RESEARCH ARTICLE

Scuphr: A probabilistic framework for cell lineage tree reconstruction

Hazal Koptagel[1,2], **Seong-Hwan Jun**[3], **Joanna Hård**[4], **Jens Lagergren**[1,2]*

1 School of EECS, KTH Royal Institute of Technology, Stockholm, Sweden, **2** Science for Life Laboratory, Stockholm, Sweden, **3** Department of Biostatistics and Computational Biology, University of Rochester Medical Center, Rochester, New York, United States of America, **4** Department of Cell and Molecular Biology, Karolinska Institutet, Stockholm, Sweden

* jensl@kth.se

Citation: Koptagel H, Jun S-H, Hård J, Lagergren J (2024) Scuphr: A probabilistic framework for cell lineage tree reconstruction. PLoS Comput Biol 20(5): e1012094. https://doi.org/10.1371/journal.pcbi.1012094

Editor: Qing Nie, University of California Irvine, UNITED STATES

Received: December 12, 2022

Accepted: April 20, 2024

Published: May 9, 2024

Data Availability Statement: The fibroblast dataset used in the biological data experiments is published and available at the European Genome-phenome Archive (EGA) with the accession number EGAS00001003108. The source code of Scuphr is available at https://github.com/Lagergren-Lab/scuphr.

Funding: Funding is provided by the Swedish Foundation for Strategic Research Grant BD15-0043 and Swedish Research Council through the grants 2022-03516 and 2018-06217 to JL. The funders had no role in study design, data collection

Abstract

Cell lineage tree reconstruction methods are developed for various tasks, such as investigating the development, differentiation, and cancer progression. Single-cell sequencing technologies enable more thorough analysis with higher resolution. We present Scuphr, a distance-based cell lineage tree reconstruction method using bulk and single-cell DNA sequencing data from healthy tissues. Common challenges of single-cell DNA sequencing, such as allelic dropouts and amplification errors, are included in Scuphr. Scuphr computes the distance between cell pairs and reconstructs the lineage tree using the neighbor-joining algorithm. With its embarrassingly parallel design, Scuphr can do faster analysis than the state-of-the-art methods while obtaining better accuracy. The method's robustness is investigated using various synthetic datasets and a biological dataset of 18 cells.

Author summary

Cell lineage tree reconstruction carries a significant potential for studies of development and medicine. The lineage tree reconstruction task is especially challenging for cells taken from healthy tissue due to the scarcity of mutations. In addition, the single-cell whole-genome sequencing technology introduces artifacts such as amplification errors, allelic dropouts, and sequencing errors. We propose Scuphr, a probabilistic framework to reconstruct cell lineage trees. We designed Scuphr for single-cell DNA sequencing data; it accounts for technological artifacts in its graphical model and uses germline heterozygous sites to improve its accuracy. Scuphr is embarrassingly parallel; the speed of the computational analysis is inversely proportional to the number of available computational nodes. We demonstrated that Scuphr is fast, robust, and more accurate than the state-of-the-art method with the synthetic data experiments. Moreover, in the biological data experiment, we showed Scuphr successfully identifies different clones and further obtains more support on closely related cells within clones.

and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

Introduction

Reconstructing cell lineage trees, from single-cell data, for healthy tissue is a fundamental computational problem with enormous potential for studies of development and differentiation [1–6]. There are two related reconstruction problems for cancer tumors: reconstruction of clonal trees and the reconstruction of tumor phylogenies from single cells. Several data types, e.g., bulk DNA, single-cell DNA, and single-cell RNA, have been used for the latter two problems [7–11]. All these reconstruction methods exploit mutations and attempt to reconstruct trees in which the proximity between a pair of cells, or clones, is correlated with the similarity between their patterns of mutations. The somatic mutation rate in humans is 10^{-9} per locus per cell division [12], and the copy number is not considered to carry substantial information regarding cell lineage membership in healthy tissue. Therefore, mutations are scarce when reconstructing lineage trees for healthy tissues, implying that more sophisticated models and computational methods are needed to capitalize fully on the existing mutations. This scarcity also highlights the need for single-cell DNA sequencing (scDNA-seq) data since it reveals more point mutations than any other current data type [13].

Regardless of its potential to reveal mutations, scDNA-seq data comes with its challenges [14–18]. Due to the small amount of genomic data available in a single cell, the genome needs to be amplified before sequencing [19]. Unfortunately, the whole-genome amplification methods, such as the multiple displacement amplification (MDA) method [20] and the multiple annealing and looping-based amplification cycles (MALBAC) method [21], introduce technical artifacts known as amplification errors (AEs) that are hard to distinguish from mutations. Moreover, so-called allelic dropout (ADO) events remain even after the amplification. In addition, the subsequent sequencing of the amplified materials introduces sequencing errors [22–25].

There have been several methods explicitly made for scDNA-seq data, both for identifying mutations (single nucleotide variant (SNV) callers) and for reconstructing cell lineage trees, although several of them are targeting cancer data. Monovar [26] is an SNV caller designed specifically for scDNA-seq data; for each position, it models the ADO with a Bernoulli distribution, the AEs with independent and identically distributed (i.i.d.) Bernoulli random variables and base-calling error probabilities depend on Phred quality scores, [27, 28], while utilizing dynamic programming. LiRA [29] and Conbase [30] are scDNA-seq SNV callers that leverage read-phasing, while the latter does variant calling based on the population of single cells.

There has been a sequence of single-cell tree reconstruction methods targeting cancer data, [31–36], leading up to the SCIΦ method [37]. Interestingly, for the cancer case, the infinite sites assumption (ISA, [38–40]) may be violated due to segmental deletions. However, for healthy tissue, the ISA is an appropriate assumption. So, since SCIΦ is based on ISA, it is also relevant to the analysis of healthy tissue. SCIΦ has a probabilistic model that allows joint SNV calling and tree reconstruction using the Markov chain Monte Carlo (MCMC) method. More recently, the Phylovar [41] method was shown to handle millions of loci and be faster than SCIΦ while having similar accuracy by taking advantage of the efficient vectorized computations.

Method overview

Scuphr is a distance-based phylogenetic inference method that reconstructs cell lineage trees from scDNA-seq data, produced by experimental procedures that amplify the cells' genomes, using amplification methods such as the MDA method [20] and the MALBAC method [21]. Analyses of this data type need to distinguish somatic mutations from sequencing errors and

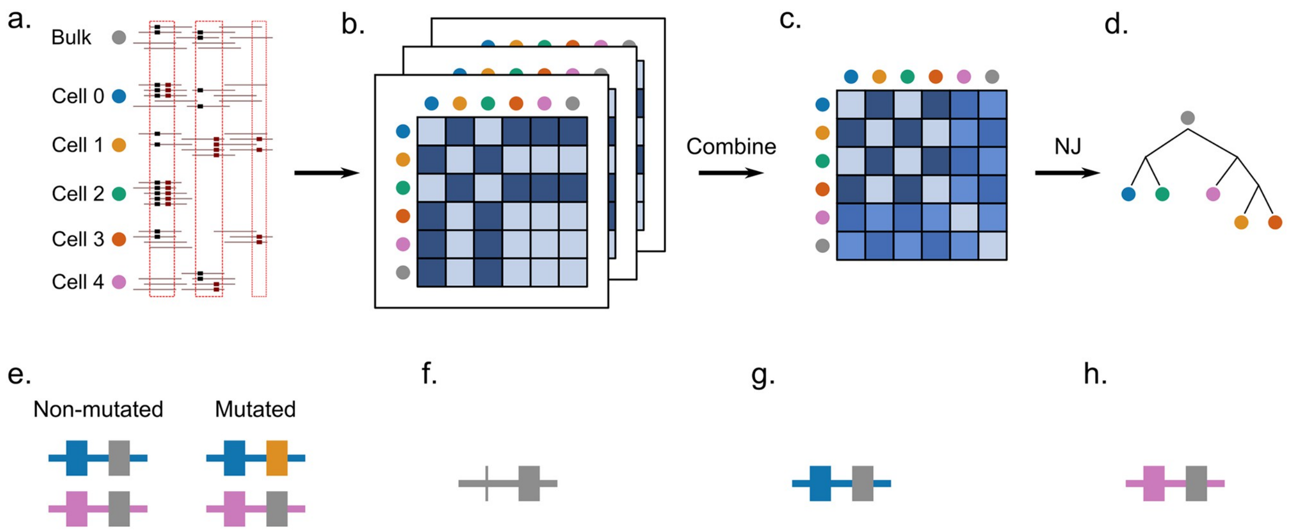

Fig 1. The workflow of Scuphr and the illustration of read-phasing. a-d: The workflow of Scuphr. **a**: The sites are selected for analysis from bulk and single-cell DNA sequencing data. **b**: The distance matrix of each chosen site is calculated independently. **c**: The distance matrices are combined. **d**: The cell lineage tree is constructed from the final distance matrix using the NJ algorithm. **e-h**: Illustration of read-phasing. The first base pair is the gSNV locus and the second base pair is the candidate site. The gSNV nucleotides are shown in blue and pink colors. The reference and the mutation nucleotides are shown in gray and yellow, respectively. **e**: The non-mutated and mutated genomes are displayed. The mutation is associated with the blue allele (blue gSNV nucleotide). **f**: An example read covering both sites, but the gSNV information is disregarded as in many methods of SNV calling and tree reconstruction. The reference nucleotide is observed at the candidate site. The read might belong to either the non-mutated or the mutated genome. **g**: An example read with available gSNV site information. We can *phase* the read (identify which allele it originates from). The read must come from the blue allele; in this case, the read comes from the non-mutated genome. **h**: An example read with available gSNV site information. The read must come from the pink allele; in this case, the read might belong to either the non-mutated or mutated genome.

https://doi.org/10.1371/journal.pcbi.1012094.g001

nucleotide substitutions caused by amplification. Therefore, Scuphr relies on a probabilistic model of read-phasing and these two error sources. Read-phasing is a technique applied to identify which allele the read comes from, which is used to distinguish if the read could be from a mutated or a non-mutated segment. We first describe our model without read-phasing and then introduce the details of the read-phasing.

The amplification process is modeled as a generalized Pólya urn process, in which a drawn ball with an error probability is replaced by one of the same color and one of another, in contrast to the ubiquitous replacement by two of the same color in the Pólya urn. The observed Phred scores define the base-calling error probabilities. The model also contains a probability for the ADO events. Another vital part of Scuphr is a dynamic programming-based inference algorithm that, based on the error model, computes the probability that two cells have different genotypes at any investigated potential mutation site.

Scuphr processes the scDNA-seq data using a *site selection* method that identifies the candidate sites that will be analyzed using the probabilistic model and contribute to the distance. The distance is obtained by combining the probabilities of different genotypes across the selected sites for each pair of cells. Finally, this distance is used as an input to a distance-based phylogenetic method, the neighbor-joining (NJ) algorithm [42].

The summary of the Scuphr workflow is shown in Fig 1. The input to Scuphr consists of bulk and single-cell DNA reads. First, candidate mutation sites are detected using the site selection method, Fig 1a. These candidate mutation sites can consist of a single base pair, like in most state-of-the-art methods, or two base pairs where the candidate mutation site is accompanied by a nearby germline SNV (gSNV). We call these site types *singleton sites* and *paired sites*, respectively. These site types will be referred to as *sites* throughout the paper.

