Tailor: a computational framework for detecting non-templated tailing of small silencing RNAs

Min-Te Chou  
*National Chiao Tung University*

Bo W. Han  
*University of Massachusetts Medical School*

Chiung-Po Hsiao  
*National Chiao Tung University*

*See next page for additional authors*

Follow this and additional works at: [http://escholarship.umassmed.edu/gsbs_sp](http://escholarship.umassmed.edu/gsbs_sp)

Part of the [Biochemistry Commons](http://escholarship.umassmed.edu/gsbs_sp), [Bioinformatics Commons](http://escholarship.umassmed.edu/gsbs_sp), and the [Computational Biology Commons](http://escholarship.umassmed.edu/gsbs_sp)

**Repository Citation**

Chou, Min-Te; Han, Bo W.; Hsiao, Chiung-Po; Zamore, Phillip D.; Weng, Zhiping; and Hung, Jui-Hung, “Tailor: a computational framework for detecting non-templated tailing of small silencing RNAs” (2015). GSBS Student Publications. 1882.  
[http://escholarship.umassmed.edu/gsbs_sp/1882](http://escholarship.umassmed.edu/gsbs_sp/1882)

This material is brought to you by eScholarship@UMMS. It has been accepted for inclusion in GSBS Student Publications by an authorized administrator of eScholarship@UMMS. For more information, please contact Lisa.Palmer@umassmed.edu.
Tailor: a computational framework for detecting non-templated tailing of small silencing RNAs

Authors
Min-Te Chou, Bo W. Han, Chiung-Po Hsiao, Phillip D. Zamore, Zhiping Weng, and Jui-Hung Hung

Comments
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by-nc/4.0/), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited.

Creative Commons License
This work is licensed under a Creative Commons Attribution-Noncommercial 4.0 License

Rights and Permissions
Citation: Nucleic Acids Res. 2015 May 24. pii: gkv537. Link to article on publisher's site

This article is available at eScholarship@UMMS: http://escholarship.umassmed.edu/gsbs_sp/1882
Tailor: a computational framework for detecting non-templated tailing of small silencing RNAs

Min-Te Chou1,†, Bo W. Han2,†, Ching-Po Hsiao1, Phillip D. Zamore2, Zhiping Weng3 and Jui-Hung Hung1,*

1Institute of Bioinformatics and Systems Biology and Department of Biomedical Science and Technology, National Chiao Tung University, HsinChu, Taiwan, 300, Republic of China, 2RNA Therapeutics Institute, Howard Hughes Medical Institute, and Department of Biochemistry and Molecular Pharmacology, University of Massachusetts Medical School, Worcester, MA 01605, USA and 3Program in Bioinformatics and Integrative Biology, University of Massachusetts Medical School, Worcester, MA 01605, USA

Received March 31, 2015; Revised April 24, 2015; Accepted May 10, 2015

ABSTRACT

Small silencing RNAs, including microRNAs, endogenous small interfering RNAs (endo-siRNAs) and Piwi-interacting RNAs (piRNAs), have been shown to play important roles in fine-tuning gene expression, defending virus and controlling transposons. Loss of small silencing RNAs or components in their pathways often leads to severe developmental defects, including lethality and sterility. Recently, non-templated addition of nucleotides to the 3′ end, namely tailing, was found to associate with the processing and stability of small silencing RNAs. Next Generation Sequencing has made it possible to detect such modifications at nucleotide resolution in an unprecedented throughput. Unfortunately, detecting such events from millions of short reads confounded by sequencing errors and RNA editing is still a tricky problem. Here, we developed a computational framework, Tailor, driven by an efficient and accurate aligner specifically designed for capturing the tailing events directly from the alignments without extensive post-processing. The performance of Tailor was fully tested and compared favorably with other general-purpose aligners using both simulated and real datasets for tailing analysis. Moreover, to show the broad utility of Tailor, we used Tailor to reanalyze published datasets and revealed novel findings worth further experimental validation. The source code and the executable binaries are freely available at https://github.com/jhhuang/Tailor.

INTRODUCTION

Over the past decade, small silencing RNAs, including microRNAs (miRNAs), endogenous small silencing RNAs (endo-siRNAs) and Piwi-interacting RNAs (piRNAs) have been shown to play indispensable roles in regulating gene expression, protecting against viral infection and preventing mobilization of transposable elements (1–4). Small silencing RNAs exert their silencing function by associating with Argonaute proteins to form RNA-induced silencing complex (RISC), which uses the small RNA guide to bind its regulatory targets and reduce gene expression. Although the studies on the biogenesis of all small silencing RNAs have made enormous progress in the past decade, the factors controlling their stability and degradation remain elusive.

Recent studies have suggested that non-templated addition to the 3′ end of small silencing RNAs, namely tailing, could play essential roles in this regard. Non-templated addition of mono- or oligo-uridylation of the pre-microRNAs (pre-miRNAs) regulates miRNA processing by either preventing or promoting Dicer cleavage in insects (5–7). The 3′ mono- or oligouridylation on small interfering RNAs in Caenorhabditis elegans is associated with negative regulation (8). Am cess et al. have demonstrated that highly conserved entropic targets trigger the tailing of mRNAs and eventually lead to their degradation in insects and mammals (9,10); a similar mechanism has been found on some endo-siRNAs as well (11). Identification of tailing events not only suggests the co-evolution of all small silencing RNAs and their targets, but also sheds light on the mechanism of their maturation and degradation.

Despite the fact that Next Generation Sequencing (NGS) has greatly facilitated the understanding of RNA tailing, computational detection of non-templated nucleotides from millions of sequencing reads is challenging. The Ketting group used MEGA BLAST to align RNA sequences
to the genome and relied on post-processing the reported mismatches to gain insights into tailing (8). However, as a heuristic algorithm, BLAST is not guaranteed to nd all the target events (12,13) and it is signi cantly slower than the NGS aligners, like MAQ (14), BWA (15), Bowtie (16) and SOAP 2 (17). The Chen group used an accurate method that iterates between Bowtie alignment and 3‘ clipping of unaligned reads (18) to nd all the perfect alignment events of trimmed reads. A similar approach has been used for removing erroneous bases at 3‘ end to increase the sensitivity of detecting miRNAs (19). Let alone that this method inevitably multiplexes the running time by them all length of tails, extra computational works are still needed to retrieve the identity of each trimmed tail. The study by Amaral et al. used a specialized suf x tree data structure to ef ciently nd all thetails without sacri cing the accuracy. However, due to the high memory footprint of the suf x tree data structure, which is about 16 to 20× of the genome size, the read mapping has to be performed for each chrom osome separately (9,20). Extra processing is still required to nd the alignment events from all chrom osomes.

Moreover, the task becomes even trickier when technical and biological confounding factors are taken into account for better capturing the true tailing events. For example, it is known that reads from T.therium H 5Seq and G enome con- analyz er platform s have preferential A-to-C conversions (21,22) and a high error rate at the 3‘ end of reads, which frequently leads to uncod ed bases, i.e. B-tails (23,24). In addition to these technical artifacts endued by the sequencers, R NA editing is another common post-transcriptional modici-ation in small silencing RNA biology that could perplex the tools with erroneous alignment. There are two major types of R NA editing in mammals, adenosine to inosine (A-to-I) and cytidine to uridine (C-to-U) editing. The major enzymes that catalyze adenosine to inosine are the adenosine deaminases acting on R NA (ADARs), whose main substrates are R NAs with double-stranded structures (25–27). Since most small silencing R NAs are originated from structural R NAs, they are all likely targets of A-to-I editing (28–30). Recent studies have shown that A-to-I editing can occur on the seed region of the R NAs with high con-occurrence rate (up to 80% in some cases) and have a direct impact on the selection of their regulatory targets (31,32). Those unaligned bases degenerate the sensitivity and accuracy of short read alignment and have a negative eect on the detection of tailing.

Most of the current methods simply ignore those confounding factors and rely on adapting existing, less specialized tools with extensive post-processing and as a consequence the perform ance, usefulness and application of tailing analysis is seriously compromised. A fast, accurate and straightforward approach to study tailing is still in need. To ease the cost of performing tailing analysis with dramatic increasing sequencing throughput, we here introduce Tailor—a new algorithm that pre-processes and maps sequences to a reference, distinguishing tails from mismatches or bad alignment events with a novel algorithm and reports both perfect and trimmed alignment events without loss of information. Tailor is capable of analyzing the non-tem plates tailing forms R NA and other types of small R NAs and produce publication-quality summary graphs. In addition, to better demonstrate the utility of Tailor, we metaanalyzed published datasets with Tailor and unearthed several interesting observations (see Applications—case studies in Results). Although the ndings still require thorough experimental validation, it is clear that Tailor would help expand the scope of the study of small silencing R NAs.

MATERIALS AND METHODS

Datasets

Therium H 5Seq read data of small R NAs from Drosophila melanogaster hen1 (SRR029608, SRR029633), D. anotox ring hen1 (SRR363984–5), A. thaliana hen1 and hen2 (SRR010683) and Ago2 associated small R NAs in cytoplasmic (SRR529097) and nuclear fraction (SRR529100) of Helia were obtained from NCBI Sequence Read Archive. The length distribution of the simulated confounded reads was from the D. melanogaster Ago3 associated small R NAs extracted from ovaries (SRR916073). In-house program was used to trim the 3‘ adaptors and map the reads with low quality. Randomly distributed reads from fruit- fly genome was generated by Artic RNA FastG enerator (33).

Ten millions reads were randomly chosen using seqtk (github.com/LH3/seqtk.git) with options ‘sam ple-s 1000000’. To remove multiple mappings reads in some simulation datasets, we used Bowtie iteratively before and after the tail appending and seed mutation to assure each read has only one occurrence in the reference.

Rationale

The principle of detecting non-tem plates bases at the 3‘ end of reads is basically to nd the longest common pre x (LCP) between the read and each of the suf xes of the reference and then report the seeds and the read as a tail. Given a read R (M base pairs [bp] long) and all the suf xes (Si) of a reference sequence G (N bp long), one can nd the LCP between R and Si by nding the longest consecutive matches from the rst base to the last. Since there are totally N suf xes of G, a trivial solution needs at most MN times of comparison to nd the LCP of R and G; however, the perform ance is unacceptably slow when G is as large as a human genome. Using index structures, such as the suf x tree or suf x array, nding LCPs between the NGS reads and the reference can be solved much more ef ciently (34). Recently, the Full-text Index in M inute space (FM -index) derived from the Burrows-Wheeler transform (BW T) (35–37) is widely used in many NGS applications (15–17). The FM -index is both time and space ef cient and can be built from a suf x array and requires only 3 to 4 bits per base to store the index. A more detailed introduction of building the FM -index of long biological sequences is given in the Supplementary Materials. However, since the FM -index is originally designed for matching all bases of a read to a sub-string of the reference, it cannot be used directly for nding tails. One straightforward solution is to align reads without those non-tem plates bases by repeatedly removing one last base in each round of the alignment process until at least one perfect match is found (18), but the approach scales the speed greatly and requires extensive post-processing. To bene t from the space and time ef ciency of the FM -index,
we further modified our matching procedure and adapted the error-tolerant strategy proposed by Langmead et al. (16) to devise an FM-index-based tail detection algorithm, Tailor, which is specialized in capturing the non-templated bases at the 3’ end of reads with confounding factors, such as sequencing errors and RNA editing.

Read mapping algorithm of Tailor

The system flow of the Tailor algorithm is outlined in Figure 1. Since searching within the FM-index initiates from the 3’ end of the query string (i.e. the read) (36), where the non-templated nucleotides append, Tailor first makes the reverse-complement of the query sequence so that searching starts from the original 5’ end to avoid excessive exhaustive search at the early stage. To do so, the reference should be reversed complemented as well, and the coordinates of each alignment should be calculated accordingly. To avoid searching against both strands simultaneously and improve the speed, Tailor concatenates the plus and minus strands of the reference and constructs one index instead of two (Figure 1A and Supplementary Materials). Tailor also stores a part of the suffix array similar to other FM-index based aligners (16,38–40) to achieve fast calculation of the next shift for getting the coordinate of each occurrence. Any alignment whose prefix matching portion exceeds the boundary of the m-appended chromosome is ignored. The searching continues until either an alignment is found or all the characters of the query to the reference (i.e. the perfect matching) or no more bases can be matched (i.e. the prefix and suffix matching). In the latter case, Tailor backtracks to the previous matched position and exhaustively enumerates all the possible prefix and suffix matches. The unmapped part remains in the query pool as a tail (Figure 1B).

Clearly, this strategy is vulnerable to confounding factors, since the first mismatch encountered directed directly denotes the tail end as the tail, which can be very misleading. To accommodate possible sequencing errors or RNA editing events in a read, we devised specialized selection rules as depicted in Figure 2. For each read, the mismatch (S = 18 by default) bases at its 5’ portion is denoted as the seed (Figure 2A). Given the fact that sequencing errors tend to occur at the 3’ end (23,24) and RNA editing events in mRNAs are enriched at the other end (i.e. the seed region) (30–32), the selection rules behave according to whether or not the first mismatch appears in the seed (Figure 2B).

If the first mismatch is not in the seed region, it is regarded as either the base of the tail or a sequencing error. In the case that the mismatch is at the last base, it is directly deemed as a valid tail (Case 2 in Figure 2B). If the tail is longer than 1 nucleotide (nt), it will be further scanned to make sure that the sequence of the tail consists of multiple non-templated nucleotides (Case 3). If the tail is only one nucleotide different from the reference, no tail but a mismatch will be reported (Case 4). Note that in order to differentiate tails from sequencing error, a string step based on the quality of sequence is necessary to avoid type I error and has been included in Tailor’s pipeline (see below; Analysis pipeline). Our current algorithm cannot differentiate the circumstances that the tail sequence is identical to the genome sequence. This problem is unlikely to be solved computationally and experimental solutions are expected to be more effective (e.g. using mutant with a defective tailing pathway).

On the other hand, if the first mismatch is in the seed, then RNA editing events occur frequently, the backtracking search will be inhibited and looks for an LCP started from the succeeding base after the mismatch. If no mismatch is found in the inhibited search, no tail but a mismatch is reported (Case 5). If a mismatch is found outside the seed, the read is reported as a tail (Case 6 and 7); otherwise, the read is dropped (Case 8). Note that the scenario that Case 4 with another mismatch in the seed is not allowed (i.e. two mismatches as in Case 8), since in principle we want to endow Tailor an error tolerance strategy consistent to that of conventional approaches under the one mismatch setting (e.g. –v1 in Bowtie).

In silico validation

We performed the core of the Tailor aligner using C++ with built-in support for multithreading. Since Tailor concatenates both strands of the chromosome es into one long reference, whose length could exceed the number represented by 32 bits, we have to use 64 bits to store the indexes in all the relevant data structures, which require about 2X memory footprint than that of other FM-index based aligners. To backward compatibility with the algorithm introduced in Amanzadeh et al. (9), which allow only case 1, 2 and 3 in Figure 2, an option (−v) is needed to turn on the detection of other cases. Tailor has a similar command line interface like other NG S aligners and reports alignment in the SAM (41) format. A tail is described as ‘soft-clipping’ in CIGAR and the sequences are reported under ‘TLZ’ in the optional fields. Mismatches, if allowed (−v), will be reported in the ‘MD’ tag (see Supplementary Materials for more details). Tailor is freely available on GitHub (http://jhung.github.io/Tailor/) under GNU General Public License 2. All the scripts used in preparing this manuscript have also been included in the main G or u repository. The tailing pipelines were in silico enated in shell scripting language and R.

Test environment and software

All software tests were performed in the x86_64 CentOS environment with 24 cores and 48G of RAM. The Bowtie software used in this study is version 1.0.0, 64-bit. The version of BWA used is 0.7.5a-r405. The version of Tailor used is 1.0.0. All command lines for all the tests are listed in the Supplementary Materials.

RESULTS

Perform anence without confounding factors

To begin with, we ignored confounding factors in the following tests to compare with conventional approaches irst. To assess the aligning speed directly, we incrementally generated 10 m millions of perfectly genome-matched reads from the D. melanogaster genome (simulated tail-free dataset) (33) and randomly appended 1–4 genome-unique nucleotides to the 3’ ends (simulated tail

Downloaded from http://nar.oxfordjournals.org/ at University of Massachusetts Medical School on August 16, 2015
To test the performance of Tailor, we compared it with two popular BW T aligners, Bowtie and BWA, by applying them on simulated small RNA datasets (Figure 3A). For the simulated tail-free dataset, Tailor outperformed Bowtie and BWA in all thread settings using 2, 4, 8, 12, and 24 threads (Figure 3A, top). All the running time plotted was the average of the actual running time of repeated experiments. But for the simulated tailed dataset, Bowtie ran slightly faster than Tailor possibly due to the fact that it reported no alignment and did not perform any disk writing (Figure 3A, bottom).

We also performed the speed test with real all small RNA sequencing data from hen1+/− and hen1−/− libraries (Supplementary Figure S1). Please note that Bowtie and BWA in the speed test setting here were not capable of detecting non-templated tails. These tests were just used to compare their execution speed but not functionality.

To prove the accuracy of Tailor when confounding factors were not considered, we then used either Tailor or the Chen method to identify the non-templated tailing events (18). To achieve maximum speed of the Chen method to our best knowledge, we used the `-3k` option of Bowtie to clip k bases off from the 3 end of each read. This strategy avoided calling secondary programs and ensured that all input data were treated as one-pass computational work was done earlier than Bowtie mapping. We started the alignment by setting k to 0. After the initial mapping, the unaligned reads were realigned with an incremented k (k = 1). This process was repeated four times. In the last iteration, all nucleotides were trimmed off from the 3 end (k = 4) and all the tailed reads should have been mapped at this point. In the simulation test, this method finished in 67±1 s with Bowtie being called 16 times (k = 4).
Figure 2. Error tolerance filtering rules. (A) Reads would have to be reverse-complemented before searching. The corresponding seed region is highlighted in green. (B) Eight rules for determining tails. See the main text for more details.

**Performance with error tolerance**

It is arguable that some NG S aligners that support local alignment, such as Bowtie2 (38) and BWA, can recover those tails with error tolerance. We simulated two datasets (one normal, one mutated, see below) whose distribution of read length follows that of the real small RNA sequencing datasets (43) (see Data sets in 'Materials and Methods' section; and also Supplementary Figure S2). For the normal dataset, two million reads were randomly sampled from the reference genome. We intentionally kept reads having just one unique occurrence in the genome and then appended a 1–4 nt non-template added tail on each read. For the mutated dataset, a similar procedure was used to generate another two million reads, but one additional step was added: we introduced one substitution in the nucleotides 2–8 of each read to simulate an RNA editing event as suggested by Vesely et al. (32). Again, this substitution was picked carefully to have only one occurrence in the genome with exactly one mismatch. The simulation guaranteed that there existed only one best alignment to the reference for each read in both datasets (see Data sets in 'Materials and Methods' section).

Then we examined the mapability of these datasets by Tailor (with −v option), Bowtie2 and BWA (see Figure 3C). Tailor clearly reported more unique mapping reads than others especially in the mutated datasets. When we looked closer to those reads that were mapped to multiple positions, we found Bowtie2 and BWA were more likely to align the tails to the reference than Tailor and create multiple alternative alignments. Note that the seed region setting was used to allow all three tools for the alignment (S = 20 and −v in Tailor and the equivalences in Bowtie2 and BWA; mismatches in the seed region were allowed) and all tools should try to align the first 20 nt of each read to the genome, but Bowtie2 and BWA still generated suboptimal alignments. Although the execution time of the three aligners with the error tolerance setting is depicted in Supplementary Figure S3. The complete commands for running all the tests are listed in Supplementary Materials.

We further checked whether the alignments and the tails were correctly reported. As shown in Figure 3D, Tailor was the only tool that gave satisfactory results reporting correct alignments and tails in the mutated dataset. There was no information in the output of BWA to recover the tails, and since most of the reads were aligned to multiple locations, extensive post-processing would be needed for extracting the tails. The simulation clearly shows that Tailor is the only practical solution for doing tailing analysis with confounding factors.

**Analysis pipeline**

In order to provide a thorough and straightforward tailing analysis of deep sequencing libraries to the scientific...
Figure 3. Speed comparison between Tailor and other software. (A) Speed comparison between Tailor, BWA and Bowtie using simulated 18–23 nt small RNA with (top) or without (bottom) non-templated tails. Tailor ran with the default setting, which allows no mismatches in the middle of the query. Tailed alignments were reported if perfect match could not be found. Bowtie ran with ‘−a−best−strata−v0’ setting to allow no mismatches while reporting all best alignments. BWA ran with the default setting. Five different CPU settings were used and the running time was plotted. Three replicates were performed. (B) Speed comparison between Tailor, BWA and Bowtie (commands can be found in Supplementary Materials) using published small RNA Illumina NGS libraries from hen1+/− and hen1−/− mutants in fruit fly and zebra fish. Same settings were used as in (A). (C) The mappability of the normal (N) and mutated (M) datasets aligned by Tailor, Bowtie2 (with local alignment) and BWA. Multiple mapping was deemed as misalignments since each read was guaranteed to have only one occurrence in the reference. (D) The unique mapping reads shown in (C) were further examined to make sure they were aligned correctly and with proper tails reported (correct tails). Unique mapping reads that didn’t have correct alignment or tails were categorized another group (wrong tails/wrong alignment). The unmappable and multiple mapping reads were grouped together (undetermined or unmappable).

We developed the interface of Tailor to take FastQ files as input and produce publication-ready figures. The flowchart of the pipeline is summarized in Supplementary Figure S4A. In brief, the input reads, with barcodes and adaptors removed, are subjected to a high-quality trimming step based on a PHRED score threshold provided by the user (e.g. to get rid of B-tails). The pipeline then applies Tailor to align the high-quality reads to the reference. The information on the length and identity of tails are then retrieved from the SAM format output and summarized to a tabular text file. Additionally, the alignments are assigned to different genomic features (miRNAs, exons, introns, etc.) using BEDTools (44). Tails from different categories are summarized. Publication quality figures depicting the length distribution are drawn using R package ggplot2 (23) (Supplementary Figure S4B). The pipeline also offers microRNA specific analysis. Balloon plots describing the 5 and 3 relative positions and the tails length are provided for a comprehensive overview (Supplementary Figure S4C).
Applications—case studies

To prove the utility of Tailor, we applied Tailor to re-analyze several publicly available small RNA sequencing datasets and revealed new facts about the data that has not been reported yet. In plants, HUA ENHANCER 1 (HEN1) methylates both mRNA and siRNA at their 3' ends to protect them from non-templated uridylation catalyzed by HEN1 SUPPRESSOR 1 (HESO1), a terminal nucleotidyltransferase that favors uridine as substrate (18,45). We applied Tailor on small RNA sequencing libraries from WT, hen1−/− and hen1−/−;heso1−/− cells of Arabidopsis and the results showed that siRNAs were subjected to both non-templated uridylation and cytosylation without HEN1 while mRNAs were mainly subjected to uridylation. Furthermore, the loss of HESO1 only reduced the uridylation but not cytosylation of siRNAs, suggesting the existence of additional nucleotidyltransferase that prefers cytosine as substrates (Figure 4A).

We then applied Tailor to two NGS libraries that cloned Ago2 associated small RNA from nuclear and cytoplasmic fraction of HeLa cells respectively (46). Since RNAs were cloned using poly-A polymerase instead of 3' adaptor ligation in the library preparation, A-tails were unable to be re-
covered com putationally. Although most mRNAs showed very similar length distribution and tailing frequency between these two sam ples, one mRNA, mR-15a, exhibited a distinct pattern. In cytoplasm, mR-15a was only 21 nt long and had modest U tailing for its 22-mers isoform. Surprisingly, in the nu cleus fraction, mR-15a peaked at 22 nt and showed strong U tailing (Figure 4B). In addition, mR-15b, which shares its seed sequence with mR-15a and only has one nucleotide different from mR-15a in the first 19 nt of its mature sequence, did not exhibit obvious variation between the two sam ples. This suggests that, either 9–12 nt, also known as the ‘central site’ or the 3 end of guide mRNA play an important role in tailing regulation.

Finally, we applied Tailor to study the possible relationship between RNA editing and tailing in mRNAs. The mRNA libraries were constructed from the whole brain tis sue cells dissected from Adar2^-/- and wild-type mice (32). A da2^-/- is known for its strongest effects on mRNA abundance and editing among the three isoforms of ADARs (47). One of the highly expressed ADAR substrates, miR-379, was shown to be directly edited at the nucleotide within the seed region and about half of the mature miR-379 were edited by ADAR2 (32). As expected, the edited form of miR-379 (i.e., miR-379-5G) was greatly reduced in Adar2^-/- mice. Surprisingly, we found that the normal miR-379 has much more tailing than miR-379-5G (see Figure 4C). Mono-A and poly-A tails (the bluish portion) were depleted in miR-379-5G, which makes the probability that ADARs and the A-to-I editing could affect the affinity between the mRNAs and the unknown enzyme responsible for adenylylating the 3 end. Since the proportion of different types of tails was unchanged upon Adar2 knockout, the tailing machinery is less likely modulated by ADAR2 directly but by the subsequent factors after editing in the seed, such as differential targeting, RNA stability change or miRNA Argonaute sorting (48).

DISCUSSION

Tailing is a molecular phenomenon that associates with the function, processing and stability of many small RNAs. Computational identification of the tailed sequences from the millions of NGS reads has been proven to be challenging and time-consuming. We herein present a tailing analysis from a work, Tailor, which aligns reads to the reference genome, reports tailing events simultaneously and visualizes analysis results. We assessed the accuracy of Tailor by comparing it with the Chen method with simulated reads and found they generated exactly the same results. While Tailor only used a third of the time to align and provided more information compared to the alternative.

When confounding factor was ignored, Tailor was not slower than other well-known methods in our tests. We demonstrated that Tailor executed in a speed that was very competitive to, if not better than, Bowtie and BWA, while providing more functionalities for detecting tailing events. When confounding factors were presented in the reads, it was argued that advanced NGS aligners that associate the local alignment (e.g., Bowtie2) could be competent in analyzing tails, but we tested them with simulated reads and showed that Tailor performed significantly better in both accuracy and efficiency.

Tailor’s shell-based command takes raw reads as input and produces comprehensive tailing analysis results and publication-quality graphs. We reproduced known conclusions drawn from the published tailing study by the pipeline with little extra scripting and post-processing. We also applied the pipeline to other datasets and shed light on other possibilities of the functional roles of tailing, such as involving in RNA processing, transport, decay and storage by interacting with other RNA binding proteins.

Our aim is to design Tailor to reduce the cost of doing tailing analysis and benchmark or even replace the conventional computational procedure in analyzing all short non-coding RNAs. We expect that Tailor could be applied to a broader scope and subsequently facilitate the understanding of biological processes related to tailing.

AVAILABILITY

Source code as an Open Source project: http://jhung.github.io/Tailor.

SUPPLEMENTARY DATA

Supplementary Data are available at NAR Online.

ACKNOWLEDGEMENT

We thank them and members of the Hwang, Wang and Zamore laboratories for helpful discussions and critical testing.

FUNDING


REFERENCES
