform that it originated from. In the illustrated case, the read perfectly aligns to isoform X of gene A, and the measured poly (A) tail length can be uniquely attributed to isoform X. (**c**) A partially sequenced transcript can map equally well to multiple transcript isoforms, making it impossible to decipher from which of the many possible isoforms the read originated from. In this case, the partially sequenced transcript aligns equally well to both isoform X and isoform Y of gene A. Thus transcript-isoform specific poly(A) tail length assignment is not possible

2 Nanopore Sequencing

In ONT sequencing approaches, a protein nanopore is suspended in a hydrophobic material (membrane) that separates two buffer-filled wells [18]. A cross-membrane voltage of −180 mV is applied such that the *trans* side of the membrane is set at a positive potential compared to the *cis* side (*see* Fig. 2). This causes a constant ionic current to flow through the pore. The molecule to be sequenced, which can be either DNA or RNA, is located on the *cis* side of the membrane. Under the influence of the applied voltage, the negatively charged nucleotide strand threads through the pore. To ensure a homogeneous translocation rate (450 bps for DNA and 70 bps for RNA [19]), and to minimize the influence of secondary structure or DNA duplex binding energy, the DNA or RNA is fed into the pore by the ratcheting action of a motor protein. The nucleotides located in the constriction of the pore—5–6 nucleobases at any given time—modulate the current passing through the

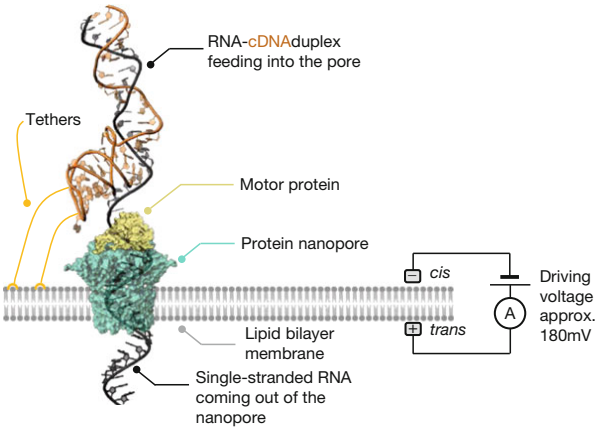


Fig. 2 RNA sequencing using ONT Direct-RNA sequencing**.** The RNA to be sequenced is first reverse-transcribed to make an RNA–cDNA duplex; this step removes RNA secondary structure that may otherwise cause pore blockage. The RNA–cDNA duplex, along with the ligated adaptor that contains the motor protein, is initially located on the *cis* side of the membrane. The tethers attached to the DNA adaptor have an affinity for the lipid membrane and help anchor the RNA–cDNA duplex to it. Under the influence of the applied voltage, the duplex shifts toward the pore, and eventually the RNA part of the duplex threads through the pore. The motor protein unwinds the RNA–cDNA duplex, and ratchets the RNA through the nanopore one base at a time. The fluctuations in the pore current as the RNA strand translocates through the pore are recorded

pore, thereby creating sequence-specific modulations in the current. This current is sampled at a rate of 3012 samples/second and saved as an array in a `.fast5` file by MinKNOW—the data acquisition and experiment management software provided by ONT. The resulting signal trace—the so-called squiggle—thus contains the information of the contiguous nucleotide strand and possible RNA modifications and should be stored as "raw data files." The raw data files are then used by a basecaller to predict the original sequence.

In the special case of ONT native RNA sequencing, the motor protein is added at the 3′-end of the molecule by poly(A)-guided ligation. Reverse transcription is optional, as the synthesized cDNA strand will not be sequenced at any time. Nevertheless, it is recommended to perform reverse transcription, as the resulting RNA–cDNA heteroduplex is devoid of secondary structure that potentially interferes with pore translocation. Furthermore, the RNA–cDNA heteroduplex is more stable than single-stranded RNA toward degradation by RNases (*see* **Note 1**). The added motor

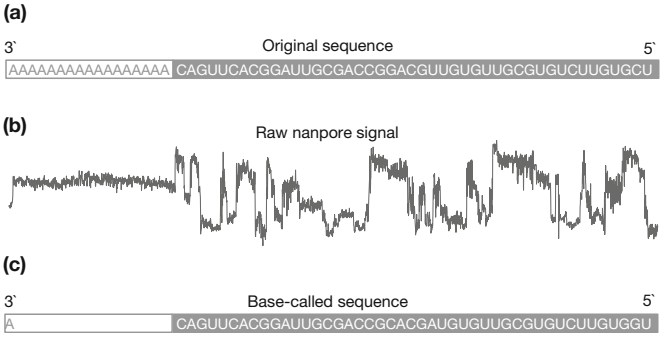


Fig. 3 Current basecalling algorithms underestimate poly(A) tail length. (**a**) A full-length mRNA with a 17-nt long poly(A) tail. (**b**) Raw signal generated by ONT sequencing when the sequence shown in (**a**) passes through the nanopore. Notice that the signal corresponding to the poly(A) tail is low-variance and monotonous. (**c**) Sequence predicted by the basecaller. Notice that the basecaller predicts only one adenosine in the poly(A) tail whereas the original sequence has 17 adenosines in the poly(A) region. This shows that although the raw signal for poly(A) tail is captured using Nanopore sequencing, it is not basecalled properly, preventing poly (A) tail length estimation directly from basecalling. N.B.: The sequences shown in this figure represent exemplified data

protein threads the RNA through the pore from its 3′-end to the 5′-end. The resulting current signal thus contains in this order: signal for the adaptor sequence that initially carried the motor protein, the poly(A) tail and the full-length transcript.

Although in theory it should be possible to infer the length of the poly(A) tail from the basecalled sequence alone, in practice this is not the case. When the raw signal is basecalled, the number of adenosines (reflected as A in sequence) called by the current basecallers in the poly(A) tail region is far lower than the actual number of adenosines in the poly(A) tail of the original RNA sequence (*see* Fig. 3). This is because the raw signal corresponding to a homopolymeric stretch of adenosine is a monotonous current devoid of any detectable transition from one adenosine to the next [20, 21]. The basecaller cannot decide where the signal of one adenosine ends and the next one starts; the entire poly(A) tail signal is therefore treated as a single adenosine base that got stalled in the nanopore for a long time. Thus, the poly(A) tail length currently cannot be faithfully estimated from basecalled sequences directly as it will often underestimate the actual poly(A) tail length. To accurately estimate the poly(A) tail length from Nanopore sequencing data, we developed an R package—*tailfindr* [12]. The software uses basecalled `.fast5` files and annotates the reads with poly(A) tail estimates (for more details refer to Subheading 4).