Second, site-associated distance matrices are calculated in parallel for each selected site, Fig 1b. Third, a single distance matrix is obtained by combining the site-associated distance matrices, Fig 1c. Finally, the cell lineage tree is reconstructed by applying the NJ algorithm to the final distance matrix, Fig 1d. In addition, the site-associated distance matrices can be sampled with replacement several times to obtain bootstrap lineage tree samples, which could be used to get consensus trees and edge supports.

The read-phasing assists in identifying missing data and separating somatic mutations and errors using patterns of co-occurrence of the nucleotides at the gSNV loci and the candidate sites. Fig 1e shows an example of a non-mutated and a mutated cell's genome. Each cell has two alleles; one maternal and one paternal. The first locus is a gSNV, where the nucleotides at the first and second allele differ. The second locus is the candidate site, where the non-mutated cell has the reference nucleotide in both alleles, and the mutated cell has a reference and an alternate nucleotide. The non-mutated and mutated cells have the same second allele, and their difference is due to the mutation located at the first allele. The mutation is associated with the blue gSNV nucleotide. In Fig 1f, a read with the candidate site is shown. One cannot decide if this read comes from a non-mutated or a mutated genome since no information is available for the gSNV locus. Therefore, it cannot be attributed to any of the alleles. In Fig 1g, a read from the site pair is observed. Since both nucleotides are observed, and the blue gSNV nucleotide accompanies the candidate reference nucleotide, one can conclude that the read comes from the non-mutated genome. However, in Fig 1h, the reference nucleotide is accompanied by the pink gSNV nucleotide, which could come from either of the genomes.

Results

The two state-of-the-art methods, SCIΦ [37] and Phylovar [41], can exploit amplified, diploid scDNA-seq data. We compared the performance of Scuphr with SCIΦ in terms of cell lineage tree reconstruction accuracy and runtime in synthetic data experiments. For the biological data experiment, we compared the performance of Scuphr to both SCIΦ and Phylovar.

Synthetic dataset experiments

In order to compare the performances of the methods in a controlled way, we synthetically generated datasets. We created the ground truth cell lineage trees, and assigned mutations to the edges of the tree; all cells under the edge inherit the mutation. We generated the genomes of the cells and simulated the amplification and read sequencing processes. We accounted for various amplification, allelic dropout, and sequencing errors, as well as different levels of gSNVs within the genome. The details of the synthetic data generation process are presented in Methods. We used the synthetic datasets to compare the methods in two ways; the lineage tree reconstruction accuracy and runtime.

Accuracy evaluation on synthetic datasets. To evaluate the performance of cell lineage tree reconstruction, we compared the topologies of the trees inferred by Scuphr and SCIΦ with the ground truth cell lineage tree. We use a similarity measure defined as one minus the normalized Robinson-Foulds (RF) distance. The similarity score is in [0, 1], where 1 means the tree topologies are the same. We investigated the accuracy across several combinations of AE rates, ADO rates, frequencies of loci with paired sites, and the number of cells. We investigated two AE rates: 10^{-5}, see Fig 2 and 10^{-3}, see Fig 3. First, for each choice of other parameters, the frequencies of loci with paired sites considered were 0.001, 0.01, 0.1, and 1. Second, for each choice of the other parameters, the ADO probabilities considered were 0, 0.1, and 0.2. Third, all parameter configurations were investigated for inputs having 10, 20, and 50 cells.

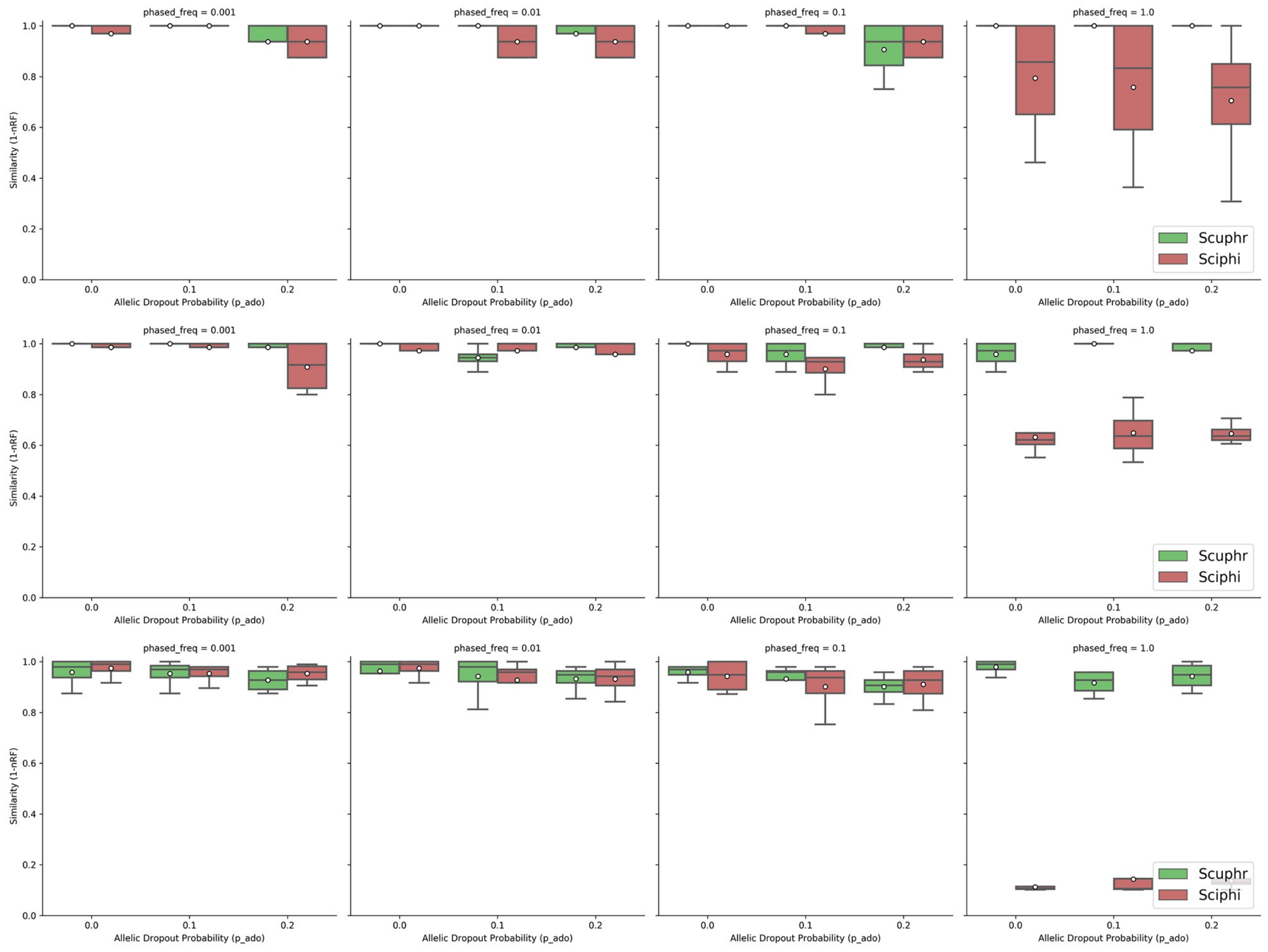

Fig 2. Similarities between the true and inferred trees for low amplification errors. Top row: Results for 10 cells. **Center row**: Results for 20 cells. **Bottom row**: Results for 50 cells.

https://doi.org/10.1371/journal.pcbi.1012094.g002

Both methods perform well for the lower error rate 10^{-5}, Fig 2. However, with a few exceptions, Scuphr has a higher mean accuracy, and other quantiles are higher for Scuphr than for SCIΦ. This trend is accentuated for the case where the frequency of loci with paired sites is 1.

For the higher error rate of 10^{-3}, Fig 3 shows a readily noticeable difference between the two methods. The average accuracy for Scuphr is always better than that of SCIΦ, and other quantiles are, in almost all cases, higher for Scuphr than for SCIΦ. In particular, when all of the loci are paired, Scuphr exploits the paired sites for read-phasing, but SCIΦ has more or less the same average accuracy for lower frequencies of paired sites. In this case, the average accuracy of Scuphr is almost twice as high as that of SCIΦ.

Interestingly, when SCIΦ is provided with the sites selected by our candidate site selection method, its accuracy is improved in several cases. However, its accuracy is also, in many instances, decreased. Moreover, for the biological data, it is, due to the number of sites, that it takes an even longer time to run SCIΦ with the sites selected by our site selection method. Therefore, in Figs 2 and 3, we presented the results of the entire SCIΦ method, as described in [37]. The accuracy obtained when SCIΦ is provided with the site selected by our candidate site selection method is described in the S1 Appendix.

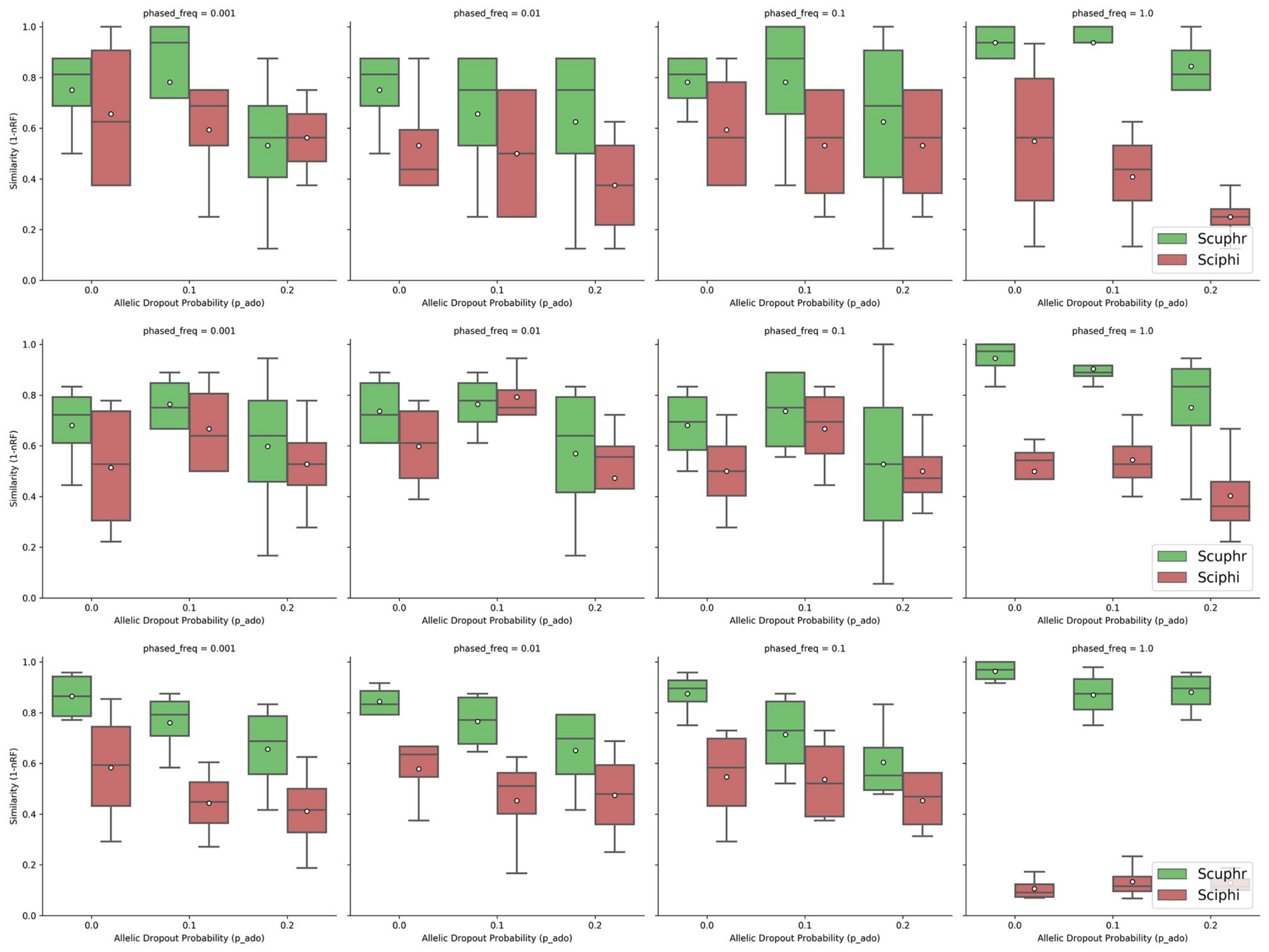

Fig 3. Similarities between the true and inferred trees for high amplification errors. Top row: Results for 10 cells. **Center row**: Results for 20 cells. **Bottom row**: Results for 50 cells.

https://doi.org/10.1371/journal.pcbi.1012094.g003

Runtime analysis on synthetic datasets. In addition to the lineage tree reconstruction accuracy, we also compared the wall-clock runtime of Scuphr to that of SCIΦ. All runtime experiments were performed on a single cluster node with 32 CPU cores, and each configuration was repeated ten times. As Scuphr can be used with default and estimated parameters, the runtime analysis for the parameter estimation step was performed separately, and the results are presented in the S1 Appendix. We ran both methods with the same sites to compare the runtimes. The time used to obtain these sites is excluded from the runtimes reported here. We run SCIΦ with both single and multiple cores. The runtimes were very similar across the number of cores, and in this section, the single-core runtimes are presented for SCIΦ. For the multiple core runs, we direct the reader to the S1 Appendix.

Fig 4 shows how the runtime changes across fractions of singleton sites and the number of cores. The left, center, and right subfigures display the runtime analysis for 10, 20, and 50 cells, respectively. Since each distance matrix is calculated independently, the main part of our proposed algorithm is embarrassingly parallel, i.e., the wall-clock runtime scales linearly with the number of available cores, as seen in the figure. Additionally, the algorithm's runtime is linear in the number of sites. For 10 and 20 cells, our software infers the lineage tree faster than

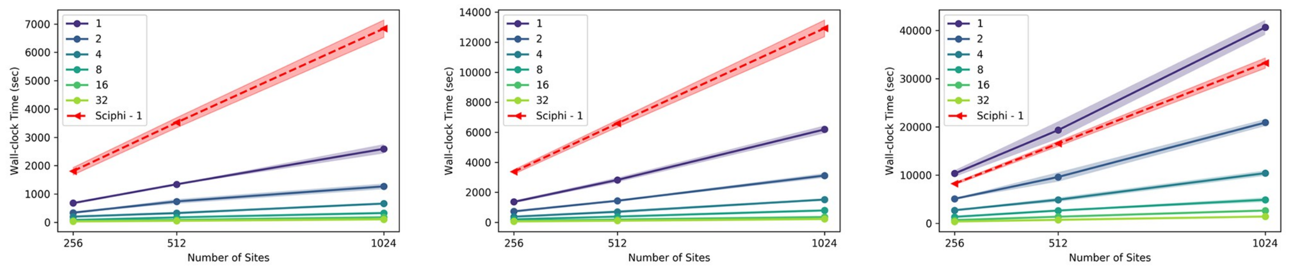

Fig 4. Runtime comparison for singleton sites. The x-axis is the number of sites, and the y-axis is the wall-clock time in seconds. The red dashed line is the runtime of SCIΦ using a single core. The remaining lines are the runtimes of the proposed method for varying numbers of cores. The left, center, and right subplots are the results for 10, 20, and 50 cells datasets, respectively.

https://doi.org/10.1371/journal.pcbi.1012094.g004

SCIΦ. Our software is faster for singleton sites and 50 cells when at least two cores are used for computation.

The wall-clock runtime comparisons for paired sites are shown in Fig 5. Also, in this case, the left, center, and right subfigures correspond to 10, 20, and 50 cells, respectively. Our paired site analysis is slower than the singleton site analysis due to the number of fragment types considered. Nonetheless, the method has the same scalability trend as in the singleton site experiments. Our method is faster than SCIΦ for the 10 cells case when at least eight cores are used and for the 20 and 50 cells datasets when at least 16 cores are used.

These runtime analyses were performed on a single cluster node. Nevertheless, one can use multiple nodes to compute the distance matrices without communication overhead. Consequently, Scuphr achieves a linear speedup in the total number of available cores on a cluster. The final step of Scuphr, the lineage tree reconstruction, runs on a single core; however, this step is very efficient and does not change the overall asymptotic runtime.

Biological data analysis

In this section, the accuracy of Scuphr SCIΦ, and Phylovar are compared using a fibroblast dataset previously used in [30]. The dataset used in this study is a slightly modified version of the dataset in [30]. For details, see S1 Appendix. The dataset consists of scDNA-seq data for 18 cells with a recent common origin and a known lineage tree topology. This single-cell DNA data has been obtained by amplifying the DNA using MALBAC before the sequencing. These cells are so closely related that very few mutations distinguish them; hence, it is very hard to

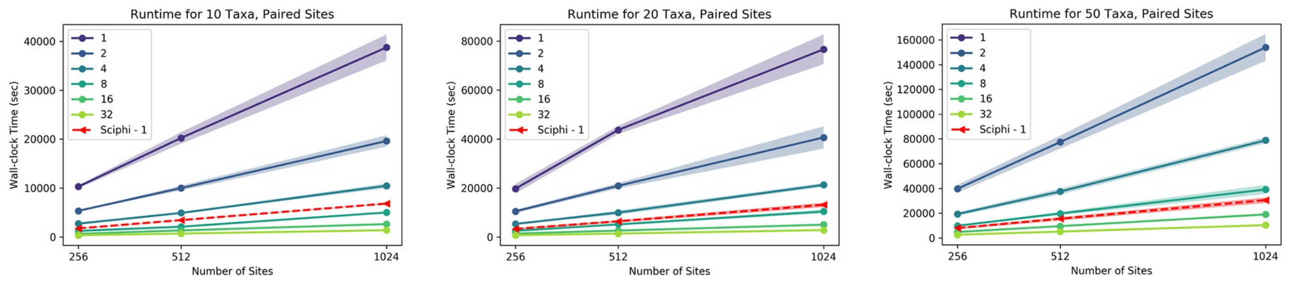

Fig 5. Runtime comparison for paired sites. The x-axis is the number of sites, and the y-axis is the wall-clock time in seconds. The red dashed line is the runtime of SCIΦ using a single core. The remaining lines are the runtimes of the proposed method for varying numbers of cores. The left, center, and right subplots are the results for 10, 20, and 50 cells datasets, respectively.

https://doi.org/10.1371/journal.pcbi.1012094.g005

reconstruct the true lineage tree topology. The cells belong to two main monophyletic groups, one containing cells 0–11 and one containing cells 12–17. The dataset also includes a bulk DNA sample from the donor, which can be used as an outgroup.

The data were preprocessed using the pipeline described in S1 Appendix, and the sites of interest were identified as described in Methods. More than 3 million sites were selected for analysis; the details are presented in S1 Appendix.

Since the fibroblast dataset is so hard that a reconstruction method would at most identify the two main monophyletic groups correctly, we devised a test based on bootstrapping, using the transfer bootstrap expectation (TBE) [43] edge supports. 100 bootstrap lineage trees were constructed by bootstrapping sites (hence, bootstrapping the distance matrices). The TBE supports of the bootstrapped trees on the true lineage tree topology were computed with the Booster software [43]. The TBE support of a branch *b* is in the [0, 1] range where "0 means that the bootstrap trees contain the edge *b* in a random fashion and 1 means that *b* appears in all bootstrap trees" [43].

Scuphr separates the two main monophyletic groups with very high TBE support, 0.8, Fig 6a. Also, all branches within two smaller monophyletic groups, cells 4-5 and cells 10-11, respectively, are correctly inferred in all bootstrap rounds.

To compare accuracy, we also applied SCIΦ to the fibroblast data (Fig 6b). Due to the input format requirement (bulk and single-cell sequencing data in Mpileup format), we could not

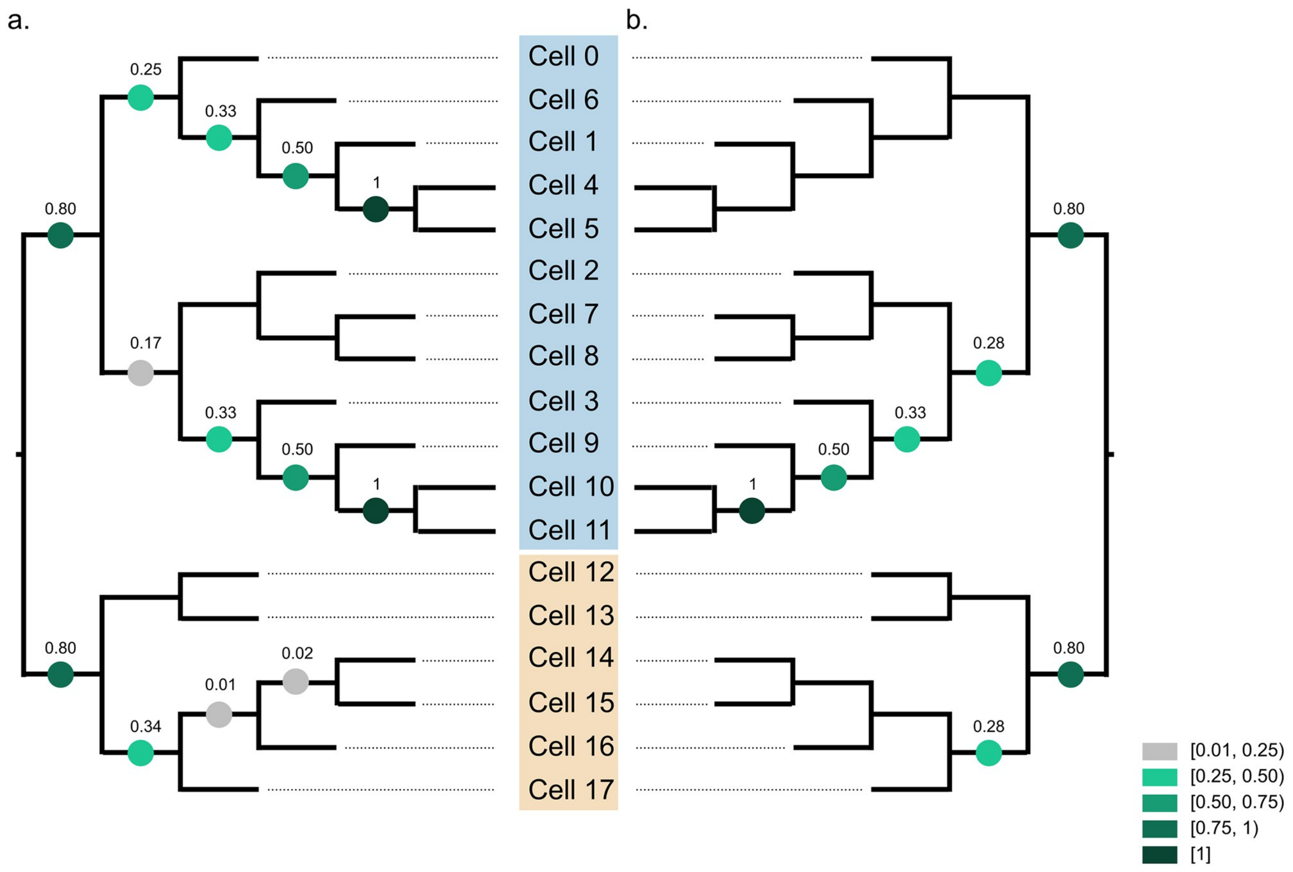

Fig 6. The TBE supports of bootstrap trees are projected onto true lineage tree topology. The clones are marked with blue and beige colors. **a**: TBE supports of Scuphr. **b**: TBE supports of SCIΦ.

https://doi.org/10.1371/journal.pcbi.1012094.g006

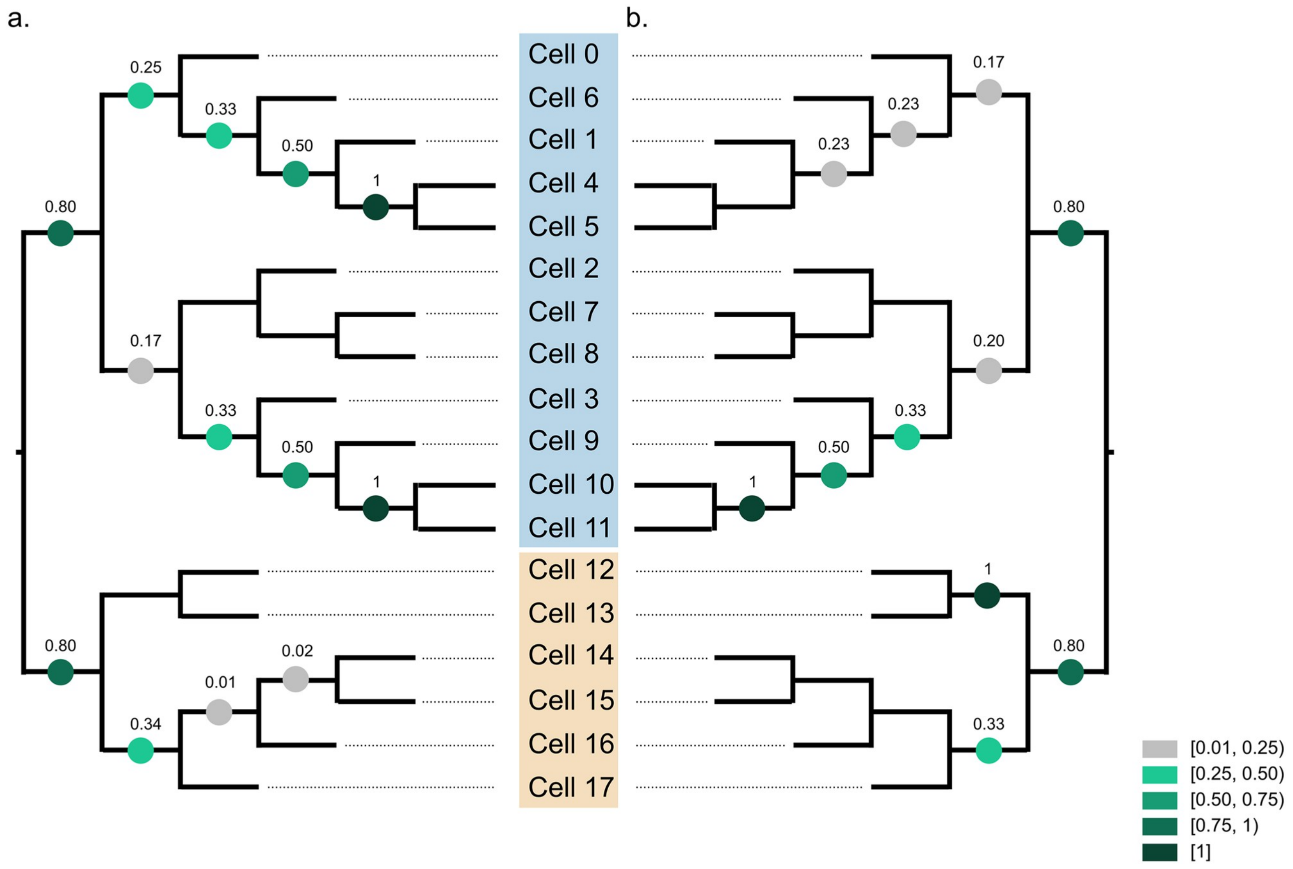

Fig 7. The TBE supports of bootstrap trees are projected onto true lineage tree topology. The clones are marked with blue and beige colors. **a**: TBE supports of Scuphr. **b**: TBE supports of Phylovar.

https://doi.org/10.1371/journal.pcbi.1012094.g007

run SCIΦ on the whole genome in a single run. Instead, we applied SCIΦ independently to each chromosome, see S1 Appendix for details. We sampled 100 bootstrap trees from the lineage trees reported by SCIΦ for the chromosomes; the trees were weighted by the number of identified mutations per chromosome. The TBE supports are evaluated on the true lineage tree topology (Fig 6b). SCIΦ got the same TBE supports for a subset of branches (separation of two clones and the subclone consisting of cells 3, 9, 10, and 11). SCIΦ reported higher support for a single branch, 0.28, than Scuphr, 0.17. For the rest of the branches, Scuphr reported higher support or equal support to that reported by SCIΦ. In the monophyletic group consisting of cells 0, 1, 4, 5, and 6, in contrast to SCIΦ, Scuphr inferred substantial subclonal structure.

Finally, we compared the performance of Scuphr against Phylovar on the biological data in Fig 7. Phylovar uses the same candidate site selection scheme as SCIΦ, hence using the same sites for analysis. Following the software documentation, we extracted the sites picked by SCIΦ and created a new Mpileup file, which enabled us to run Phylovar on a cluster node. We ran Phylovar 5 times, each with 20 hill-climbing chains on a cluster node of 32 cores, and obtained 100 trees for analysis. For experimental details, see S1 Appendix.

As with the other methods, Phylovar successfully separated two clones. Although Phylovar was able to infer the relation of subclone consisting of cells 0, 1, 4, 5, and 6 better than SCIΦ; its support scores are lower than Scuphr, Fig 7b. Moreover, Phylovar was unable to identify

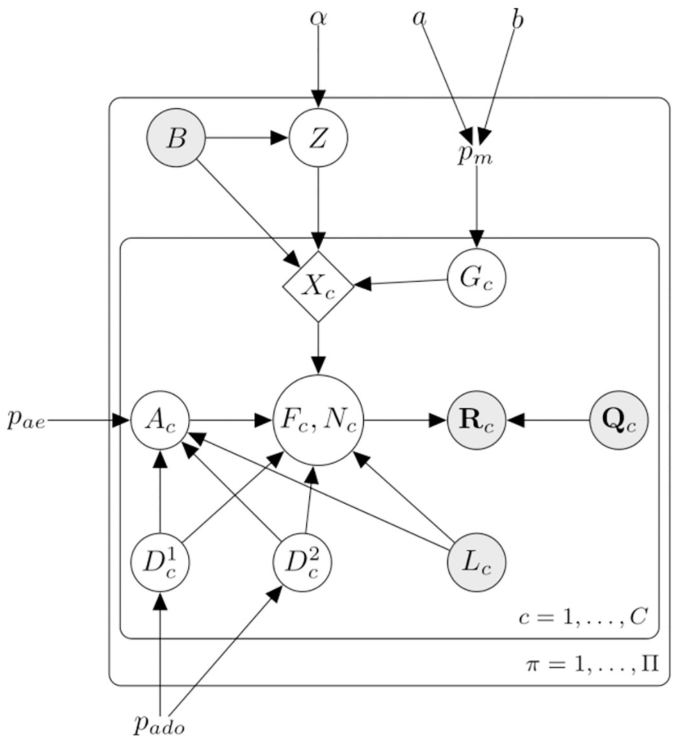

Fig 8. The graphical model of Scuphr. The shaded nodes are the observed random variables. The site superscript π is omitted for brevity. α, a, and b are the model hyperparameters. p_m, p_{ae}, p_{ado} are the model parameters and correspond to the mutation, amplification error, and allelic dropout probabilities respectively. B is the bulk genotype, Z is the common mutation type. Each single-cell $c \in [C]$ has a mutation status G_c. B, Z and G_c deterministically define a cell's genotype; X_c. D_c^1 and D_c^2 indicate the dropout status of each allele, and A_c indicates the number of amplification errors. F_c and N_c are the fragment types and their corresponding counts. $\mathbf{R}_c$ and $\mathbf{Q}_c$ are the reads and their Phred quality scores, and L_c is the total number of reads of the cell.

https://doi.org/10.1371/journal.pcbi.1012094.g008

the sibling relation of cells 4 and 5, which was supported by all Scuphr trees. Phylovar obtained similar support scores for the remainder of the blue clone as the other methods. In the beige clone, Phylovar had high support for a sibling relation (cells 12 and 13), which was not supported by either Scuphr or SCIΦ.

Materials and methods

In this section, we describe the proposed model step-by-step. First, we present the probabilistic graphical model of Scuphr in detail and outline important components and formulas. Second, we describe how to reconstruct the cell lineage tree from the outcome of the first part. Third, we describe the candidate loci selection criteria in detail. Fourth, we show how the model parameters are estimated. Fifth, the simulated data generation procedure is presented. Finally, two accuracy metrics, the similarity score, and the TBE support, used in the study are described.

The probabilistic model

First, we introduce some important concepts. Recall that, as most state-of-the-art methods do, a set of candidate mutation loci is used for analysis. We call a candidate locus that covers a single base pair a *singleton site*. Moreover, Scuphr can facilitate gSNVs near candidate loci and do read-phasing. A candidate mutation locus paired with a gSNV site is called a *paired site*. We will refer to all site types as *sites* throughout the Methods section for brevity. Unless otherwise stated, the model descriptions hold for all site types. Let Π be the set of sites selected for analysis and C be the number of single cells.

The probabilistic graphical model. Fig 8 shows the probabilistic graphical model of Scuphr at $\pi \in \Pi$. a, b, and α are the model hyperparameters. p_{ado}, p_{ae}, and p_m are the model's parameters and correspond to the ADO, AE, and mutation probabilities. A set of reads, their corresponding base-calling error probabilities, and the coverage of each cell c are observed and represented by $\mathbf{R}_c$, $\mathbf{Q}_c$, and L_c. Moreover, the bulk genotype, B, is observed. The single-cell mutation status is represented with G_c; G_c, B, and the common mutation type random variable, Z, define the single-cell genotype, X_c. The aforementioned scDNA-seq specific challenges are modeled with D_c^1, D_c^2, and A_c random variables; D_c^1 and D_c^2 model the ADO events of each allele, and A_c represents the number of errors that have happened during amplification. The fragment types generated at the end of the amplification process and their counts are expressed by F_c and N_c, respectively.

The graphical model consists of cell lineage, DNA amplification, and read sequencing.

- **Cell lineage**: At π, all the non-mutated cells have the bulk genotype, and all the mutated cells must share the same mutation type under the ISA. This shared mutation type is modeled with a Dirichlet-Categorical distribution with the hyperparameter α. The cells' mutation statuses are modeled with i.i.d. Bernoulli random variables with a mutation probability, p_m. The mutation probability has a Beta prior distribution with hyperparameters a and b. The mutation status random variable, bulk, and mutation type variables define the genotype of the cell, X_c.
- **DNA amplification**: Here, we model the ADO events of each allele with Bernoulli random variables, with the same ADO probability p_{ado}. The number of AEs that have happened during the amplification is modeled with a Binomial random variable with probability p_{ae}, and its number of trials depends on the ADO random variables and the observed read coverage. These ADO and AE random variables, the observed read coverage, and the single-cell genotype form the amplified fragments, F_c and their corresponding counts, N_c.
- **Read sequencing**: Finally, the amplified fragments are sequenced and create observed reads. Since the read sequencing is an erroneous process, the observed Phred scores are used to obtain the base-calling error probabilities, Q_c, and the uncertainty of read sequencing is modeled.

We briefly discussed the graphical model. More details of its components are presented in the following parts of this section. Fig 9 illustrates the cell lineage, DNA amplification, and sequencing steps described above. For simplicity, the site superscript π is omitted.

Distance matrix. For each selected site, $\pi \in \Pi$, we construct a symmetric nonnegative $(C+1) \times (C+1)$ distance matrix M^π. $M_{c,c'}$ is the distance between single-cells c and c', computed by

$$M_{c,c'} = \frac{\sum_\pi M^\pi_{c,c'} \times \mathbb{1}[L^\pi_c > 0, L^\pi_{c'} > 0]}{\sum_\pi \mathbb{1}[L^\pi_c > 0, L^\pi_{c'} > 0]}, \ \forall (c, c') \in [C], \tag{1}$$

where $\mathbb{1}$ is the indicator function, and L^π_c is the coverage of c at π. Only the sites where both cells have coverage are considered during distance computation. The matrix's last row and

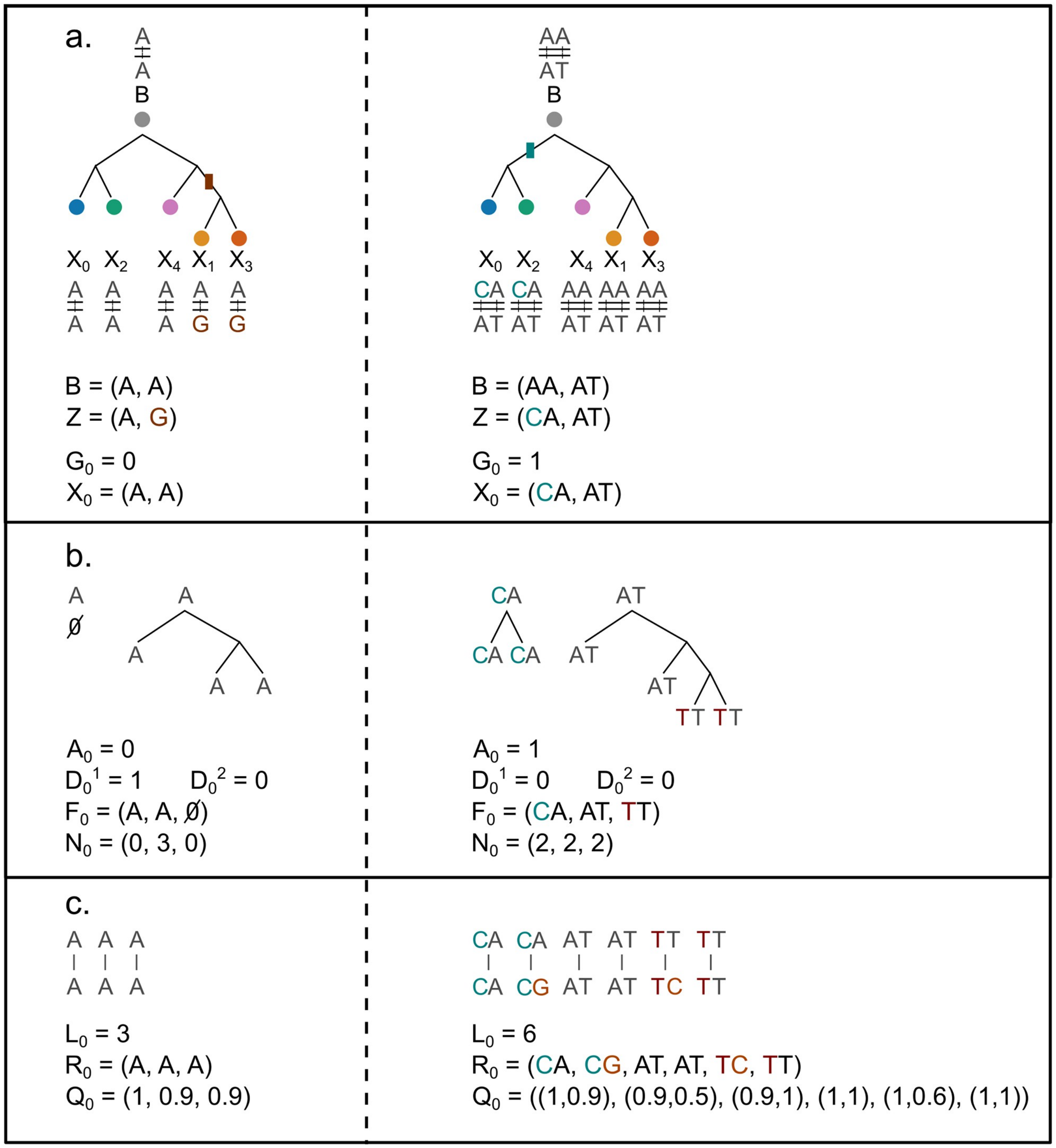

Fig 9. Illustration of the random variables for a singleton site on the left column and a paired site on the right column. a: The cell lineage tree. Bulk, common mutation type, and single-cell genotypes for the sites of interest are shown. Mutations and the branch they originate from are colored differently. **b**: The DNA amplification step for cell 0 is illustrated. AE and ADO event indicators are specified. The corresponding fragment types and the counts are written. **c**: The sequencing step is illustrated; the reads and the base-calling error probabilities are generated from the fragments. Sequencing errors are colored differently.

https://doi.org/10.1371/journal.pcbi.1012094.g009

column are the distances between the cell and the non-mutated bulk, b, computed by

$$M_{c,b} = \frac{\sum_\pi M^\pi_{c,b} \times \mathbb{1}[L^\pi_c > 0]}{\sum_\pi \mathbb{1}[L^\pi_c > 0]}, \ \forall c \in [C]. \tag{2}$$

Distance between two single-cells at π. The binary random variable G^π_c represents the mutation status of single-cell c at π: $G^\pi_c = 0$ if the cell is not mutated and $G^\pi_c = 1$ if the cell is mutated. The distance between two single-cells at π is

$$\begin{aligned} M^\pi_{c,c'} &= P(G_c = 0, G_{c'} = 1 | B, \mathbf{R}_{1:C}, \mathbf{Q}_{1:C}, L_{1:C}, \Theta) \\ &\quad + P(G_c = 1, G_{c'} = 0 | B, \mathbf{R}_{1:C}, \mathbf{Q}_{1:C}, L_{1:C}, \Theta), \end{aligned} \tag{3}$$

where B is the bulk genotype, $\Theta = \{\alpha, a, b, p_{ado}, p_{ae}\}$ is the set of model parameters and hyper-parameters, and $\mathbf{R}_{1:C}$, $\mathbf{Q}_{1:C}$, and $L_{1:C}$ are the observed reads, base-calling error probabilities, and read coverages of single-cells. The distance to non-mutated bulk is simply

$$M_{c,b} = \sum_{i=0}^{1} P(G_c = 1, G_{c'} = i | B, \mathbf{R}_{1:C}, \mathbf{Q}_{1:C}, L_{1:C}, \Theta).$$

On the random variables of the right-hand side of the above equation, we omit the π super-script and use the same convention in the following parts.

The mutation status probability of two cells at a site satisfies

$$P(G_c, G_{c'} | B, \mathbf{R}_{1:C}, \mathbf{Q}_{1:C}, L_{1:C}, \Theta) \propto P(G_c, G_{c'}, \mathbf{R}_{1:C} | B, \mathbf{Q}_{1:C}, L_{1:C}, \Theta).$$

$P(G_c, G_{c'}, \mathbf{R}_{1:C} | B, \mathbf{Q}_{1:C}, L_{1:C}, \Theta)$ is computed through series of marginalizations done on the latent variables defined in the graphical model, shown in Eq 4. In order to describe the process clearly, we introduce the marginalizations one at a time in the following subsections.

$$\begin{aligned} &P(G_c, G_{c'}, \mathbf{R}_{1:C} | B, \mathbf{Q}_{1:C}, L_{1:C}, \Theta) \\ &= \sum_Z \sum_{G_{1:C\,\{c,c'\}}} \int_{p_m} \sum_{F_{1:C}, N_{1:C}} \sum_{D^1_{1:C}} \sum_{D^2_{1:C}} \sum_{A_{1:C}} \\ &\qquad P(G_c, G_{c'}, \mathbf{R}_{1:C}, Z, G_{1:C\setminus\{c,c'\}}, p_m, F_{1:C}, N_{1:C}, D^1_{1:C}, D^2_{1:C}, A_{1:C} | B, \mathbf{Q}_{1:C}, L_{1:C}, \Theta) dp_m \end{aligned} \tag{4}$$

Lineage model. The ISA implies that (i) a site can be mutated at most once, and (ii) all the mutated cells at the loci share the same mutation. We marginalize the mutation types

$$\begin{aligned} &P(G_c, G_{c'}, \mathbf{R}_{1:C} | B, \mathbf{Q}_{1:C}, L_{1:C}, \Theta) \\ &\qquad = \sum_z P(Z = z | B, \alpha) \ P(G_c, G_{c'}, \mathbf{R}_{1:C} | Z = z, B, \mathbf{Q}_{1:C}, L_{1:C}, \Theta), \end{aligned}$$

where Z is the mutation type random variable and follows the Dirichlet-Categorical distribution as described previously;

$$P(Z = z | B, \alpha) = \frac{\mathrm{B}(\sum_{k=1}^{K} \alpha_k, 1)}{\mathrm{B}(\alpha_z, 1)}, \tag{5}$$

where K is the number of possible mutation genotypes, α is the concentration parameter, and B is the Beta function. B stands for the bulk genotype whereas B is the Beta function. Z differs from the non-mutated bulk genotype by a single nucleotide, e.g., we may have $B = (A, A)$ and $Z = (A, G)$ for a singleton site or $B = (AA, AT)$ and $Z = (AA, GT)$ for a paired site. See S1 Appendix for details.

Marginalization of other single-cells. Using the notation $G_{1:C\setminus\{c,c'\}}$ for the mutation status random variables of all cells except c and c', the joint distribution of the reads and the mutation statuses of the single-cells c and c' can be expressed as

$$\begin{aligned} &P(G_c, G_{c'}, \mathbf{R}_{1:C}|Z, B, \mathbf{Q}_{1:C}, L_{1:C}, \Theta) \\ &\quad = \sum_{G_{1:C\ \{c,c'\}}} P(G_{1:C}|a, b)\ P(\mathbf{R}_{1:C}|G_{1:C}, Z, B, \mathbf{Q}_{1:C}, L_{1:C}, p_{ae}, p_{ado}) \\ &\quad = \sum_{m=0}^{C} \sum_{\substack{G_{1:C\setminus\{c,c'\}}: \\ \sum_i G_i = m}} P(G_{1:C}|a, b)\ P(\mathbf{R}_{1:C}|G_{1:C}, Z, B, \mathbf{Q}_{1:C}, L_{1:C}, p_{ae}, p_{ado}), \end{aligned}$$

where $m = \sum_{i=1}^{C} G_i \in [0, C]$ is the number of mutated cells at the site π. The summation over mutation counts and mutation statuses in the above equation can be computed efficiently using dynamic programming.

As mentioned earlier, we assign Beta prior distribution on the mutation probability, p_m, with the hyperparameters a and b. Given p_m, the mutation statuses of all single-cells are conditionally independent and are i.i.d. Bernoulli random variables. The joint distribution of single-cell mutation statuses is

$$\begin{aligned} P(G_{1:C}|a, b) &= \int_{p_m} P(G_{1:C}|p_m, a, b)\ P(p_m|a, b)\ dp_m \\ &= \frac{\mathrm{B}(m + a, C - m + b)}{\mathrm{B}(a, b)}. \end{aligned}$$

The derivation is presented in S1 Appendix.

Genotypes of single-cells. We define the auxiliary random variables, $X_{1:C}$, to denote single-cell genotypes. The genotype of a single-cell c is

$$X_c = \begin{cases} B, & \text{if } G_c = 0 \\ Z, & \text{if } G_c = 1. \end{cases} \tag{10}$$

The values of $X_{1:C}$ are deterministic functions of $G_{1:C}$, Z, and B. From now on, we will use $X_{1:C}$ notation instead of $\{G_{1:C}, Z, B\}$, e.g.,

$$P(\mathbf{R}_{1:C}|G_{1:C}, Z, B, \mathbf{Q}_{1:C}, L_{1:C}, p_{ae}, p_{ado}) = P(\mathbf{R}_{1:C}|X_{1:C}, \mathbf{Q}_{1:C}, L_{1:C}, p_{ae}, p_{ado}).$$

Conditional independence of single-cells. Given the genotypes of single cells, the amplification, and the allelic dropout probabilities, the likelihoods of reads are conditionally independent. The read likelihood is factorized by

$$P(\mathbf{R}_{1:C}|X_{1:C}, \mathbf{Q}_{1:C}, L_{1:C}, p_{ae}, p_{ado}) = \prod_{c=1}^{C} P(\mathbf{R}_c|X_c, \mathbf{Q}_c, L_c, p_{ae}, p_{ado}).$$

Introduction of fragments. We introduce the *fragments* created during the DNA amplification. The fragment types, F_c, and their counts, N_c, are marginalized as follows;

$$\begin{aligned} &P(\mathbf{R}_c|X_c, \mathbf{Q}_c, L_c, p_{ae}, p_{ado}) \\ &\quad = \sum_{F_c, N_c} P(\mathbf{R}_c|F_c, N_c, \mathbf{Q}_c)\ P(F_c, N_c|X_c, L_c, p_{ae}, p_{ado}). \end{aligned}$$

Amplification model. The DNA amplification is modeled with a generalized Pólya urn model. Two ADO events determine the initial state of the urn. These ADO events are modeled by two Bernoulli random variables with the same ADO probability, p_{ado};

$$\begin{aligned} &P(F_c, N_c|X_c, L_c, p_{ae}, p_{ado}) \\ &\quad = \sum_{D_c^1=0}^{1}\sum_{D_c^2=0}^{1} P(D_c^1, D_c^2|p_{ado})\ P(F_c, N_c|D_c^1, D_c^2, X_c, L_c, p_{ae}), \end{aligned}$$

and

$$P(D_c^1, D_c^2|p_{ado}) \quad = p_{ado}^{D_c^1+D_c^2}\ (1-p_{ado})^{2-(D_c^1+D_c^2)}.$$

In the case of no ADO events, the process starts with one copy of each allele. The process starts with the other allele if there is one ADO event.

One can describe the urn process as follows; the urn is initialized with one or two colored balls. At each step, a ball is drawn from the urn, a copy of the ball is made, and both the original and the copy is put back into the urn. The outcome of this process can be represented by one or two lineage trees where the roots of the trees are the original copies of the alleles. We will refer to these trees as *amplification trees*. Given the initial state and the final number of balls in the urn, which is observed as the read coverages, the total number of edges in amplification trees is (an additional incoming edge to the root is introduced to account for subsampling)

$$E_{D_c^1, D_c^2}^{L_c} = \begin{cases} 2L_c - 2, & \text{if } D_c^1 = 0, D_c^2 = 0 \\ 2L_c - 1, & \text{if } D_c^1 = 0, D_c^2 = 1 \\ 2L_c - 1, & \text{if } D_c^1 = 1, D_c^2 = 0 \\ 0, & \text{if } D_c^1 = 1, D_c^2 = 1. \end{cases}$$

DNA amplification sometimes replaces a nucleotide; therefore, occasionally, the copy of a ball has a different color than the original. Let A_c be the random variable describing the number of AEs has happened during the DNA amplification,

$$\begin{aligned} &P(F_c, N_c|D_c^1, D_c^2, X_c, L_c, p_{ae}) \\ &\quad = \sum_{A_c} P(A_c|D_c^1, D_c^2, L_c, p_{ae}) \sum_{F_c, N_c} P(F_c, N_c|D_c^1, D_c^2, A_c, X_c, L_c). \end{aligned}$$

The probability of the number of AEs is a Binomial distribution over the edges of the amplification trees

$$P(A_c|D_c^1, D_c^2, L_c, p_{ae}) = \binom{E_{D_c^1, D_c^2}^{L_c}}{A_c} p_{ae}^{A_c}\ (1-p_{ae})^{E_{D_c^1, D_c^2}^{L_c} - A_c},$$

where p_{ae} is the probability of an AE happening on an edge. In practice, p_{ae} is very small (e.g., $[10^{-6}, 3\times10^{-4}]$ [19, 30]); hence we neglect the cases where $A_c > 1$.

Marginalizing amplification trees. Let the fragment types and counts be $F_c = (F_c^1, F_c^2, F_c^3)$ and $N_c = (N_c^1, N_c^2, N_c^3)$, respectively. Let $d(F_i||F_j)$ be the function that computes the Hamming distance between two fragment types, F_i and F_j.

The first two elements of F_c are the cell genotype, $(F_c^1, F_c^2) = X_c$, and the third element is the fragment type caused by an AE. In the case of no AE, $F_c^3 = \emptyset$. In the case of one AE, F_c^3 must differ a single nucleotide from its originating fragment, either $d(F_c^3||F_c^1) = 1$ or $d(F_c^3||F_c^2) = 1$.

Similar to the F_c tuple, N_c^1, N_c^2, and N_c^3 are the numbers of fragments of the first allele, second allele, and fragments carrying a nucleotide introduced by the AE. The total number of fragments is $L_c = N_c^1 + N_c^2 + N_c^3$. In case of an AE event, $N_c^3 > 0$; otherwise, N_c^3 is zero. Finally, the first two elements of fragment counts must satisfy the ADO events, i.e., $N_c^1 = 0$ if $D_c^1 = 1$ and $N_c^2 = 0$ if $D_c^2 = 1$.

With the above conditions satisfied, given the cell genotype, read coverage, ADO, and AE events, the probability of a fragment type and count pair is a product that contains up to three important factors. The first factor regards dividing L_c fragments into one or two amplification trees. There is a single way to partition if there is an ADO event, e.g., $(0, L_c)$ if the first allele is dropped out or $(L_c, 0)$ if the second allele is dropped. Otherwise, due to the Pólya urn, the number of reads follows a Beta-Binomial distribution, and each partition, $\{(1, L_c - 1), (2, L_c - 2), \ldots, (L_c - 1, 1)\}$ has a $1/(L_c - 1)$ probability. See S1 Appendix for details. The second factor regards the AE event; if an AE has happened, how many ways are there to get N_c^3 erroneous fragments? Here we should note that, even though we know how L_c is partitioned into amplification trees, we do not know the internal structure of the trees. We need to consider all possible ways of forming the amplification trees (i.e., marginalization of the amplification tree topologies). We model an amplification tree as described in S1 Appendix. Assuming a count configuration of $N_c = (N_c^1, N_c^2, N_c^3)$ where $N_c^i > 0$ for all $i \in \{1, 2, 3\}$ and that the AE is happening strictly on the first amplification tree, there are $C(N_c^1 + N_c^3)$ possible tree topologies of the first amplification tree, each with the probability of $1/C(N_c^1 + N_c^3)$. The subscript c is the single-cell id, whereas $C(.)$ or $C(., .)$ is a function that returns the number of possible tree topologies given specific tree details in the parenthesis. Moreover, $C(N_c^1 + N_c^3, N_c^3)$ out of $E^{L_c}_{D_c^1, D_c^2}$ edges satisfy the specified count configuration in this marginalized amplification tree space. When all these are combined, the second component becomes
$C(N_c^1 + N_c^3, N_c^3)/(C(N_c^1 + N_c^3) \times E^{L_c}_{D_c^1, D_c^2})$. An example of tree topologies and edges supporting a count configuration is shown in Fig 10. We direct the reader to Tables A and B in S1 Appendix for all the count and fragment type configurations.

The final component is the probability of F_c^3 given AE; in the case of no AE, the probability is

$$P(F_c^3|A_c = 0) = \begin{cases} 1, & \text{if } F_c^3 = \emptyset \\ 0, & \text{otherwise,} \end{cases}$$

and in the case of AE, the probability is 1/3 for the singleton sites (1 × 3 = 3 different possible genotypes differ by 1 nucleotide), and 1/6 for paired sites (there are 2 × 3 = 6 possibilities).

The product of the three components leads to the fragment type and counts probability; $P(F_c, N_c|D_c^1, D_c^2, A_c, X_c, L_c)$, which is detailed in Tables A and B in S1 Appendix.

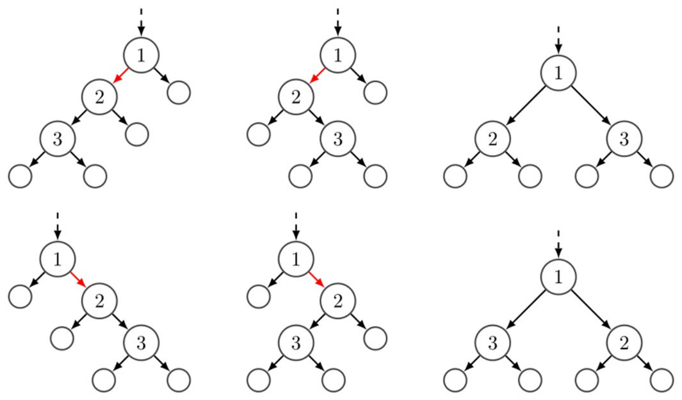

Fig 10. Illustration of $C(4) = 6$ possible 4-trees. The labeled nodes indicate the order of the amplification events. The dashed line represents an incoming edge to the root to account for subsampling during the sequencing of the fragments. The red edges are all possible 3-edges in 4-trees, $C(4, 3)$.

https://doi.org/10.1371/journal.pcbi.1012094.g010

Read sequencing. Read sequencing is also erroneous and depends on sequencing technology [22–25]. We use the Phred quality scores (ρ) to compute the base-calling error probabilities, Q, [27, 28]; $Q = 10^{-0.1\times\rho}$.

For a single read with a known originating fragment, the likelihood of the read at a singleton site is

$$P(R_c^l | F_c^l, Q_c^l = q) = \begin{cases} 1 - q, & \text{if no error} \\ \frac{q}{3}, & \text{if error,} \end{cases} \tag{6}$$

and the likelihood of the read at a paired site is

$$P(R_c^l | F_c^l, Q_c^l = (q^s, q^{s'})) = \begin{cases} (1 - q^s)\,(1 - q^{s'}), & \text{if no errors} \\ (1 - q^s)\,\frac{q^{s'}}{3}, & \text{if error only at locus } s \\ \frac{q^s}{3}\,(1 - q^{s'}), & \text{if error only at locus } s' \\ \frac{q^s}{3}\,\frac{q^{s'}}{3}, & \text{if errors at both positions.} \end{cases} \tag{7}$$

We use dynamic programming to efficiently compute the likelihood of multiple reads, $P(\mathbf{R}_c = R_c^{1:L_c} | F_c, N_c, \mathbf{Q}_c = Q_c^{1:L_c})$, of cell c. In the dynamic programming algorithm, we introduce the reads of a cell one at a time, assign them to different fragments, and track how the likelihood of reads changes with the addition of the new read. For instance, when introducing the l'th read, we know what the likelihood of the previous reads ($R_c^{1:l-1}$) is if they all originate from the first fragment, all from the second one, or any other partition (e.g., 2 reads from first fragment, no reads from second fragment, $l - 3$ from the third fragment). After adding the L_c'th read, we extract the likelihood of the corresponding N_c configuration from the dynamic programming. The pseudocode is shown in S1 Appendix.

Runtime analysis. The runtime complexity for the dynamic programming algorithm computing the probability of the observed reads for cell c, $P(\mathbf{R}_c | X_c, \mathbf{Q}_c, L_c, p_{ae}, p_{ado})$, is L_c^3—resulting in $O(|\Pi| C L^3)$ where L is the maximum number of reads over cell and site. When we perform pairwise distance calculation for each site, we incur the cost of $|\Pi| C^2$ times the look up of of the dynamic programming table across possible mutations $m = 0, \ldots, C$, yielding $|\Pi| C^3$. This results in the total run time of $O(|\Pi|(C^3 + CL^3))$.

Lineage tree reconstruction

The standard NJ algorithm [42] and its variants, such as FastNJ [44], are commonly used for distance-based methods. In the final step of Scuphr, the standard NJ algorithm is applied to the distance matrix to reconstruct the cell lineage tree using the implementation in the Dendropy library [45]. The tree is re-rooted, so the bulk node becomes the root and indicates the non-mutated state.

Site selection

We use several heuristics to identify candidate loci for analysis. Even though Scuphr can run on the sites with no observed alternate nucleotides, these sites would not contribute information about the topology of the lineage tree and waste computational resources. Instead, we select a subset of the genome that could provide information regarding the topology.

Paired site selection. The main goal for the paired site selection is to find pairs of loci with a sufficient amount of alternative nucleotides *and* a heterozygous site nearby that can be used for read-phasing.

First, we identify the gSNVs using the unamplified bulk reads taken from another tissue. We run FreeBayes [46] software and set the read depth threshold to 10 and alternative nucleotide frequency to 0.2. The sites with heterozygous genotypes are considered the gSNV sites. Second, we check single-cell reads that cover the gSNV site and ensure at least two single-cells display the gSNV; that is, both nucleotides must be present in at least 20% of the reads from both cells. After this verification, we look for the candidate mutation sites around the gSNV. Both gSNV and the candidate sites must be covered with the same read to facilitate read-phasing. All nucleotides covered by a read come from the same allele. The reference nucleotide of the candidate site is determined from bulk reads; the site must have at least 10 reads in bulk data, and at least 80% of the reads are one nucleotide, which is referred to as the *reference*. For a site to be picked, at least 2 and at most $C - 1$ cells must agree on an alternative nucleotide (at least 20% of the reads should be different from the reference). The signal from a single-cell or all single-cells does not contribute to the information for lineage tree reconstruction. If multiple gSNVs are near the candidate site, the closest gSNV is used to form the pair. Finally, one last gSNV check is done to ensure the single-cell reads (that cover both the candidate and gSNV sites) meet the gSNV requirement described above. The further the candidate site is from the gSNV site, the fewer reads cover both sites.

Singleton site selection. During the singleton site selection, the gSNV heuristics are omitted. The candidate mutation site identification is performed using the same heuristics as the paired site selection.

Hybrid site selection. In the hybrid case, the algorithm works with both paired and singleton sites. A candidate mutation site is paired with a gSNV if the paired site criteria are met; otherwise, the site is picked as a singleton site.

Parameter estimation and hyperparameter settings

We run the Metropolis-Hastings algorithm for 5, 000 iterations with three different initial values to infer the parameters p_{ado} and p_{ae}. We discard the first 20% samples as burn-in. The acceptance ratio of our Metropolis-Hastings algorithm is

$$\begin{aligned} A(\Theta^*, \Theta) &= \min\left(1,\ \frac{q(p_{ado}, p_{ae} | p^*_{ado}, p^*_{ae})}{q(p^*_{ado}, p^*_{ae} | p_{ado}, p_{ae})}\ \frac{P(p^*_{ado}, p^*_{ae})}{P(p_{ado}, p_{ae})}\ \mathcal{L}\right) \\ &= \min\left(1,\ \frac{P(p^*_{ado})P(p^*_{ae})}{P(p_{ado})P(p_{ae})}\ \mathcal{L}\right), \end{aligned}$$

where

$$\mathcal{L} = \frac{\prod_\pi P(\mathbf{R}^\pi_{1:C} | p^*_{ado}, p^*_{ae}, B^\pi, \mathbf{Q}^\pi_{1:C}, L^\pi_{1:C}, \alpha, a, b)}{\prod_\pi P(\mathbf{R}^\pi_{1:C} | p_{ado}, p_{ae}, B^\pi, \mathbf{Q}^\pi_{1:C}, L^\pi_{1:C}, \alpha, a, b)}.$$

The likelihood is calculated similarly to the earlier derivations in Methods;

$$\begin{aligned} &P(\mathbf{R}^\pi_{1:C} | p_{ado}, p_{ae}, B^\pi, Q^\pi_{1:C}, L^\pi_{1:C}, \alpha, a, b) \\ &= \sum_z P(Z^\pi | B^\pi, \alpha) \\ &\quad \sum_{G^\pi_{1:C}} P(G^\pi_{1:C} | a, b)\ P(\mathbf{R}^\pi_{1:C} | G^\pi_{1:C}, Z^\pi, p_{ado}, p_{ae}, B^\pi, \mathbf{Q}^\pi_{1:C}, L^\pi_{1:C}). \end{aligned}$$

We use a Gaussian random walk proposal in which each parameter is treated independently, and samples are proposed using a Gaussian distribution with a 0.01 standard deviation. We set uniform prior probabilities for the parameters, calculate the likelihood of the observed reads $R^{1:\Pi}_{1:C}$ based on our model, and accept or reject the samples. The means of the samples are used as p_{ado} and p_{ae} parameters during the analysis.

Scuphr has three hyperparameters; α, a, and b. α is the concentration parameter of the Dirichlet-Categorical distribution used for the mutation type probabilities. We set α to all-ones vector. The a and b hyperparameters are for the Beta prior to mutation probability p_m. We assigned uniform prior to the mutation probability by setting $a = 1$ and $b = 1$. However, the user can set these parameters and, thereby, modify the algorithm's tendency towards mutations.

For our experiments, we randomly picked 20 sites that are used to estimate the parameters. We sampled the initial value of p_{ado} from $U[0, 1]$ and p_{ae} from $U[0, 0.1]$. We set the initial range of p_{ae} to $[0, 0.1]$ because the AE probabilities are reported to be small [19, 30].

Simulation of synthetic datasets

We generated the synthetic dataset as follows. First, we generated random binary cell lineage trees with C leaves. We assigned $2 \times (C - 1) \times \mu$ mutations, $\mu \in \{10, 20\}$, to the tree's edges and ensured each edge had at least one mutation. For each dataset, we generated a 1 million base pairs long diploid genome that was used for the bulk and the single cells. We randomly picked mutation loci from odd-numbered bases in the genome. For each of the phasing frequencies $\rho \in \{0.001, 0.01, 0.1, 1\}$, we picked $\rho \times 500,000$ base pairs as gSNV sites and placed them randomly in even indexed locations in the genome. So for $\rho = 1$, every second position in the genome is a gSNV site, and each read containing a mutation has an accompanying gSNV site. This construction is sufficient since the distance between sites does not affect the site selection or the subsequent analysis. The gSNV sites are shared by the single-cell genomes, and the mutation sites are shared according to their placement in the cell lineage tree. Further, we masked the single-cell genomes to account for cell-specific ADO events. For each pair of sites (which consists of consecutive positions, one even and one odd), we dropped the maternal and paternal alleles independently with $p_{ado} \in \{0, 0.1, 0.2\}$.

The fragments were generated using the Pólya urn process for each site. The masked single-cell genome determined the initial state of the urn. If both alleles are dropped out, no fragments were generated. In the case of a single ADO, all the fragments are generated from the non-dropped allele. If there is no dropout, the fragments are simulated from both alleles, i.e., the urn was initialized with two balls of different colors. The number of fragments per pair of sites was sampled from a Poisson distribution with rate parameter $\lambda \in \{10, 20\}$, i.e., an interval that contains the read depth found in our biological data. Whenever a fragment is copied, an AE occurs with p_{ae} independently. So, even though we base our inference on assuming that there is at most one AE per site, we allowed multiple errors during the data simulation. We use $p_{ae} = 10^{-5}$ or $p_{ae} = 10^{-3}$ for all cells of a dataset; throughout the paper, we refer to these datasets as a dataset with *low* and *high* AEs, respectively.

We simulated how the fragments are sequenced so that reads are obtained, as follows. Phred quality scores were sampled from a discrete Uniform distribution in the range [30, 42]. A sequencing error was introduced based on the corresponding base-calling error probability.

We used a straightforward approach for the bulk reads and replicated the bulk genome 15 times. One can view this step as the bulk data is generated using 15 single-cell genomes without any mutations. This replication does not contain a DNA amplification step since the bulk data consists of reads sequenced from unamplified fragments.

Accuracy metrics

This section describes the two accuracy metrics used for the analysis.

The similarity score. The Robinson-Foulds (RF) distance [47] is a symmetric difference metric commonly used for phylogenetic tree comparisons [36, 48]. The metric calculates the total number of bipartitions in either of the trees but not in both. We normalized the RF scores (nRF) by the total number of non-trivial bipartitions in both tree topologies (a leaf's edge is considered trivial. If two tree topologies have the same leaf set, there will be edges that define the same bipartition. There will be C identical bipartitions, regardless of the internal structure of the trees) nRF = RF/I_B, where I_B is the total number of internal edges in the two tree topologies. Notice that the number of non-trivial edges of a tree depends on its topology, e.g., whether the tree is binary or not. We used a similarity score,

$$\text{similarity score} \quad = 1 - \text{nRF},$$

in [0, 1]. If the trees have the same topology, the similarity score is 1. If the trees do not have any common non-trivial bipartition, the similarity score is 0.

Transfer bootstrap expectation. The TBE is introduced as an alternative metric to compute the support of bootstrap trees on a reference tree topology [43]. Compared to Felsenstein's bootstrap proportions [49], which checks how frequently an edge appears in the bootstrap trees, the TBE metric penalizes the slight topological differences less. TBE metric computes the number of operations (e.g., taxa removal) required to match an edge in the bootstrap tree to the reference tree.

TBE score of an edge is in [0, 1], where 1 means the edge exists in all bootstrap trees and 0 means the edge appears at random. The higher the TBE score of an edge, the better.

Discussion

We evaluated the performance of Scuphr on a biological dataset with SCIΦ and Phylovar, and on several simulated datasets with SCIΦ. Our investigation focused on the algorithm's robustness for varying numbers of single-cells, read coverages, and technological artifacts such as AEs, ADOs, and sequencing errors. We observed that for the low amplification error datasets, Scuphr performs on par or better than SCIΦ in most cases. For the high amplification error datasets, Scuphr consistently outperforms SCIΦ. When provided by the candidate mutation sites selected by our method, SCIΦ's performance becomes similar to Scuphr in most low amplification error datasets; however, Scuphr continues to outperform SCIΦ in the high amplification error datasets.

In addition, we showed that the algorithm scales with the number of single cells and sites. Moreover, it scales inversely with the number of cores. For instance, using a single core, the main part of Scuphr takes approximately 1.6 hours for 20 cells and 1024 singleton sites. The time required for 100, 000 sites would be approximately 166.7 hours; however, with a modest infrastructure of five compute nodes with 32 CPU cores each, Scuphr's runtime can be reduced to approximately 1 hour. This advantage makes it possible to analyze millions of sites in the genome, whereas most state-of-the-art methods can handle only a few thousand sites.

Finally, we evaluated the performance of Scuphr using a biological dataset of 18 single cells acquired from [30]. We selected approximately 3.4 million candidate sites for analysis and used bootstrapping to obtain edge supports on the reference tree topology. Although the biological dataset was challenging, Scuphr assigned high support values to the edges that separate the two main clones and some closely related cells. Scuphr outperformed both SCIΦ and Phylovar by obtaining higher edge supports for a subclone, but was unable to support a sibling relation that was successfully identified by Phylovar.

The graphical model of Scuphr that is tailored for the healthy, diploid scDNA-seq data, combined with the NJ algorithm yields higher or as good accuracy as the state-of-the-art methods. Scuphr's computational complexity is affected quadratically by the number of single cells. However, this issue can be redeemed with the embarrassingly parallel nature of the method. In addition, the dynamic programming technique is leveraged to calculate the read likelihoods efficiently. In the experiments, we showed how the runtimes of singleton and paired sites differ. The computations of the paired sites are considerably slower due to the more common mutation types and amplification error types to marginalize over. Since there are far fewer paired sites selected from the biological dataset for analysis, this wouldn't cause an issue. In addition, as a design choice, we used bulk data taken from another tissue to represent the unamplified, non-mutated state. Even though there are publicly available datasets (such as the fibroblast data used in this study) that contain both single-cell and bulk data, this might not always be the case. In the absence of the bulk data, publicly available known single nucleotide polymorphism datasets could be used for read-phasing. Alternatively, the analysis could be limited to singleton sites and the site selection could be done by comparing the reads of single-cells to a reference genome.

As our goal is to reconstruct the cell lineage tree topology, we have not placed emphasis on estimating the the branch lengths. With our experimental studies, we have shown that Scuphr has high accuracy in terms of tree topology reconstruction; however, we caution that the branch lengths estimates obtained in the tree reconstruction step needs further validation.

Conclusion

Single-cell DNA sequencing technologies enable detailed analyses of development and cell differentiation [1–3]. We presented Scuphr, a probabilistic framework that reconstructs cell lineage trees from healthy, diploid single-cells using whole-genome amplified DNA sequencing data. Scuphr is designed with the challenges of the scDNA-seq data in mind, it fits well with the biological findings, and in particular, it obtains better accuracy with leveraging read-phasing.

In addition to the distance-based and MCMC-based methods, various variational inference based methods have recently been developed for tree reconstruction tasks [50–53]. These methods typically operate in the standard phylogeny setting and require a good set of initial trees for their analysis. In the potential future development of such methods where the domain moves toward the single-cell setting, Scuphr could provide a good set of bootstrap trees as input quickly.

Scuphr is designed for healthy, diploid scDNA-seq data. However, it can be enhanced to handle cancer data by incorporating copy number variations into its model. We will investigate how the extended model handles the challenges of single-cell tumor data and compare its performance with state-of-the-art methods in our future work.

Supporting information

S1 Appendix. Supplementary information. The file includes additional formulations, biological dataset, and benchmarking details.
(PDF)

Acknowledgments

The computations and data handling were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC).

Author Contributions

Conceptualization: Hazal Koptagel, Seong-Hwan Jun, Jens Lagergren.

Formal analysis: Hazal Koptagel, Jens Lagergren.

Funding acquisition: Jens Lagergren.

Investigation: Hazal Koptagel, Joanna Hård, Jens Lagergren.

Methodology: Hazal Koptagel, Seong-Hwan Jun, Jens Lagergren.

Project administration: Jens Lagergren.

Software: Hazal Koptagel, Seong-Hwan Jun.

Supervision: Jens Lagergren.

Validation: Hazal Koptagel.

Visualization: Hazal Koptagel.

Writing – original draft: Hazal Koptagel, Jens Lagergren.

Writing – review & editing: Hazal Koptagel.

References

1. Lodato MA, Woodworth MB, Lee S, Evrony GD, Mehta BK, Karger A, et al. Somatic mutation in single human neurons tracks developmental and transcriptional history. Science. 2015; 350(6256):94–98. https://doi.org/10.1126/science.aab1785 PMID: 26430121
2. Marioni JC, Arendt D. How single-cell genomics is changing evolutionary and developmental biology. Annu Rev Cell Dev Biol. 2017; 33:537–553. https://doi.org/10.1146/annurev-cellbio-100616-060818 PMID: 28813177

3. Lodato MA, Rodin RE, Bohrson CL, Coulter ME, Barton AR, Kwon M, et al. Aging and neurodegeneration are associated with increased mutations in single human neurons. Science. 2018; 359(6375):555–559. https://doi.org/10.1126/science.aao4426 PMID: 29217584

4. Lee-Six H, Øbro NF, Shepherd MS, Grossmann S, Dawson K, Belmonte M, et al. Population dynamics of normal human blood inferred from somatic mutations. Nature. 2018; 561(7724):473–478. https://doi.org/10.1038/s41586-018-0497-0 PMID: 30185910

5. Bae T, Tomasini L, Mariani J, Zhou B, Roychowdhury T, Franjic D, et al. Different mutational rates and mechanisms in human cells at pregastrulation and neurogenesis. Science. 2018; 359(6375):550–555. https://doi.org/10.1126/science.aan8690 PMID: 29217587

6. Coorens THH, Moore L, Robinson PS, Sanghvi R, Christopher J, Hewinson J, et al. Extensive phylogenies of human development inferred from somatic mutations. Nature. 2021; 597(7876):387–392. https://doi.org/10.1038/s41586-021-03790-y PMID: 34433963

7. Navin N, Kendall J, Troge J, Andrews P, Rodgers L, McIndoo J, et al. Tumour evolution inferred by single-cell sequencing. Nature. 2011; 472(7341):90–94. https://doi.org/10.1038/nature09807 PMID: 21399628

8. Roth A, Khattra J, Yap D, Wan A, Laks E, Biele J, et al. PyClone: statistical inference of clonal population structure in cancer. Nat Methods. 2014; 11(4):396–398. https://doi.org/10.1038/nmeth.2883 PMID: 24633410

9. Deshwar AG, Vembu S, Yung CK, Jang GH, Stein L, Morris Q. PhyloWGS: reconstructing subclonal composition and evolution from whole-genome sequencing of tumors. Genome Biol. 2015; 16:35. https://doi.org/10.1186/s13059-015-0602-8 PMID: 25786235

10. Safinianaini N, de Souza CPE, Lagergren J. CopyMix: mixture model based single-cell Clustering and Copy Number Profiling using Variational Inference. bioRxiv. 2021;.

11. Jun SH, Toosi H, Mold J, Engblom C, Chen X, O'Flanagan C, et al. Reconstructing clonal tree for phylo-phenotypic characterization of cancer using single-cell transcriptomics. Nat Commun. 2023; 14(1):982. https://doi.org/10.1038/s41467-023-36202-y PMID: 36813776

12. Lynch M. Evolution of the mutation rate. Trends Genet. 2010; 26(8):345–352. https://doi.org/10.1016/j.tig.2010.05.003 PMID: 20594608

13. Belkadi A, Bolze A, Itan Y, Cobat A, Vincent QB, Antipenko A, et al. Whole-genome sequencing is more powerful than whole-exome sequencing for detecting exome variants. Proc Natl Acad Sci U S A. 2015; 112(17):5473–5478. https://doi.org/10.1073/pnas.1418631112 PMID: 25827230

14. Navin NE. Cancer genomics: one cell at a time. Genome Biol. 2014; 15(8):452. https://doi.org/10.1186/s13059-014-0452-9 PMID: 25222669

15. Gawad C, Koh W, Quake SR. Single-cell genome sequencing: current state of the science. Nat Rev Genet. 2016; 17(3):175. https://doi.org/10.1038/nrg.2015.16 PMID: 26806412

16. Dong X, Zhang L, Milholland B, Lee M, Maslov AY, Wang T, et al. Accurate identification of single-nucleotide variants in whole-genome-amplified single cells. Nat Methods. 2017; 14(5):491–493. https://doi.org/10.1038/nmeth.4227 PMID: 28319112

17. Zafar H, Navin N, Nakhleh L, Chen K. Computational approaches for inferring tumor evolution from single-cell genomic data. Curr Opin Syst Biol. 2018; 7:16–25. https://doi.org/10.1016/j.coisb.2017.11.008

18. Lähnemann D, Köster J, Szczurek E, McCarthy DJ, Hicks SC, Robinson MD, et al. Eleven grand challenges in single-cell data science. Genome Biol. 2020; 21(1):31. https://doi.org/10.1186/s13059-020-1926-6 PMID: 32033589

19. Bourcy CFAd, de Bourcy CFA, De Vlaminck I, Kanbar JN, Wang J, Gawad C, et al. A Quantitative Comparison of Single-Cell Whole Genome Amplification Methods. PLoS ONE. 2014; 9(8):e105585. https://doi.org/10.1371/journal.pone.0105585 PMID: 25136831

20. Dean FB, Hosono S, Fang L, Wu X, Faruqi AF, Bray-Ward P, et al. Comprehensive human genome amplification using multiple displacement amplification. Proc Natl Acad Sci U S A. 2002; 99(8):5261–5266. https://doi.org/10.1073/pnas.082089499 PMID: 11959976

21. Zong C, Lu S, Chapman AR, Xie XS. Genome-wide detection of single-nucleotide and copy-number variations of a single human cell. Science. 2012; 338(6114):1622–1626. https://doi.org/10.1126/science.1229164 PMID: 23258894

22. Pfeiffer F, Gröber C, Blank M, Händler K, Beyer M, Schultze JL, et al. Systematic evaluation of error rates and causes in short samples in next-generation sequencing. Sci Rep. 2018; 8(1):10950. https://doi.org/10.1038/s41598-018-29325-6 PMID: 30026539

23. Petrackova A, Vasinek M, Sedlarikova L, Dyskova T, Schneiderova P, Novosad T, et al. