Title
Read Simulator for Single Cell RNA Sequencing

Author
Li, Wenshan

Publication Date
2019

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Read Simulator
for Single Cell RNA Sequencing

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Computer Science

by

Wenshan Li

2019

ABSTRACT OF THE THESIS

Read Simulator
for Single Cell RNA Sequencing

by

Wenshan Li
Master of Science in Computer Science
University of California, Los Angeles, 2019
Professor Wei Wang, Chair

Techniques for single-cell RNA sequencing (scRNA-seq) has enabled unprecedented insights into gene expressions in cell level. Drop-seq is one of the prominent scRNA-seq protocols, and there has been a rapid growth in related analysis tools for Drop-seq data. These methods are tested either using spike-in experiments or on simulation datasets as the real word gene differential expressions are usually unknown. Since spike-in experiments are expensive and time consuming, simulated datasets have become a reasonable alternative method. However, current RNA-seq simulators mostly target at bulk RNA sequencing, which provokes the need of a scRNA-seq simulator for the Drop-seq technology.

In this paper, we present Dropify, an end-to-end framework to simulate the sequencing reads of a Drop-seq experiment. Dropify is able to simulate large amout of Drop-seq reads according to the user's experimental setting. Data generated by Dropify is a reasonable approximation to real Drop-seq data.

The thesis of Wenshan Li is approved.

Sriram Sankararaman

Jason Ernst

Wei Wang, Committee Chair

University of California, Los Angeles

2019

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

Single-cell RNA sequencing (scRNA-seq) has been gaining momentum due to its ability to measure gene expression levels in an isolated cell and analyze differential gene expression across cells. Prior to the development of scRNA-seq, populations of cells were typically sequenced in bulk. Bulk RNA sequencing (bulk RNA-seq) measures the mean expression level of genes but are limited when it comes to understanding heterogeneous populations of cells [WNK13, ZPO17, KKS15]. There are many scenarios where scRNA-seq techniques are preferred over bulk RNA-seq. To name a few applications, scRNA-seq is used to analyze cell development for tumor progression and early embryonic stages [WNK13, KKS15, TBW09]. The technology also helps gain insight on how specific alleles affect the expression of genes, which further leads to gene-regulatory networks and pathway analyses [KKS15, HLB18].

RNA sequencing methods can be divided into two categories: full-length methods such as Smart-seq [RLW12], and tag-based methods such as STRT [IKM11], CEL-seq [HWS12], inDrop [KMA15], MARS-seq [JKK14], SCRB-seq [SCS14] and Drop-seq [MBS15]. For tag-based protocols, mRNAs are captured by a 30-bp oligo dT sequence present at the end of all primer beads. This is also known as the priming process [ZVP17]. To be more specific, primer beads have an affinity for any region with consecutive adenine nucleotides, including the poly-A tail at the 3' end of a transcript and sequences within the transcript. As a result of the priming process, tag-based methods tend to have more reads sequenced closer to the tagged end of a transcript (i.e., the poly-A tail at the 3' end) [IZJ14]. In the sequencing process, tag-based methods focus on a particular region of the transcript, and thus only a portion of each RNA molecule is sequenced [KDA14]. For full length methods, however, the molecule is divided and sequenced in multiple fragments. As a result, full-length methods

aim to capture a more uniform read coverage across a transcript. While tag-based methods lead to a non-uniform read coverage, they have a deciding advantage compared to full-length methods. Tag-based methods eliminate amplification bias by identifying the specific transcript molecule that a read comes from, thus providing a more accurate gene quantification. Due to this accuracy, tag-based methods are more preferable than full-length methods under many circumstances, especially when studying gene expression between different cells [KDA14].

Drop-seq is one of the widely-used tag-based scRNA-seq protocols [MBS15]. It can efficiently profile thousands of individual cells by separating them into microdroplets. The main advantage of this protocol is that a high number of scRNA-seq libraries can be sequenced at low cost. [ZVP17] states that Drop-seq is the most cost-effective method compared with other prominent scRNA-seq protocols, including Smart-seq, MARS-seq, SCRB-seq, and CEL-seq. Due to its low costs, Drop-seq is more preferable over other methods, especially when quantifying transcriptomes of large numbers of cells with low sequencing depth. As Drop-seq continues gaining more popularity, more Drop-seq specific computational tools are developed. These tools are tested either using spike-in experiments or on simulation datasets. Since spike-in experiments are expensive, simulated datasets have become a reasonable alternative. However, current RNA-seq simulators mostly target at bulk RNA-seq. This provokes the need of a specific scRNA-seq simulator for Drop-seq.ate

Drop-seq is a tag-based protocol, and its read distribution has an inherent bias across every transcript. In this paper, we refer this bias as the poly-A bias. We define a poly-A region as a consecutive sequence of 20 or more adenine nucleotides. The poly-A bias is generally observed from locations that are close to a poly-A region, which can be either the 3' end poly-A tail or the poly-A sequence inside the transcript. Compared to other locations on a transcript, more reads are generated near the poly-A regions, resulting in a skewed read distribution. The poly-A bias needs to be taken into account when developing simulation tools for Drop-seq. A successful simulator must be able to capture this positional bias in the sequencing process.

Current efforts in simulating RNA-seq target the more mature field of bulk RNA-seq as

opposed to scRNA-seq. ART [HLM11], Flowsim [BML10], Grinder [AWR12], FASTQsim [Shc14] and Polyester [FJL15] are widely-used bulk RNA-seq simulators that can provide a successful approximation of real data. While reads from Drop-seq are not uniformly distributed across a transcript due to the poly-A bias, current bulk RNA-seq simulators either assume a completely uniform distribution of reads, or generate the location of reads based on GC bias due to the PCR amplification [ERP16]. In other words, existing bulk RNA-seq simulators are not feasible for Drop-seq or any other tag-based scRNA-seq techniques. Since scRNA-seq is relatively new, the existing simulating techniques focus on simulating gene expression counts. Current simulation methods, including Splatter [ZPO17], Lun [LBM16], Lun 2 [LM17] and BASiCS [VMR15], aim to model the highly variable gene coverage and high dropout rates seen in scRNA-seq gene expression. They are successful in capturing scRNA-seq gene expression trends; however, they do not generate sequencing reads based on counts. As a result, a read simulator specifically targeting the Drop-seq technology must be developed, with the ability to generate sequencing reads and capture the trends found in Drop-seq data, namely the poly-A bias caused by the priming process.