In the following sections, we will describe how to successfully perform library preparation for native RNA sequencing using Nanopore, and how to use *tailfindr* to obtain isoform-specific poly(A) tail measurements from the obtained data.

3 Library Preparation

ONT sequencing provides single-molecule long-read sequencing applications for RNA for the first time. However, the quality of the produced data and—most importantly—the quantity of data output directly depends on the quantity and quality of the provided RNA. It is therefore essential to make sure that enough RNA of good quality can be achieved prior to planning the experiment. Any RNA degradation not only affects the read length of the data obtained, but also makes library preparation inefficient, as it is based on poly(A)-dependent ligation of DNA adapters (Fig. 4). Therefore, all experimental procedures upstream of sequencing should be reviewed for forces that could degrade molecules, such as vigorous shaking or pipetting. Furthermore, RNA should be extracted as fresh as possible, or alternatively stored at -80C in RNA storage medium (TRI reagent or RNALater). Extraction should be chosen to avoid any contaminants, as these could be detrimental to the sequencing chemistry. In our experience, silica-column based purification strategies not only degrade RNA by physical force, but also retain Guanidine-hydrochloride contamination. We thus recommend the use of phenol-chloroform extraction methods, such as the use of TRI reagent. These are more time-consuming, but in our hands yield higher quality RNA with minimal contaminant carry-over. An example workflow for the use of TRI reagent for purification, as well as poly(A) enrichment based on the Poly(A)Purist MAG Kit, is described in an exemplary protocol at the end of this section.

Enriching for poly(A)-containing RNA is necessary in current ONT protocols, as the adapters are added specifically to the poly (A) tail. The addition of adapters happens through RNA ligation. However, the presence of high amounts of nonpolyadenylated RNA (such as rRNA) can significantly impact the efficiency of adapter ligation by titrating the enzyme or adapters by nonproductive binding events. Poly(A) enrichment based on magnets is the gold-standard experimental approach, but any other strategies that do not involve physical forces—such as vortexing, vigorous pipetting, or column-based purification—would work as well.

The efficiency of library preparation solely depends on the efficiency of DNA–RNA ligation procedures. A schematic workflow of Nanopore Library preparation is provided in Fig. 4. Any contaminant that reduces ligation efficiency will impact the sequencing performance of the library. It is thus important to follow the

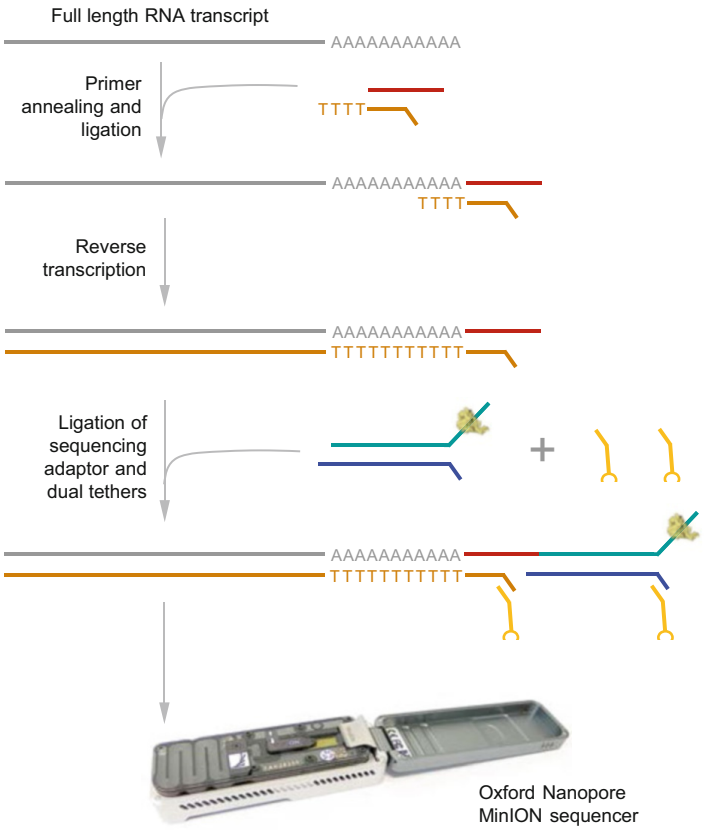


Fig. 4 Representation of ONT Direct-RNA library preparation protocol. The reverse transcription adaptor containing a T-overhang is ligated to the full-length RNA (shown in grey). This adaptEr can only bind at the 3′-end of the transcript and initiates reverse transcription. The reverse transcription creates a DNA strand (shown in orange). In this way, an RNA–cDNA duplex is formed. Next, a sequencing adaptor containing the motor protein is ligated to the RNA–cDNA duplex along with dual tethers. During sequencing on a ONT MinION sequencer, these tethers anchor the DNA strand to the lipid bilayer membrane, which helps to efficiently feed the RNA strand through the pore

recommendations given in the Nanopore protocols (nanoporetech.com) for RNA quality and quantity measures. The only exception are ligation and bead purification incubation times, which we routinely double. A longer incubation time at room temperature might increase the risk of RNA degradation, yet also increases the chance of successful ligation or DNA binding or elution events to beads, which leads to a more efficient library preparation. Finally, it is

crucial to proceed quickly from the final ligation to actual sequencing and avoid harsh chemicals and temperatures with the final library, as an active protein has been added whose function is essential for sequencing. An example protocol for library preparation including total RNA extraction and poly(A) enrichment together with notes arising from our library preparation experience can be found at dx.doi.org/10.17504/protocols.io.9cjh2un.

4 Bioinformatics Analysis

To accurately estimate the poly(A) tail length from Nanopore native RNA sequencing data, we developed an R package—*tailfindr* [12]. Briefly, *tailfindr* estimates poly(A) tail length by first locating the monotonous stretch of current signal corresponding to the poly(A) tail within the raw signal, and then calculating its duration in samples (*see* Fig. 5). Next, a read-specific translocation rate is computed; it specifies the average of samples per nucleotide translocation. After estimating this translocation rate, it is used to normalize the tail length in samples found earlier to yield tail length in nucleotides. During all these steps, *tailfindr* only needs basecalled FAST5 files to estimate the poly(A) tail length, making it independent of downstream data processing and thus implementable in real-time data analysis pipelines. The following paragraphs will give you detailed instructions on how to use *tailfindr* toward obtaining isoform-specific poly(A) measurements from Nanopore native RNA sequencing.

4.1 Requirements

4.1.1 Test Dataset

We extracted RNA from Zebrafish (*Danio rerio*) using the protocol described above, and sequenced it on a MinION sequencer. A subset of reads from this experiment can be downloaded from tiny.cc/polya_rna_data. We will now demonstrate the various steps involved in transcript isoform-specific poly(A) tail length assessment using this example dataset, but you can use your own dataset as well.

4.1.2 Hardware Requirements

The example dataset can be processed on any laptop or desktop computer running a UNIX-based operating system with at least 3GB of free disk space. For a large real-world dataset, it is recommended that the pipeline is run on a Linux cluster, or a powerful workstation. For accelerating the basecalling speed, GPUs can be used. For more details on which GPUs are compatible with the current basecaller, please refer to this link: https://community.nanoporetech.com/posts/guppy-3-0-gpu-recommendati (requires community login).

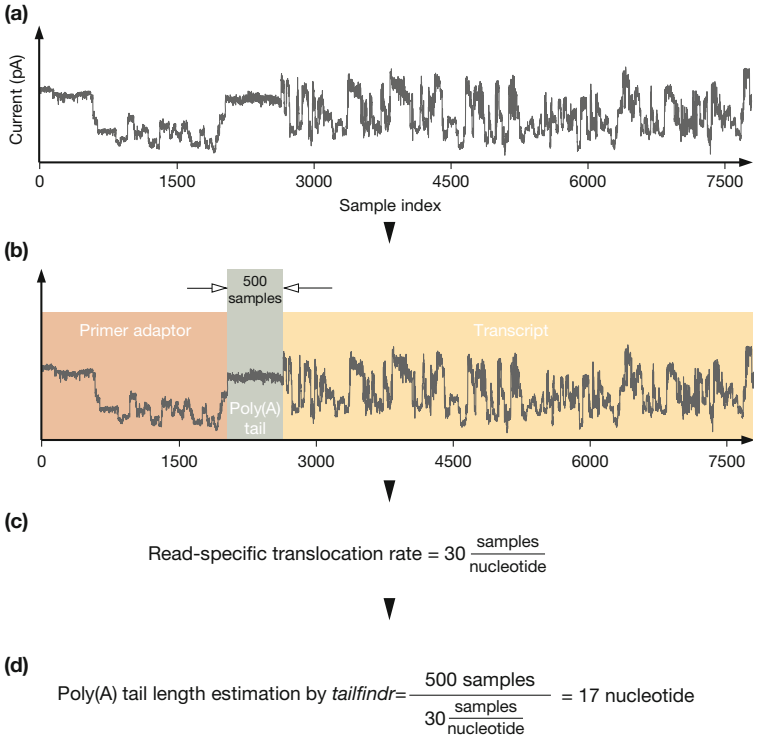


Fig. 5 A simplified view of how *tailfindr* estimates poly(A) tail length. (**a**) Complete raw signal corresponding to an RNA transcript translocating through the pore. The signal consists of a series of current samples measured in picoAmperes (pA). (**b**) *tailfindr* first locates the monotonous signal corresponding to the poly(A) tail (highlighted in brown). In this example, the segment is 500 samples long. (**c**) Next, *tailfindr* estimates the read-specific translocation rate, that is, the average number of samples generated per nucleotide in a given read. (**d**) Poly(A) length is calculated by dividing the tail length in samples by the read-specific translocation rate

4.1.3 Software Requirements

The following software should be installed on the analysis computer:

- Python 3 environment.
- R (version 3.5.3 or greater).
- Git.

4.2 Data Analysis Pipeline

There are various steps involved in going from raw reads produced by ONT sequencing to transcript isoform-specific poly(A) tail length assignment, as shown in Fig. 6. We will now describe each of these steps in detail.

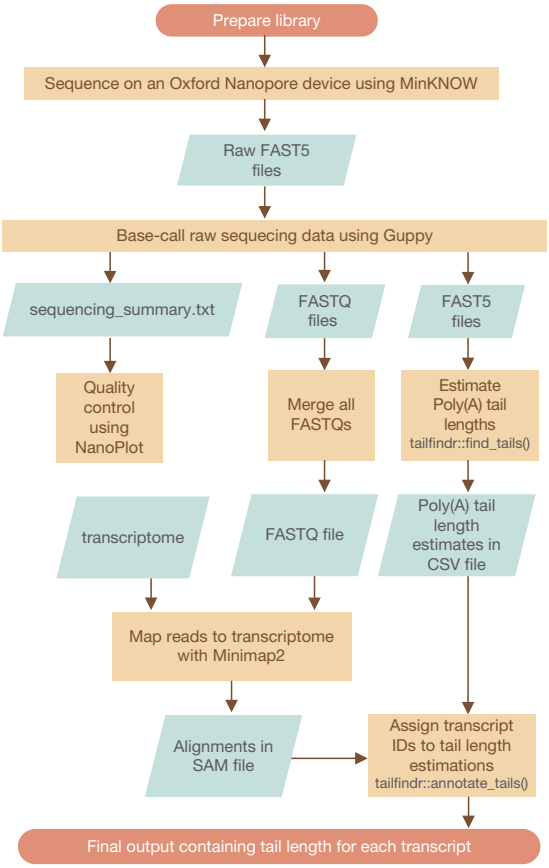


Fig. 6 Flowchart for poly(A) tail length estimation using Nanopore sequencing and *tailfindr*

4.2.1 *Basecalling*

Nanopore sequencing produces raw FAST5 files that record the current signal through the pore as an RNA molecule translocates through it (*see* Fig. 7a). The first step is to basecall this raw signal to find the nucleotide sequence corresponding to the recorded current. There are many basecallers that can do this; please refer to [22] for a review on this topic. Some of the basecallers have been developed by ONT, while others are developed by Nanopore users. Albacore is a widely used basecaller developed by ONT. Guppy—a recently released basecaller, also developed by ONT—has now replaced Albacore because it has better basecalling performance

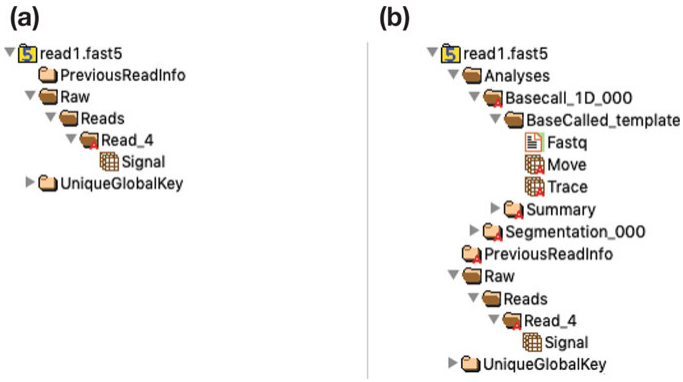


Fig. 7 Structure of a FAST5 file as displayed by HDFView software. (**a**) File structure of a raw FAST5 file generated by MinKNOW. (**b**) The same file as in (**a**) after basecalling by Guppy. During basecalling a new FAST5 is generated that contains not only raw signal data, but also additional basecalling information. Notice how additional levels of information (`Analyses`, `Basecall_1D_000` etc.) have now been added in this new file

and is faster than its predecessor. We recommend using the latest version of Guppy which can be downloaded from the ONT Community website https://community.nanoporetech.com/downloads.

Basecalling a raw FAST5 file using Guppy will add a `Basecall_1D_000` group to the FAST5 file hierarchy (*see* Fig. 7b). This basecall group contains a `Move` table (`Events` table in case of Albacore) which is used by *tailfindr* to compute the read-specific translocation rate. The structure of a FAST5 file—raw or basecalled—can be easily explored by opening it in HDFView (https://www.hdfgroup.org/downloads/hdfview/).

Guppy has both CPU and GPU versions. If you have access to an Nvidia GPU, then install and use the GPU version of Guppy, as it is faster to basecall on GPUs compared to CPUs. Here, we will demonstrate basecalling using the CPU version of Guppy (*see* **Note 2** on where to get the latest version of Guppy). Assuming that you have a Quad Core processor (with two threads per processor; eight threads in total) and 16GB of RAM, basecalling can be done by executing the following on the command line:

```
guppy_basecaller \
--config rna_r9.4.1_70bps_hac.cfg \
--input_path \path\to\raw\reads\folder \
--recursive \
--save_path \path\to\save\basecalled\data\to \
--fast5_out \
```

```
--trim_strategy none \
--num_callers 1 \
--cpu_threads_per_caller 8 \
2>&1 | tee logfile.txt
```

Parameter Description

`--config` specify the model configuration to be used during basecalling. In this case, we have chosen the "high accuracy" (hac) RNA model for pore version 9.4.1. The hac models yield more accurate basecalls at the cost of basecalling speed.

Refer to:

- **Note 3** for choosing a faster basecalling model.
- **Note 4** for selecting an appropriate config file for your experiment in case you are not sure.
- **Note 5** if your data is from a legacy RNA kit.

`--input_path` specify the path of the folder containing raw FAST5 files produced by the ONT sequencing platform. When using the example dataset, extract it first, and then specify the path of the extracted directory here.

`--recursive` specifies that the `input_path` directory should be recursively searched to discover all raw FAST5 files within any subfolders.

`--save_path` specify the path of the directory where basecalled files should be stored.

`--fast5_out` specifies that in addition to the FASTQ files, the basecaller should also output FAST5 files. Basecalled files containing FAST5 output is essential for *tailfindr* to calculate the read-specific translocation rate for normalizing the poly(A) tail length.

`--trim_strategy` should be set to `none` so that the basecaller does not trim off the adaptor sequence that was added to the 3′ end of the poly(A) + RNA.

`--num_callers` specifies how many basecallers to use in parallel.

`--cpu_threads_per_caller` specifies how many threads should be used per basecaller. In general, `num_callers * cpu_threads_per_caller` should not exceed the total number of threads available on the machine. Furthermore, there must be at least `4GB + 1GB * num_callers` RAM available. In our case, both these criteria are satisfied for the machine that we are using. For more exhaustive information on these settings, please refer to the document "Guppy basecaller and Guppy basecaller server" (https://community.nanoporetech.com/protocols/Guppy-protocol/v/gpb_2003_v1_revm_14dec2018/guppy-basecaller-and-guppy-basecaller-server) on the Nanopore Community.

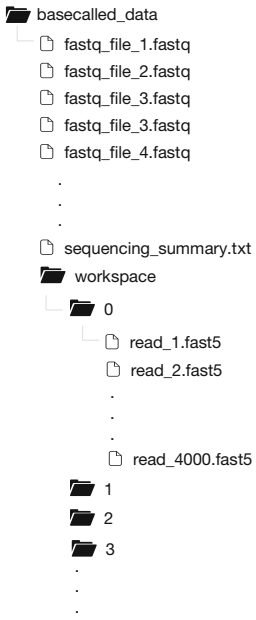


Fig. 8 Structure of the output directory produced by the Guppy basecaller. Each FASTQ file in the output of the basecaller contains sequence and quality scores for 4000 (default) reads. The `sequencing_summary.txt` file contains a summary of useful basecalling information, which is used by tools such as NanoPlot. The `workspace` folder contains numbered subfolders, each of which contain 4000 basecalled FAST5 files, which are used by tools such as *tailfindr*

`2>&1 | tee logfile.txt` specifies that the output, and any errors produced by the command, should be saved in a text file in addition to being displayed in the terminal. It is a good practice to do this for troubleshooting in case of a computer crash, power failure etc.

After successfully running the above, the basecalled FASTQ and FAST5 file can be found in the directory as specified in `save_path`. The structure of this directory is depicted in Fig. 8. Please refer to **Note 6** to find how the structure of this directory changes when multi-fast5 files are basecalled.

4.2.2 Quality Control After Basecalling

Running quality control checks after basecalling is an optional but recommended step as it can reveal important information about the sequencing run such as the length of the reads (Fig. 9a), sequencing performance over time (Fig. 9b), and the quality of the reads

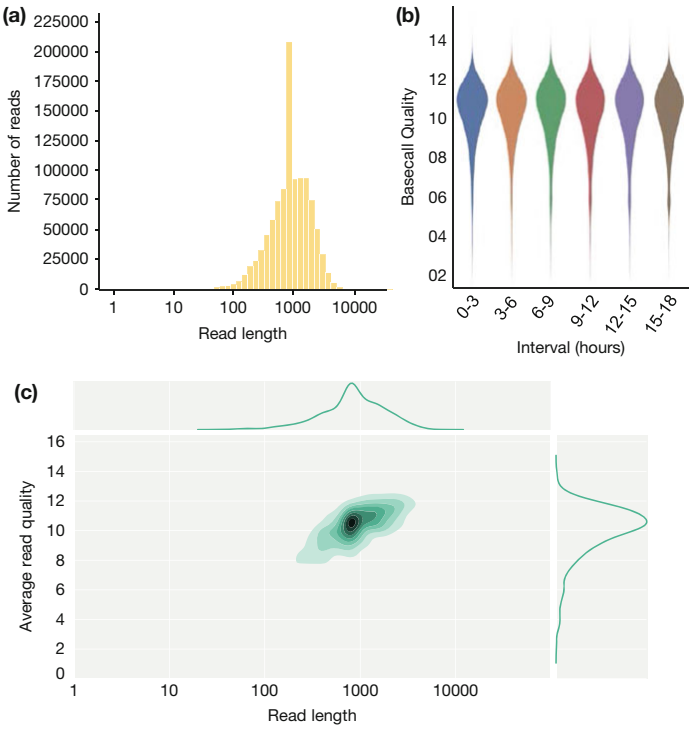


Fig. 9 A subset of figures generated by NanoPlot. (**a**) Read length histogram. This plot can be useful in understanding if RNA degradation significantly affected the sample. This particular histogram was generated for a sequencing run in which Zebrafish RNA was spiked with a synthetic GFP RNA construct of approx. 800 bp in length. The spike in the histogram around 800 represents these GFP reads, and the background represents the read length distribution for the Zebrafish transcriptome. (**b**) Basecall quality vs. time of sequencing. This plot is useful in assessing if the sequencing chemistry—which might degrade over time—is having an adverse effect on the quality of the reads. Ideally, the basecalling quality should not drop dramatically during the sequencing run. (**c**) Read length vs. average read quality plot. It is useful in understanding how the read quality varies over read length. In a good sequencing run, the read quality for the majority of the reads should be around 8–14 for RNA (9–20 for DNA). Higher reads quality are good, and lower read qualities for majority of reads might warrant revisiting the library preparation steps and figuring out what might have gone wrong

(Fig. 9c). There are many tools to perform QC on Nanopore data, but the ones that produce the most informative plots are NanoPlot (https://github.com/wdecoster/NanoPlot) and PycoQC (https://a-slide.github.io/pycoQC/) [23, 24].

Here, we will use NanoPlot to perform quality control checks on the basecalled data. NanoPlot requires only the `sequencing_summary.txt` file produced by Guppy. To run NanoPlot, first

activate a Python 3 environment, and then run the following in the command line:

```
NanoPlot \
--summary \path\to\sequencing_summary.txt \
--outdir \output\path \
--loglength
```

Parameter Description

`--summary`, path of the summary file generated by Guppy.

`--outdir`, path of the directory where NanoPlot output should be saved.

`--loglength`, specifies that the read lengths should scaled logarithmically in the plots.

The output of NanoPlot is an HTML file that can be viewed in any browser of your choice.

4.2.3 *Installing and Running* Tailfindr

We are now ready to estimate poly(A) tail lengths in the basecalled data using *tailfindr*. Please refer to its documentation (https://github.com/adnaniazi/tailfindr) to learn how to install it. After installing *tailfindr*, poly(A) tail lengths can be estimated by using the following commands in R:

```
library(tailfindr)
df <- find_tails(fast5_dir = '/path/to/basecalled_data',
 save_dir = '/path/to/save/folder/',
 csv_filename = 'rna_tails.csv',
 num_cores = 2)
```

tailfindr discovers all FAST5 files recursively within the `fast5_dir`. The resulting CSV file—as specified in the `csv_filename` parameter—is saved in the `save_dir`. `num_cores` specifies the number of physical cores on the machine to be used when running *tailfindr*.

Please refer to:

- **Note 7** if you are running *tailfindr* on MinKNOW Live-basecalled data.
- **Note 8** if you want to generate plots highlighting the poly (A) tail region in the raw current data.
- **Note 9** on how to use *tailfindr* for estimating poly(A)/(T) length in cDNA data.

The output of *tailfindr* is CSV file contain six columns as described in Table 1.

Table 1
Description of columns in the CSV output of *tailfindr*

Column name	Column type	Description
read_id	Character	Read ID as given in the FAST5 file
tail_start	Numeric	Sample index of start site of the tail in raw data
tail_end	Numeric	Sample index of end site of the tail in raw data
samples_per_nt	Numeric	Read-specific translocation rate in terms of samples per nucleotide
tail_length	Numeric	Tail length in nucleotides. It is the difference between tail_end and tail_start divided by samples_per_nt
file_path	Character	Absolute path of the FAST5 file

Now that we have the poly(A) tail length for each read in the CSV file, it is possible to perform quality control checks of this data. For example, a distribution of poly(A) tail lengths can be plotted to see if it aligns with the expected distribution of poly(A) tail lengths. Furthermore, a distribution of the translocation rate `samples_per_nt` can also be plotted. Ideally, this distribution should be unimodal with no skew (*see* **Note 9**).

4.2.4 Concatenate FASTQ Files

During the basecalling step, Guppy produced both FASTQ and FAST5 files. By default, each FASTQ file contains sequences of 4000 reads (*see* Fig. 8). Downstream processing software, such as the mapper *Minimap2* [25], require only a single FASTQ file as input. Therefore, all FASTQ files produced by Guppy should be concatenated. Execute the following script in command line to combine all FASTQ file into one:

```
BASECALLED_DATA_PATH=/directory/containing/basecalled/data
OUTPUT_PATH=/directory/where/concatenated/fastq/is/to/stored
# Do not edit the code below this line
cd $BASECALLED_DATA_PATH
find ${BASECALLED_DATA_PATH} -name '*.fastq' | cat > ${OUT-
PUT_PATH}/filenames.txt
{ xargs cat < ${OUTPUT_PATH}/filenames.txt ; } > ${OUTPUT_-
PATH}/all_reads.fq
```

The above shell script searches `BASECALLED_DATA_PATH` directory for all files with `.fastq` extension and produces the following two files in the `OUTPUT_PATH` directory:

1. `filenames.txt` file that contains the names of all FASTQ files that were found in `BASECALLED_DATA_PATH` directory, and will be concatenated.

2. `all_reads.fq` file that contains the concatenated FASTQ sequences from all the FASTQ files recorded in the `file-names.txt` file.

4.2.5 Alignment of Data to Transcriptome

Although we have estimated poly(A) tail lengths for all reads, we still do not know which transcript each of these reads originated from. To find the transcript identities, the reads must be mapped to the transcriptome of the organism from which the RNA was extracted (please refer to **Note 10** if no reliable transcriptome is present and data should be mapped to a reference genome). The alignment information can then be merged with *tailfindr* output to associate the poly(A) tail length estimations to their respective transcript IDs.

To map the data to the transcriptome, we will use Minimap2 (https://github.com/lh3/minimap2) [25]. Minimap2 needs a single FASTQ file containing all the reads to be aligned. Run the following command in command line to invoke Minimap2:

```
minimap2 \
 -ax map-ont \
 /path/to/reference.fa \
 /path/to/all_reads.fq > /path/to/alignments.sam
 2>&1 | tee logfile.txt
```

Parameter Description

Here is a description of the parameters used in the above command:

`-a` specifies that CIGAR string and output alignments should be produced in the SAM format.

`-x` use predefined settings for mapping. As each of these sequencing technologies differ in their insertion, deletion and error rates, there are a number of presets available in Minimap2 to choose from; `map-ont` is one of them. It specifies that Minimap2 should use alignment parameters fine-tuned for ONT sequencing data. This is because Minimap2 can align reads from Illumina, PacBio, and ONT sequencing.

4.2.6 Annotating Tailfindr Output with Transcript IDs

Now that we have the poly(A) tail length estimates from *tailfindr* in a CSV file, and the alignment information in a SAM file, we are ready to merge them together. This will annotate each read with its corresponding transcript ID. To do this, invoke *tailfindr's* built-in convenience function `annotate_tail()` in R:

```
df_annotate <-
annotate_tails(
sam_file = "/path/to/sam/file.sam",
 tails_csv_file = "/path/to/tails.csv",
```

Table 2
Description of columns added to the *tailfindr* CSV output by merging SAM information

Column name	Column type	Description
transcript_id	Character	Transcript ID from the transcriptome
mapping_quality	Numeric	Mapping quality of the transcript
sam_flag	Numeric	SAM flag

This command will add three more columns to the input CSV file as described in Table 2.

We now have the tail length and the corresponding transcript IDs in the `annotated_tails.csv` file. Thus we have successfully annotated each read with a transcript-isoform ID and a corresponding poly(A) tail length.

4.2.7 What Next?

Now that we have transcript-specific poly(A) tail lengths, we can do a number of things. For example, we can plot the distribution of poly(A) tail length of our dataset. We can also annotate the poly (A) tail length of a transcript with additional features such as gene name, gene length and its function. These steps are beyond the scope of this chapter, however, the reader should note that they can be easily done within R using the *biomaRt* Bioconductor package [26]. With gene name annotations, we can for instance generate a scatter plot of poly(A) tail length vs. gene length to see if there is any interesting relationship between the two. Additionally it is possible to plot poly(A) tail distributions from transcript isoforms of the same genes. Many further possibilities for data analysis exist, and implementation depends on the particular research question. The here described *tailfindr*-based pipeline provides the first step towards exploring these possibilities enabling the study of isoform-specific poly(A) tail-dependent regulation.

5 Conclusion

We have here demonstrated how long-read ONT native RNA sequencing in combination with *tailfindr* can be used for transcriptome-wide isoform-specific poly(A) tail profiling. This method simplifies isoform-specific poly(A) tail measurements and avoids common caveats from short-read based sequencing approaches, namely (1) the possible introduction of amplification artefacts, (2) transcript isoform quantification based on statistical analysis of short reads spanning exon borders, and (3) elaborate and time-consuming sequencing sample preparation.

The portability and low investments for ONT sequencers, coupled with its ability to basecall and analyze sequencing data in real-time, enables anyone to sequence anything, anywhere. Additionally, the ability of direct RNA sequencing to detect any epigenetic modification in native RNA alleviates the need for separate assays for detecting each RNA modification. This enables future studies—both in the field and in a laboratory settings—to assay poly(A) tail length and RNA modifications in a single experiment. Such a transcriptome-wide holistic approach would provide a valuable insight in understanding RNA biology—one long molecule at a time.

6 Notes

1. Reverse-transcribing RNA into an RNA–cDNA duplex is an optional but recommended step. Without performing this step, the throughput will be about 30% lower and basecalling quality scores will also be slightly lower. Most likely this is caused by secondary RNA structure affecting pore translocation, making current signal more variable. Additionally, RNA degradation causes the average read length to be shorter. We recommend that you perform this step unless you have a very good reason not to.
2. We demonstrated how to basecall reads using the latest basecaller at the time of this writing provided by ONT—Guppy v3.2.4. However, it should be noted that the basecalling technology is constantly evolving. Always check ONT's Software Download section (https://community.nanoporetech.com/downloads) to read about the latest version of the basecaller and how to use it, as these might significantly increase basecalling accuracy and thus transcript isoform assignment.
3. If basecalling speed is more important than basecalling accuracy, then use the fast model configuration file `rna_r9.4.1_70bps_fast.cfg`.

   At the time of this writing, the fast models are approximately 5–8 times faster than the high accuracy model. Table 3 shows a comparison between raw read accuracy of fast and high accuracy models.
4. If your experiment uses a pore version other than 9.4.1, then ensure that you specify a configuration file that matches the version of the pore used. You can find a list of all available configuration files for every flow cell and sequencing kit by executing the following command:

Table 3
A comparison of raw read accuracies between fast and high-accuracy basecalling models

Sample type	Model name	Raw read accuracy
DNA	Fast basecalling	92.1%
	High-accuracy basecalling	95.0%
RNA	Fast Basecalling	88.6%
	High-accuracy basecalling	93.9%

```
guppy_basecaller --print_workflows
```

If you are still unsure as to which configuration file to use, then, instead of specifying the configuration file, you can also let Guppy choose the appropriate configuration file for you. In this case, however, you have to specify the `flowcell` and `kit` arguments. Assuming if the flow cell and kit used in the experiment are FLO-MIN106 and SQK-RNA001, respectively, then use the following command in command line to invoke Guppy:

```
guppy_basecaller \
 --flowcell FLO-MIN106 \
 --kit SQK-RNA001 \
 --input_path \path\to\raw\reads\folder \
 --recursive \
 --save_path \path\to\save\basecalled\data\to \
 --fast5_out \
 --trim_strategy none \
 --num_callers 1 \
 --cpu_threads_per_caller 8 \
 2>&1 | tee logfile.txt
```

5. The use of *tailfindr* is compatible with any RNA kit—including legacy kits—as all of these kits sequence both the transcript and the poly(A) tail. Thus, you can use *tailfindr* to find poly(A) tail lengths on any older RNA dataset where the initial aim of the study was something entirely different. For *tailfindr* to work, the only requirement is the availability of FAST5 files—either raw or basecalled; *tailfindr* cannot be used if the only file remaining from past experiments are FASTQ files. We recommend that you always re-basecall the old previously basecalled FAST5 files before using *tailfindr* on it, and specify an appropriate value for `basecall_group` parameter when invoking *tailfindr*.

6. If the raw FAST5 files produced by MinKNOW have only one read per FAST5 file, then reads within the workspace folder are arranged in numbered subfolders such that each folder contains 4000 FAST5 reads, as depicted in Fig. 8. However, if the raw FAST5 files, produced by the sequencer, have multiple reads per FAST5 file, then there are no subfolders within workspace folder, and each basecalled FAST5 file in `workspace` folder will contain multiple reads (default is 4000) inside them.
7. MinKNOW—the data acquisition software used during sequencing on ONT sequencers—can basecall while the raw data is being acquired. This feature is called "MinKNOW Live Basecalling." Currently, *tailfindr* does not support MinKNOW live basecalled data because these FAST5 files do not contain Event/Move table (*see* Fig. 10a). The Event/Move table is required by *tailfindr* to compute a read-specific translocation rate in order to normalize the poly(A) tail length in samples to yield poly(A) tail length in nucleotides.

   To circumvent this problem, please basecall MinKNOW live-basecalled data again using standalone Guppy or Albacore. This will add an additional Basecall group (`Basecall_1D_001`) in the file structure of the FAST5 file (*see* Fig. 10b). When using *tailfindr* on these re-basecalled reads, you must correctly specify the Basecall group containing the Event/Move table. For example, the read shown in Fig. 10, the Event/Move table in the re-basecalled file is in the `Basecall_1D_001` in the FAST5 file structure hierarchy. *Tailfindr*, in the case, should be invoked in R as shown below:

```
df <- find_tails(fast5_dir = '/path/to/basecalled_data',
 save_dir = '/path/to/save/folder/',
 basecall_group = 'Basecall_1D_001',
 csv_filename = 'rna_tails.csv',
 num_cores = 2)
```

   The default value of basecall_group is `Basecall_1D_000`, which in the command above, has been changed to `Basecall_1D_001`.
8. *tailfindr* allows you to generate plots that show the tail location in the raw squiggle (*see* Fig. 11). You can save these plots as interactive .html files by using `'rbokeh'` as the `plotting_library`. You can then interactively zoom in on the tail region in the raw squiggle and see the exact location of the tail. To generate these plots, execute the following command in R:

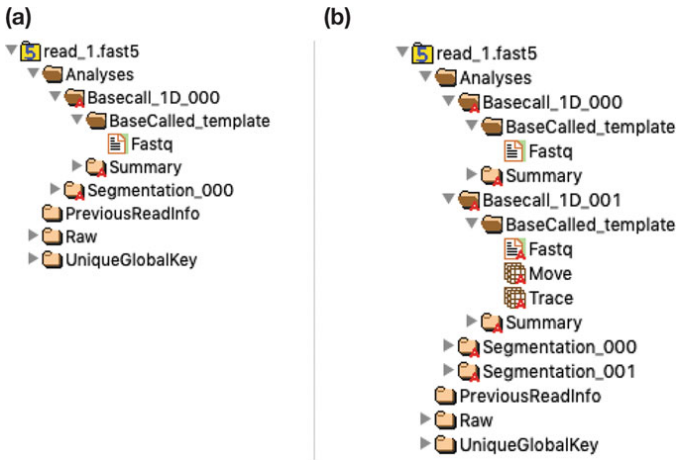


Fig. 10 Hierarchy of contents within basecalled FAST5 files as viewed through the HDFView software. (**a**) Contents of a MinKNOW live Basecalled read. Notice that under the Basecall_1D_000 group, there is no Move table, which is required by *tailfindr* to find the read-specific translocation rate (**b**) Contents of the read shown in (**a**) after it has been basecalled again using standalone Guppy. Notice the addition of Basecall_1D_001 group in the FAST5 file hierarchy, which now contains Move table. *Tailfindr* should now be invoked with basecall_group parameter set to 'Basecall_1D_001' to ensure that it can find the Move table

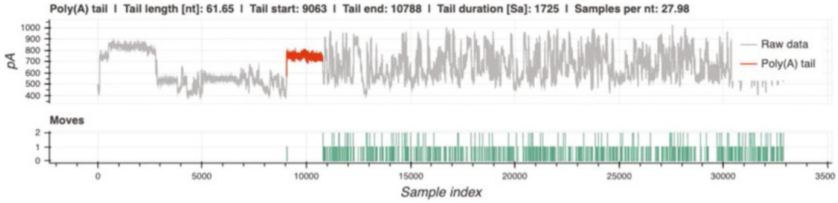


Fig. 11 A plot generated by *tailfindr*. The poly(A) tail is highlighted in red in the current trace. Each spike in the bottom panel shows the locations in the current trace where the basecaller has detected a nucleotide transition. Notice how the poly(A) tail region is devoid of any base transition. This is because the basecaller cannot distinguish when one adenosine base in the poly(A) tail ended and the next one started. It can detect a nucleotide transition only if a more diverse sequence is encountered

```
df <- find_tails(fast5_dir = '/path/to/basecalled_data',
 save_dir = '/path/to/save/folder/',
 csv_filename = 'rna_tails.csv',
 save_plots = TRUE,
 plotting_library = 'rbokeh',
 num_cores = 2)
```

Generating plots can slow down the performance of *tailfindr*. We recommend that you generate these plots only for a small subset of your reads.

9. Although we have demonstrated how to perform poly(A) tail profiling using Nanopore sequencing of native RNA, it is also possible to perform poly(A)/(T) profiling using complementary DNA (cDNA) sequencing data produced by Nanopore sequencing. Sequencing cDNA instead of RNA has many advantages:
   (a) cDNA is more stable compared to RNA which can degrade quickly if not handled very carefully at every step of library preparation protocol,
   (b) cDNA sequencing requires less starting material compared to RNA sequencing,
   (c) cDNA sequencing on Nanopore devices produces ten times more data per flowcell compared to RNA sequencing because of the faster motor protein, and,
   (d) poly(A) tail length estimates in DNA are more robust compared to RNA because the motor protein used in DNA sequencing ratchets the DNA at a more controlled speed compared to the motor protein used in RNA sequencing (*see* Fig. 12).

   For more information on poly(A)/(T) profiling in cDNA, please refer to the *tailfindr* paper [12] and documentation on GitHub.
10. Reads from RNA sequencing can be mapped either to a reference transcriptome, or to a genome. The latter is more cumbersome but could yield the identification of new transcript

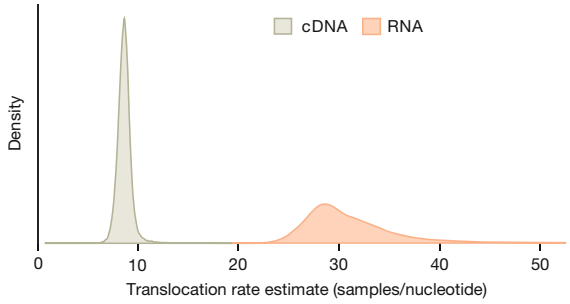


Fig. 12 Comparison of cDNA and RNA translocation rate estimates. RNA translocates at a slower speed compared to DNA. Furthermore, the spread in RNA translocation rate is greater than that of cDNA. This in turn translates to more spread in RNA poly(A) tail lengths compared to cDNA poly(A)/(T) tail lengths

isoforms. This is especially useful if the reference transcriptome is known to be erroneous and is being assessed for the first time by long-read sequencing. For aligning reads to a genome with Minimap2, use the following command:

```
minimap2 \
 -ax splice -uf -k14 \
 /path/to/reference_genome.fa \
 /path/to/all_reads.fq > /path/to/alignments.sam
 2>&1 | tee logfile.txt
```

Parameter Description

Here is a description of additional parameters in the above command:

`-splice` Specifies that spliced alignment should be done.

`-uf` By default, spliced alignment assumes the read orientation relative to the transcript strand is unknown and therefore it tries two rounds of alignment to infer the read orientation. This flag forces Minimap2 to consider only the forward transcript strand during mapping.

`-k14` For noisy Nanopore Direct RNA-seq reads, it is recommended to use a smaller k-mer size for increased sensitivity to the first or the last exons. Default value of k-mer size is 15.

Acknowledgments

Adnan M. Niazi and Maximilian Krause contributed equally to this work.

References

1. Bardwell VJ, Zarkower D, Edmonds M, Wickens M (1990) The enzyme that adds poly(A) to mRNAs is a classical poly(A) polymerase. Mol Cell Biol 10:846–849. https://doi.org/10.1128/mcb.10.2.846
2. Huang Y, Carmichael GG (1996) Role of polyadenylation in nucleocytoplasmic transport of mRNA. Mol Cell Biol 16:1534–1542. https://doi.org/10.1128/mcb.16.4.1534
3. Meyer S, Temme C, Wahle E (2004) Messenger RNA turnover in eukaryotes: pathways and enzymes. Crit Rev Biochem Mol Biol 39:197–216. https://doi.org/10.1080/10409230490513991
4. Beilharz TH, Preiss T (2007) Widespread use of poly(A) tail length control to accentuate expression of the yeast transcriptome. RNA 13:982–997. https://doi.org/10.1261/rna.569407
5. Subtelny AO, Eichhorn SW, Chen GR et al (2014) Poly(A)-tail profiling reveals an embryonic switch in translational control. Nature 508:66–71. https://doi.org/10.1038/nature13007
6. Lima SA, Chipman LB, Nicholson AL et al (2017) Short poly(A) tails are a conserved feature of highly expressed genes. Nat Struct Mol Biol 24:1057–1063. https://doi.org/10.1038/nsmb.3499
7. Chang H, Lim J, Ha M, Kim VN (2014) TAIL-seq: genome-wide determination of poly (A) tail length and 3′ end modifications. Mol Cell 53:1044–1052. https://doi.org/10.1016/j.molcel.2014.02.007

8. Woo YM, Kwak Y, Namkoong S et al (2018) TED-Seq identifies the dynamics of poly (A) length during ER stress. Cell Rep 24:3630–3641.e7. https://doi.org/10.1016/j.celrep.2018.08.084
9. Hite JM, Eckert KA, Cheng KC (1996) Factors affecting fidelity of DNA synthesis during PCR amplification of d(C-A)n•d(G-T)n microsatellite repeats. Nucleic Acids Res 24:2429–2434. https://doi.org/10.1093/nar/24.12.2429
10. Murray EL, Schoenberg DR (2008) Assays for determining poly(A) tail length and the polarity of mRNA decay in mammalian cells. Methods Enzymol 448:483–504. https://doi.org/10.1016/S0076-6879(08)02624-4
11. Hommelsheim CM, Frantzeskakis L, Huang M, Ülker B (2014) PCR amplification of repetitive DNA: a limitation to genome editing technologies and many other applications. Sci Rep 4:5052. https://doi.org/10.1038/srep05052
12. Krause M, Niazi AM, Labun K et al (2019) tailfindr: alignment-free poly(A) length measurement for Oxford Nanopore RNA and DNA sequencing. RNA 25:1229. https://doi.org/10.1261/rna.071332.119
13. Legnini I, Alles J, Karaiskos N et al (2019) FLAM-seq: full-length mRNA sequencing reveals principles of poly(A) tail length control. Nat Methods 16:879–886. https://doi.org/10.1038/s41592-019-0503-y
14. Workman RE, Tang AD, Tang PS et al (2019) Nanopore native RNA sequencing of a human poly(A) transcriptome. Nat Methods 16:1297. https://doi.org/10.1038/s41592-019-0617-2
15. Byrne A, Cole C, Volden R, Vollmers C (2019) Realizing the potential of full-length transcriptome sequencing. Philos Trans R Soc Lond Ser B Biol Sci 374:20190097. https://doi.org/10.1098/rstb.2019.0097
16. Garalde DR, Snell EA, Jachimowicz D et al (2018) Highly parallel direct RNA sequencing on an array of nanopores. Nat Methods 15:201–206. https://doi.org/10.1038/nmeth.4577
17. Liu H, Begik O, Lucas MC et al (2019) Accurate detection of m6A RNA modifications in native RNA sequences. Nat Commun 10:4079
18. Butler TZ, Pavlenok M, Derrington IM et al (2008) Single-molecule DNA detection with an engineered MspA protein nanopore. Proc Natl Acad Sci U S A 105:20647–20652. https://doi.org/10.1073/pnas.0807514106
19. Cherf GM, Lieberman KR, Rashid H et al (2012) Automated forward and reverse ratcheting of DNA in a nanopore at 5-Å precision. Nat Biotechnol 30:344–348. https://doi.org/10.1038/nbt.2147
20. Rang FJ, Kloosterman WP, de Ridder J (2018) From squiggle to basepair: computational approaches for improving nanopore sequencing read accuracy. Genome Biol 19:90. https://doi.org/10.1186/s13059-018-1462-9
21. Jain M, Fiddes IT, Miga KH et al (2015) Improved data analysis for the MinION nanopore sequencer. Nat Methods 12:351–356. https://doi.org/10.1038/nmeth.3290
22. Wick RR, Judd LM, Holt KE (2019) Performance of neural network basecalling tools for Oxford Nanopore sequencing. Genome Biol 20:129. https://doi.org/10.1186/s13059-019-1727-y
23. De Coster W, D'Hert S, Schultz DT et al (2018) NanoPack: visualizing and processing long-read sequencing data. Bioinformatics 34:2666–2669. https://doi.org/10.1093/bioinformatics/bty149
24. Leger A, Leonardi T (2019) pycoQC, interactive quality control for Oxford Nanopore Sequencing. J Open Source Softw 4:1236. https://doi.org/10.21105/joss.01236
25. Li H (2018) Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics 34:3094–3100. https://doi.org/10.1093/bioinformatics/bty191
26. Durinck S, Spellman PT, Birney E, Huber W (2009) Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. Nat Protoc 4:1184–1191. https://doi.org/10.1038/nprot.2009.97

Graphic design: Communication Division, UiB / Print: Skipnes Kommunikasjon AS

uib.no

ISBN: 9788230849484 (print)
9788230858554 (PDF)