In this paper, we propose Dropify, an end-to-end framework to simulate the sequencing reads of a Drop-seq experiment. Affected by the poly-A bias, reads are not uniformly drawn at the positions of a transcript, but have a higher probability being generated from positions near a poly-A region. A poly-A bias model is learned through the read distribution of different transcripts from real Drop-seq data. Based on the model, each position in a transcript is assigned with a different probability to be selected as the start position of a read. To simulate a new set of reads, Dropify begins with taking a list of reference transcripts to be sequenced and a transcript count matrix which specifies the number of transcripts per cell. Based on the learned probability, a list of starting positions of reads are generated with the positional sequencing biases. Polyester [FJL15] is then applied to generate reads for each starting position and implant sequencing error on reads. Lastly, to identify reads from different mRNA molecules and cells, we add the cell and molecular barcode to each read.

Dropify is a read level simulator for scRNA-seq data. It is a framework designed to provide reasonable simulation of Drop-seq data according to the user's experimental setting. Reads

generated by Dropify serve as a robust approximation to real data. Our results demonstrate that Dropify is able to simulate reads from various genes, and even different species. With the help of Dropify, researchers are able to avoid conducting complex experiments but simulate large amount of Drop-seq reads in a flexible and efficient way.

CHAPTER 2

Method

We propose a framework, Dropify, to simulate RNA-seq reads from the Drop-seq technology [MBS15]. Figure 2.1 demonstrates the overall framework. The input to the simulator is a list of reference transcripts and a matrix containing the number of transcripts for every cell in the experiment. The transcript count matrix can either be user defined or produced by a gene expression quantification simulator, such as Splatter [ZPO17] or BASiCS [VMR15]. Due to the poly-A bias caused by the priming process, reads are not uniformly distributed across a transcript. To simulate reads generated by Drop-seq, we learn the poly-A bias from real data. This is done by generating the probability for sequencing a read at different positions along a transcript. The probability distribution is then applied to the input transcripts, where we sample the starting positions accordingly for reads. Polyester [FJL15] takes these read positions paired with the input transcript sequences to generate sequencing reads. Lastly, we create unique barcodes for each transcript-cell combination to form the paired-end reads.

2.0.1 Generating Read Positions

To incorporate the poly-A bias in read simulation, a probability model is learned from a training dataset (described in Section 3) to mimic the skewed read distribution across transcripts obeserved in Drop-seq. A poly-A region is defined as a consecutive sequence of n adenine nucleotides or more ($n = 20$ by default). The model specifies the probability of generating a read at each position in relation to a poly-A region. Using a transcript's reference sequence and the learned poly-A bias model, we generate a transcript specific probability distribution of generating a read at different positions. We sample from the transcript's probability distribution to obtain a list of positions. The positions selected from

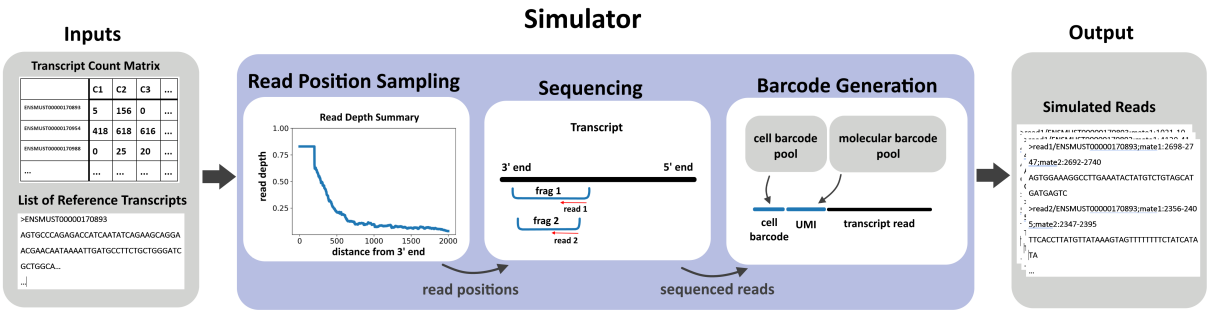

Figure 2.1: The simulator pipeline. The simulator takes a list of transcript and a transcript–cell counts matrix as input. The input transcripts should be provided in FASTA format; the transcript count matrix contains rows which represent a transcript, columns which represent a cell, and entries which specify the number of transcripts per cell. The simulation process begins with generating read positions, where we adapt our learned poly-A bias probability distribution to transcripts to sample read start positions. During fragmentation, the input reference transcripts are broken into short fragments and sequenced beginning at the start positions obtained in the previous step. Single-end RNA-seq reads are generated from each fragment containing concrete cDNA sequences. Finally, cell barcodes and molecular barcodes (UMI) are simulated by randomly generating sequences of nucleotides and added to the appropriate read. The result of the simulator is saved in FASTA format, where each file contains the reads for one cell.

this step are used as the read start positions.

2.0.1.1 Model Training

As discussed in Section 1, the priming process in Drop-seq experiments to capture mRNA molecules causes a significant poly-A bias in read distribution. The probability of observing a read at each position along a transcript is not equal, yet with a higher probability for positions closer to a poly-A region. We present a learned poly-A bias model to describe the distribution of reads affected by the poly-A bias in Drop-seq data, and use that to assign the probability of generating a read based on the distance to the nearest poly-A region on a transcript.

The model is built on a training dataset D consisting of reads from a Drop-seq experiment. The reads are aligned to the reference genome. We identify the corresponding transcript for each read by comparing the aligned position from the alignment results with the transcript's genome position. A read distribution is generated for each transcript, which specifies the read count at each position along the transcript. To incorporate the poly-A bias, for every position at a given transcript, we calculate its distance to the nearest poly-A region. Since the sequencing process starts with the 3' end, only poly-A regions on the 3' end side are considered when calculating the distance. Specifically, if there is no poly-A sequence within the transcript, the distance to the nearest poly-A region is same as the distance to the 3' end. Eventually, for every transcript j in training data D, we generate a read distribution for that transcript with respect to the distance to a poly-A region. To be more specific, the read distribution G_j for transcript j specifies the read count d_{ij} at each distance i to its nearest poly-A region.

We use a mixture model to estimate the sampling probability at different distances. Let H be the probability distribution for a transcriptome, i be the distance to the nearest poly-A region, and D is the training data. The probability of generating a read at distance i is defined in Equation 2.1. The probability is determined by two components, the raw probability r_i and a length factor L determined by the read length. We use a weight function $w(i)$ to

adjust how r_i and L contribute to the sampling probability at distance i. w is defined in Equation 2.2.

$$H(x = i|D) = w(i)\frac{1}{L} + (1 - w(i))r_i \tag{2.1}$$

$$w(i) = \begin{cases} 1 & \text{if } i \leq L \\ \frac{1}{i} & \text{if } i > L \end{cases} \tag{2.2}$$

The raw probability r_i from Equation 2.1 serves as one factor to estimate the sampling probability. We calculate r_i based on the amount of reads observed in the training data. Suppose x_{ij} is the read count at distance i from the transcript read distribution G_j, $max(x_j)$ is the maximum read count of G_j, and $|T|$ is the number of transcripts in D. r_i is defined in Equation 2.3. Since it is possible that reads are mapped to wrong location during the alignment process, we normalize the read count per transcript to avoid risking that transcripts with a great number of incorrectly mapped reads skew the average of all data. For each transcript, the read count at distance i is normalized by its maximum read count. We calculate r_i by summing over the normalized read count at distance i over all transcripts, and taking the average of the result. r_i is regarded as the raw probability of generating a read at distance i.

$$r_i = \frac{1}{|T|}\sum_{j=0}^{|T|}\frac{x_{ij}}{\max(x_j)} \tag{2.3}$$

The read length serves as the other factor that affects the sampling probability. The read length is adjustable in Drop-seq experiments. It has been shown that for scRNA-seq data, different read length has an impact on the read coverage distribution across a transcript [REL17]. For short transcripts of which the length is close to the read length, the reads tend to be uniformly distributed instead of showing the poly-A bias trend. Generally, for nucleotides closer to a poly-A region, the read distribution is almost uniform. Therefore, we adopt a uniform sampling strategy for the first L nucleotides starting from the poly-A

region. By default, $L = 4R$ nucleotides, where R is the read length. We choose L based on cross validation results, but user may change L depending on the experimental setting.

H is the learned poly-A bias distribution. w determines the weight of two components. When the distance i to the nearest poly-A region is below the threshold L, the model adopts a uniform sampling strategy. The probability of each distance to be sampled is equal. However, as the distance increases, the read distribution is no longer uniform but transforms into a skewed shape affected by the poly-A bias. Therefore, once the distance exceeds the threshold L, the other component r_i is introduced into the model, the raw probability estimated from the transcript read distribution form the training data. With the distance increasing, the uniform sampling strategy is weighed less, and r_i becomes the major factor. When the distance is large, the sampling probability is almost completely dominated by r_i. H is used as the probability distribution when we sample read start positions for individual transcripts.

2.0.1.2 Sampling Positions

The probability of generating a read at each position along a transcript is based on the learned distribution H and the locations of poly-A regions on a transcript. For a given transcript t, we create a probability distribution $P_t(x = i|H)$ as defined in equation 2.4 describing the probability of generating a read at each position i in relation to the 3' end of the transcript.

N represents the number of poly-A regions between position i and the 3' end contained on the transcript, d_{ij} is the distance from position i to the jth poly-A region, c serves as the weight factor that determines the weight of the poly-A tail at the 3' end of the transcript. p_i and $p_{d_{ij}}$ are calculated from the learned poly-A bias model, that specifies the probability of generating a read at position i and d_{ij} as defined in equation2.5.

$$P_t(x = i|H) = \begin{cases} cp_i + \frac{1-c}{N} \sum_{j=1}^{N} p_{d_{ij}}, & \text{if } N \geq 1 \\ p_i, & N = 0 \end{cases} \tag{2.4}$$

$$p_i = H(x = i|D) \tag{2.5}$$

The calculation of P_t considers both the poly-A tail at the 3' end and the poly-A regions within a transcript. If there is no poly-A region between position i and the 3' end, the poly-A tail is the only factor that contributes to the bias. If there are at least one poly-A region between position i and the 3' end, the model calculates the poly-A tail and the poly-A regions inside the transcript separately.

We use c as the weight factor to adjust the weight of the poly-A tail. The length of a poly-A region can affect the ability to attract primer beads. If a poly-A tail is longer than the poly-A regions contained in the transcript, it could be beneficial to weigh the probability due to the distance from the poly-A tail more heavily than the probability due to contained poly-A regions. Since most transcripts do not contain a long consecutive sequence of adenine nucleotides, c is set to 0.5 by default.

We calculate p_i and $p_{d_{ij}}$ from the poly-A bias distribution H estimated from the training data. After generating the probability distribution P_t for each individual transcript, we sample from P_t to obtain a list of read starting positions.

As a result of computational analysis of Drop-seq data, the raw sequence reads are transformed into a digital gene expression matrix, which measures the number of times each gene is expressed in each individual cell. Since the mRNA molecules are augmented during the amplification process to guarantee sufficient read coverage for individual transcript, the counts from the digital gene expression matrix do not represent the actual number of sequenced reads. An amplification factor is introduced to determine the amount of reads sequenced from each transcript. To be more specific, the amplification factor determines the number of time we sample from the probability distribution P_t for each individual transcript.

2.0.2 Sequencing

Given a list of starting positions and the reference transcript sequences, we use Polyester [FJL15] to generate sequence fragments for the input transcripts. Based on these fragments, Polyester simulates reads with certain probability of sequencing error.

During fragmentation, transcript sequences are broken into short fragments. Polyester

is applied to generate these fragments. The fragment lengths are sampled from a normal distribution provided in Polyester, with fragment mean $\mu = 100$ and fragment standard deviation $\sigma = 10$. We apply a list of positions sampled from the transcript probability distribution P_t in the previous step as the start positions of each fragment. Single-end reads are drawn from the first R nucleotides of the fragments, where R is the read length.

Drop-seq sequencing data is presented in the form of paired-end reads, where the first read yields the cell barcode and molecular barcode (UMI), and the second read represents part of the transcript sequence. The second read is generated by Polyester. For the sequencing error model, since Drop-seq data is sequenced on the Illumina platform, we apply the empirical error model provided by Polyester to simulate the sequencing errors. The empirical model is estimated from Illumina data. It calculates the probabilities of making each of the four possible sequencing errors at each position in the read. Users may also choose other sequencing error models provided by Polyester. The output of this step is in the form of single end reads and is written to FASTA format.

2.0.3 Barcode Simulation

The Drop-seq technology generates reads in paired-end. The first end contains a 12-nt cell barcode and 8-nt molecular barcode (UMI). The cell barcodes are identical across all the primers on one bead, but different from the ones on other beads, indicating the cell of origin, while a UMI is different on each primer. In Drop-seq, both cell barcodes and molecular barcodes are constructed through the completion of "split-and-pool" synthesis cycles[MBS15]. In the simulation process, the cell barcode is a permutation of different DNA bases (A, G, C, or T) with size 12, and the UMI is a permutation with size 8. Based on the input transcript count matrix, each cell is distributed with a unique cell barcode, and each transcript is distributed with a unique UMI. Reads that originated from the same transcript have the same UMI, but reads from different transcripts possess different UMIs.

CHAPTER 3

Datasets

We use public Drop-Seq datasets available on Gene Expression Omnibus (GEO)[1] to train and evaluate our model. The first dataset (accession number GSM1544798) contains a mixture of human and mouse cells, with a read length of 50bp. The second dataset (accession number GSM2177570) is composed of the mouse retina cells, with a read length of 60bp. Reads of the mouse cells from GSM1544798 are used for training; the rest of the data are used as the test sets for evaluation. We follow the Drop-seq core computational protocol suggested by Steve McCarroll's lab [MBS15] for data pre-processing. Reads are alinged by STAR [DDS13]. The reference genome and isoform annotation information is based on hg19 for human and mm10 for mouse. Since reads are generally shorter than transcripts from which they are derived, a single read may map to multiple genes and isoforms, complicating expression analyses [LRS09]. Therefore, we discard reads that map to multiple locations in order to reduce the uncertainty. In efforts to remove potential noises in the data, we filter out the reads that have a base quality score less than 10.

The training set consists of reads from the mouse cells in sample GSM1544798. We add more stringent rules for filtering this dataset. Since multiple isoform genes add ambiguities and uncertainties when performing expression analyses [LRS09], the model is only trained on single isoform transcripts. We also remove less expressed transcripts whose average read depth per nucleotide is less than 2. This leaves us with 401 transcripts to train our model on.

[1] https://www.ncbi.nlm.nih.gov/geo/

CHAPTER 4

Experiment

4.0.1 Evaluation Metrics

To demonstrate reads generated by Dropify exhibit realistic properties, we must be able to measure how well the read distribution of our simulated reads matches those of the true Drop-seq reads. When we create a read using Dropify, we already know which transcript the read has been sequenced from and we can directly plot a read to its transcript of origin. However, the alignment process, as a required step to generate the read distributions, can potentially map reads to incorrect locations. Alignment error exists when reads are either be mapped to the wrong transcript or completely unmapped. To reintroduce the bias caused by the aligner to the simulated data, we run the same alignment tool (the RNA-seq aligner STAR [DDS13]) on the simulated reads before evaluation.

After aligning the simulated data, there becomes a discrepancy between the total number of reads on a given transcript before alignment and after alignment. This makes it difficult to determine whether differences between the simulated data and real data are due to the fact that there is a difference in the read counts, or if there is actually a difference in the general shape of the distributions. Therefore, a normalization step is required to correct for the reduced read counts of simulated reads after alignment. To normalize the data, the read count at each position is scaled by the total number of reads before alignment to the total number of reads after alignment. For a specific transcript, let p be the read distribution before alignment, a be the read distribution after alignment, a_i represents the read count at position i on the transcript after alignment. We define n_i as the read count at position i after normalization. n_i is calculated as shown in Equation 4.1. The result from alignment and normalization process is shown in Figure 4.1. The whole process corrects for the bias

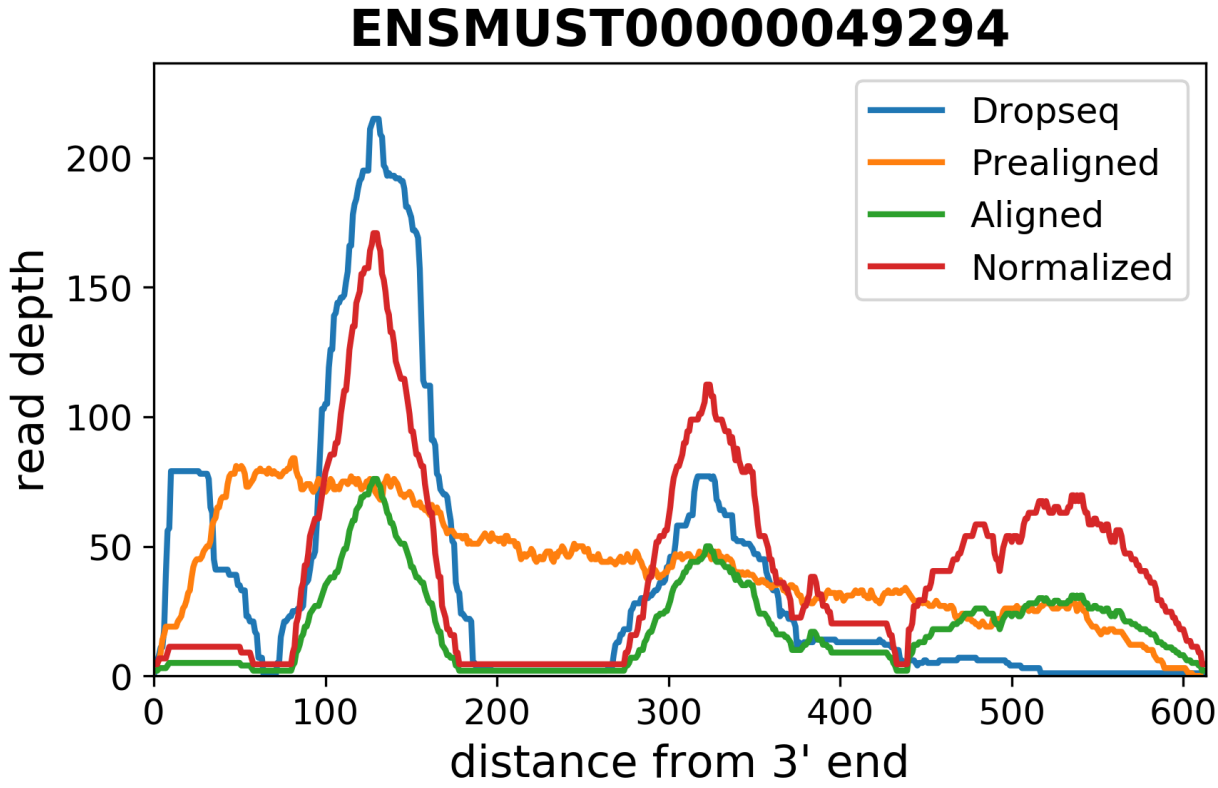

Figure 4.1: This is an example of how the alignment and normalization process affects an individual transcript's read distribution. Gaps in the aligned read distribution such as the one between position 180 to 280 mean that reads originally generated from that region are either mapped to another location or completely thrown out. As a result, the aligned line (green) has much fewer reads than the line before alignment (orange, which is also the simulated line). We need to normalize the read count to correct for the effects of the aligner. After increasing the number of reads to match that of the reads before alignment, we obtain a distribution that can be evaluated against the original Drop-seq distribution (blue) as shown by the normalized line (red).

introduced by the aligner, thus providing a more reasonable approach to evaluate the model.

$$n_i = \frac{\sum p}{\sum a} a_i \tag{4.1}$$

To evaluate the performance of our model, we show how closely we can model the read count distribution along a transcript. We allow for slight positional variability by binning together multiple nucleotides at a time and taking average of those nucleotides as the read count value. A sliding window of 100 nucleotides is applied along a transcript to produce a smoother distribution as shown in figure 4.2. After producing these distributions, we use two different metrics to quantitatively score how closely the simulated distributions match up to real Drop-seq distributions. The first score is the pearson correlation coefficient defined in equation 4.2. N is the number of bins in the transcript, x is the read count at each position in our simulated distribution and y is the read count at each position in the Drop-seq sequenced distribution. The pearson score is a value from 0 to 1 where 1 means that the two distributions are very similar.

$$\text{Pearson} = \frac{N \sum xy - (\sum x * \sum y)}{\sqrt{N \sum x^2 - (\sum x)^2} \sqrt{N \sum y^2 - (\sum y)^2}} \tag{4.2}$$

The second score that we use is BC distance defined in 4.3, which generates a number between 0 and 1, and 0 means that the distributions are very similar.

$$\text{BC dist} = \frac{\sum(|x - y|)}{\sum(x + y)} \tag{4.3}$$

4.0.2 Naive vs. Our Model

This section compares how well Dropify performs against an industry baseline method. Current simulators either use a uniform model or apply CG bias to generate reads across a transcript [ZVP17]. However, to model the read distribution form a tag-based scRNA-seq protocol such as Drop-seq, the simulator must include positional bias, namely the poly-A bias, which results in an increase in number of reads near poly-A regions. [AC18] suggests that there is a strong 3' end peak in read coverage, as a long poly-A tail exists at the 3' end

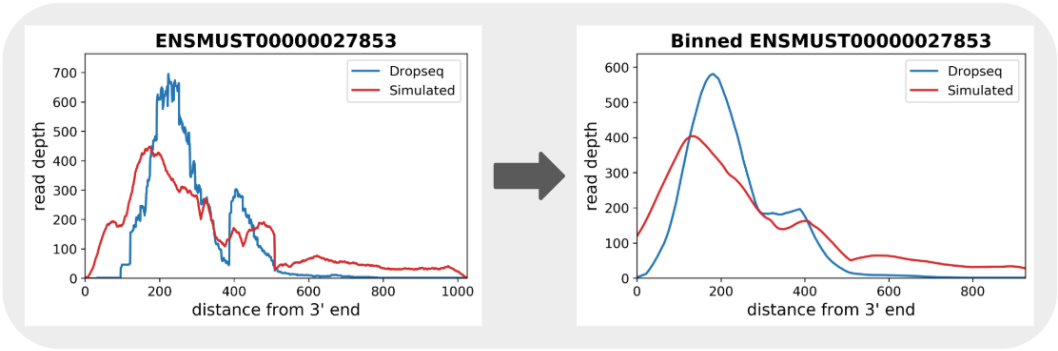

Figure 4.2: The read distribution shown on the left has slight variability in position. For example, the blue line starts around position 100 which could vary based on indescribable features. We allow for slight variability using the binning process. The figure on the right shows the same read counts binned by 100 nucleotides.

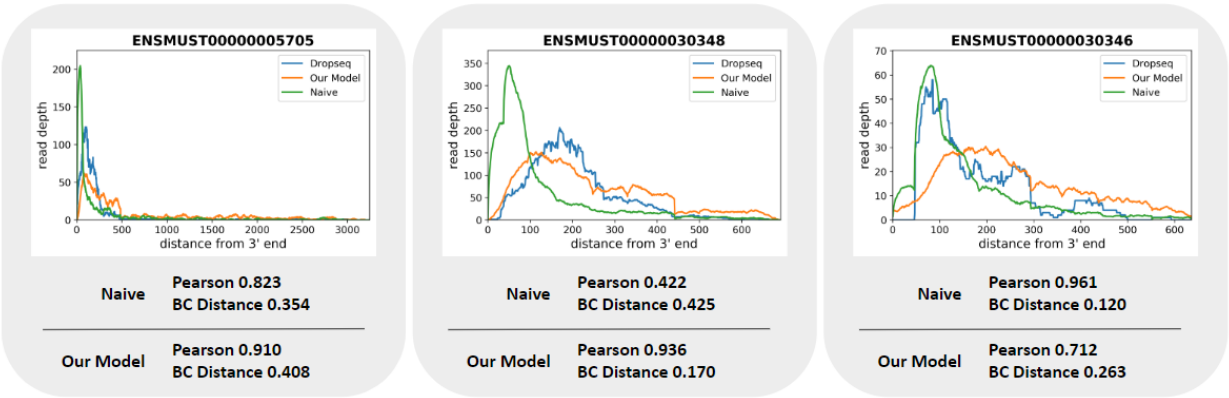

Figure 4.3: These plots show examples of transcript read distributions from simulated reads by Dropify against the naive model on Drop-seq sequenced data. The blue line represents Drop-seq data, the orange line represents the simulated data by Dropify, and the green line is the simulated data by the naive model. All the simulated results are plotted after alignment and normalization.

of the transcript. An intuitive approach to generate this peak would be to sample from a distribution where the probability of generating a read at the 3' end is significantly higher than at any other positions. Equation 4.4 defines a distribution that follows this trend, where i is the distance from a nucleotide to the 3' end of the transcript. $P(x = i)$ defines the probability of generating a read at position i. We use $P(x)$ as the baseline model.

$$P(x = i) = \frac{1}{i} \tag{4.4}$$

Table 4.1 presents the evaluation results of simulated reads by Dropify in comparison to those by the naive model on the training and testing datasets. Dropify performs better than the naive model by 55.7% in terms of pearson correlation coefficient. Figure 4.3 shows the read coverage of Drop-seq data, Dropify, and the naive model for transcripts ENSMUST00000005705, ENSMUST00000030348, and ENSMUST00000030346. The naive model consistently overproduces reads at the 3' end of the transcript suggesting that it doesn't describe the true read coverage distribution over a transcript. As seen in transcript ENSMUST00000005705, both models generate a peak at the 3' end with a decline in number of reads as distance from the 3' end increases. However, the naive model still overproduces reads at the 3' end of the transcript. The same trend is more obvious as shown in transcript ENSMUST00000030348 which is a slightly shorter transcript. The read distribution is relatively flattened, yet the naive model generates a sharp peak at the 3' end. Despite this notion, the naive model can perform well on a few transcripts such as the one shown in ENSMUST00000030346 where the distribution from the real data has a relatively high peak. This case is a small subset of the data, and Dropify performs better on the data as a whole.

4.0.3 Transcripts from Multiple Isoform Genes vs. Single Isoform Genes

Multiple isoform genes present an interesting problem for analyzing scRNA-seq data. If there is a read mapped to the 3' end of a transcript and the only distinct differences between isoforms are regions other than the 3' end, then it is extremely difficult to determine which

	Naive Model		Our Model	
	Pearson	BC Distance	Pearson	BC Distance
GSM1544798 mouse (training set)	0.546	0.437	0.710	0.488
GSM2177570 (test set)	0.510	0.414	0.794	0.312

Table 4.1: Comparison between naive model and our empirically produced model. We calculate these results using all the transcripts in each sample that have reads mapped to them. Our model performs better in terms of pearson correlation and BC distance.

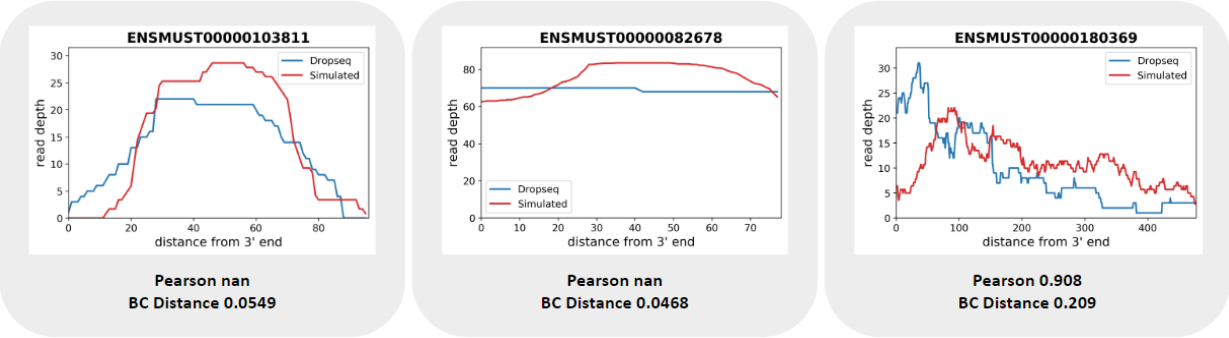

Figure 4.4: These graphs shows some examples pulled from the testing set coming from sample GSM2177570. All of these transcripts have only one isoform. We compare how closely the simulated reads by Dropify match the read distribution of Drop-seq reads. We plot the simulated line after alignment and normalization.

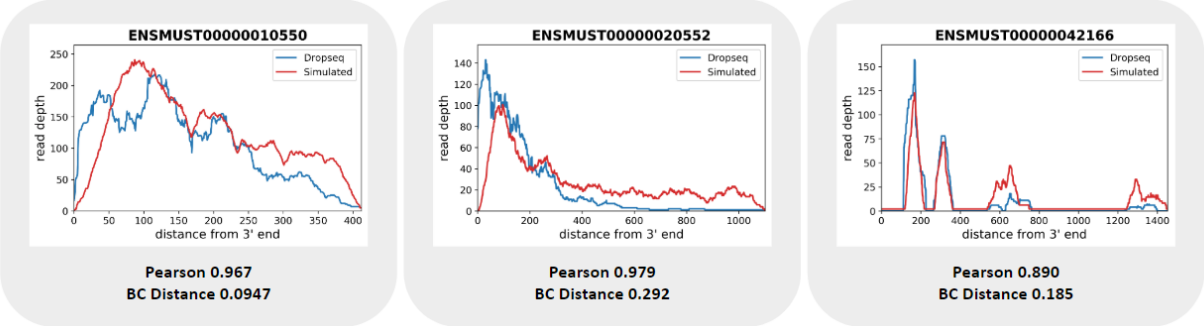

Figure 4.5: These graphs show some examples of transcripts that have more than one isoform from sample GSM2177570. We compare how closely the simulated reads by Dropify match the read distribution of Drop-seq reads. We plot the simulated line after alignment and normalization.

transcript the read was originally sequenced from [AC18]. Most gene quantification tools are especially burdened by the multiple isoform issue and are only able to report quantification at the exon level as opposed to at the transcript level [AC18]. Despite these challenges, Dropify is able to accurately simulate read coverage distribution of transcripts with multiple isoforms. Table 4.2 shows how well Dropify simulates read coverage on transcripts coming from single isoform genes verses all transcripts.

Transcripts from single isoform genes perform with a pearson correlation coefficient of 0.857 which is better than that of multiple isoform genes. Figure 4.4 presents some examples of the read distribution along individual transcripts. Transcripts ENSMUST00000103811 and ENSMUST00000082678 are shorter, so they are more affected by the beginning part of the poly-A bias probability distribution. Because of this, there is no definitive tail following the 3' end peak. Transcript ENSMUST00000180369 is longer and therefore, it is apparent that there are less reads further from the 3' end.

Dropify performs slightly better on the single isoform case than the multiple isoform case. The pearson correlation for the single isoform case in our training set is 14% higher than the multiple isoform case. However, that gap decreases in the testing data. The pearson correlation coefficient is 1% higher in the single isoform case than the multiple

	Single Isoform Transcripts		All Transcripts	
	Pearson	BC Distance	Pearson	BC Distance
GSM1544798 mouse (training set)	0.857	0.634	0.710	0.488
GSM2177570 (test set)	0.804	0.186	0.794	0.312

Table 4.2: Comparison between transcripts from single isoform genes and the entire set of transcripts.

isoform case from the testing data. A possible reason behind why the training correlation for the single isoform case is higher is that we trained the poly-A bias model on that case, whereas the other sets are not used in generating the model. Figure 4.5 shows some isolated transcripts that come from multiple isoform genes. Transcripts ENSMUST00000010550 and ENSMUST00000020552 follow the 3' end peak trend which Dropify is able to reproduce. Many transcripts also appear to look like transcript ENSMUST00000042166 where there is no clear 3' end peak with a gradual decline. Instead, there might be sporadic peaks and gaps in read distribution. These artifacts are due to the aligner. Even though the aligner modifies the true read coverage distribution, we can still evaluate our model since we reintroduce the biases from the alignment process in our evaluation metrics.

4.0.4 Testing Different Species

In previous sections, we test Dropfiy using a dataset composed of mouse cells. To test the robustness of the simulator, we also compare how well Dropify produces reads for mouse data against human data. Table 4.3 shows the results from the different species testing datasets.

The human testing set of reads from our simulated data appear to be more correlated to real Drop-seq data than the mouse testing set. One potential explanation could be that the

	Different Species	
	Pearson	BC Distance
GSM1544798 mouse (training set)	0.710	0.488
GSM2177570 (test set)	0.794	0.312
GSM1544798 human (training set)	0.834	0.285

Table 4.3: Comparison of mouse and human read coverage distribution correlation.

sample of human cells comes from the same experiment that we trained our model on. The major takeaway from this experiment is that human transcript sequences are completely novel to the model. Regardless, Dropify is able to be extended and simulate these new transcripts because they appear to follow the same poly-A bias as the training data.

CHAPTER 5

Conclusion

In this paper, we propose an end-to-end framework for producing simulated reads from a Drop-seq experiment. Dropify uses an empirically produced probability distribution to model positional bias in read coverage. By sampling from this distribution, we are able to generate the positions of reads where the simulated reads follow the trend found in Drop-seq data. Reads generated by Dropify exhibit realistic properties and are useful in practice.

REFERENCES

[AC18] Ángeles Arzalluz-Luque and Ana Conesa. "Single-cell RNAseq for the study of isoformshow is that possible?" *Genome biology*, **19**(1):110, 2018.

[AWR12] Florent E Angly, Dana Willner, Forest Rohwer, Philip Hugenholtz, and Gene W Tyson. "Grinder: a versatile amplicon and shotgun sequence simulator." *Nucleic acids research*, **40**(12):e94–e94, 2012.

[BML10] Susanne Balzer, Ketil Malde, Anders Lanzén, Animesh Sharma, and Inge Jonassen. "Characteristics of 454 pyrosequencing dataenabling realistic simulation with flowsim." *Bioinformatics*, **26**(18):i420–i425, 2010.

[DDS13] Alexander Dobin, Carrie A Davis, Felix Schlesinger, Jorg Drenkow, Chris Zaleski, Sonali Jha, Philippe Batut, Mark Chaisson, and Thomas R Gingeras. "STAR: ultrafast universal RNA-seq aligner." *Bioinformatics*, **29**(1):15–21, 2013.

[ERP16] Merly Escalona, Sara Rocha, and David Posada. "A comparison of tools for the simulation of genomic next-generation sequencing data." *Nature Reviews Genetics*, **17**(8):459, 2016.

[FJL15] Alyssa C Frazee, Andrew E Jaffe, Ben Langmead, and Jeffrey T Leek. "Polyester: simulating RNA-seq datasets with differential transcript expression." *Bioinformatics*, **31**(17):2778–2784, 2015.

[HLB18] Byungjin Hwang, Ji Hyun Lee, and Duhee Bang. "Single-cell RNA sequencing technologies and bioinformatics pipelines." *Experimental & molecular medicine*, **50**(8):96, 2018.

[HLM11] Weichun Huang, Leping Li, Jason R Myers, and Gabor T Marth. "ART: a next-generation sequencing read simulator." *Bioinformatics*, **28**(4):593–594, 2011.

[HWS12] Tamar Hashimshony, Florian Wagner, Noa Sher, and Itai Yanai. "CEL-Seq: single-cell RNA-Seq by multiplexed linear amplification." *Cell reports*, **2**(3):666–673, 2012.

[IKM11] Saiful Islam, Una Kjällquist, Annalena Moliner, Pawel Zajac, Jian-Bing Fan, Peter Lönnerberg, and Sten Linnarsson. "Characterization of the single-cell transcriptional landscape by highly multiplex RNA-seq." *Genome research*, **21**(7):1160–1167, 2011.

[IZJ14] Saiful Islam, Amit Zeisel, Simon Joost, Gioele La Manno, Pawel Zajac, Maria Kasper, Peter Lönnerberg, and Sten Linnarsson. "Quantitative single-cell RNA-seq with unique molecular identifiers." *Nature methods*, **11**(2):163, 2014.

[JKK14] Diego Adhemar Jaitin, Ephraim Kenigsberg, Hadas Keren-Shaul, Naama Elefant, Franziska Paul, Irina Zaretsky, Alexander Mildner, Nadav Cohen, Steffen Jung, Amos Tanay, et al. "Massively parallel single-cell RNA-seq for marker-free decomposition of tissues into cell types." *Science*, **343**(6172):776–779, 2014.

[KDA14] Eleonora de Klerk, Johan T Den Dunnen, and Peter ACt Hoen. "RNA sequencing: from tag-based profiling to resolving complete transcript structure." *Cellular and molecular life sciences*, **71**(18):3537–3551, 2014.

[KKS15] Aleksandra A Kolodziejczyk, Jong Kyoung Kim, Valentine Svensson, John C Marioni, and Sarah A Teichmann. "The technology and biology of single-cell RNA sequencing." *Molecular cell*, **58**(4):610–620, 2015.

[KMA15] Allon M Klein, Linas Mazutis, Ilke Akartuna, Naren Tallapragada, Adrian Veres, Victor Li, Leonid Peshkin, David A Weitz, and Marc W Kirschner. "Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells." *Cell*, **161**(5):1187–1201, 2015.

[LBM16] Aaron TL Lun, Karsten Bach, and John C Marioni. "Pooling across cells to normalize single-cell RNA sequencing data with many zero counts." *Genome biology*, **17**(1):75, 2016.

[LM17] Aaron TL Lun and John C Marioni. "Overcoming confounding plate effects in differential expression analyses of single-cell RNA-seq data." *Biostatistics*, **18**(3):451–464, 2017.

[LRS09] Bo Li, Victor Ruotti, Ron M Stewart, James A Thomson, and Colin N Dewey. "RNA-Seq gene expression estimation with read mapping uncertainty." *Bioinformatics*, **26**(4):493–500, 2009.

[MBS15] Evan Z Macosko, Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, Allison R Bialas, Nolan Kamitaki, Emily M Martersteck, et al. "Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets." *Cell*, **161**(5):1202–1214, 2015.

[REL17] Simone Rizzetto, Auda A Eltahla, Peijie Lin, Rowena Bull, Andrew R Lloyd, Joshua WK Ho, Vanessa Venturi, and Fabio Luciani. "Impact of sequencing depth and read length on single cell RNA sequencing data of T cells." *Scientific reports*, **7**(1):12781, 2017.

[RLW12] Daniel Ramsköld, Shujun Luo, Yu-Chieh Wang, Robin Li, Qiaolin Deng, Omid R Faridani, Gregory A Daniels, Irina Khrebtukova, Jeanne F Loring, Louise C Laurent, et al. "Full-length mRNA-Seq from single-cell levels of RNA and individual circulating tumor cells." *Nature biotechnology*, **30**(8):777, 2012.

[SCS14] Magali Soumillon, Davide Cacchiarelli, Stefan Semrau, Alexander van Oudenaarden, and Tarjei S Mikkelsen. "Characterization of directed differentiation by high-throughput single-cell RNA-Seq." *BioRxiv*, p. 003236, 2014.

[Shc14] Anna Shcherbina. "FASTQSim: platform-independent data characterization and in silico read generation for NGS datasets." *BMC research notes*, **7**(1):533, 2014.

[TBW09] Fuchou Tang, Catalin Barbacioru, Yangzhou Wang, Ellen Nordman, Clarence Lee, Nanlan Xu, Xiaohui Wang, John Bodeau, Brian B Tuch, Asim Siddiqui, et al. "mRNA-Seq whole-transcriptome analysis of a single cell." *Nature methods*, **6**(5):377, 2009.

[VMR15] Catalina A Vallejos, John C Marioni, and Sylvia Richardson. "BASiCS: Bayesian analysis of single-cell sequencing data." *PLoS computational biology*, **11**(6):e1004333, 2015.

[WNK13] Angela R Wu, Norma F Neff, Tomer Kalisky, Piero Dalerba, Barbara Treutlein, Michael E Rothenberg, Francis M Mburu, Gary L Mantalas, Sopheak Sim, Michael F Clarke, et al. "Quantitative assessment of single-cell RNA-sequencing methods." *Nature methods*, **11**(1):41, 2013.

[ZPO17] Luke Zappia, Belinda Phipson, and Alicia Oshlack. "Splatter: simulation of single-cell RNA sequencing data." *Genome biology*, **18**(1):174, 2017.

[ZVP17] Christoph Ziegenhain, Beate Vieth, Swati Parekh, Björn Reinius, Amy Guillaumet-Adkins, Martha Smets, Heinrich Leonhardt, Holger Heyn, Ines Hellmann, and Wolfgang Enard. "Comparative analysis of single-cell RNA sequencing methods." *Molecular cell*, **65**(4):631–643, 2017.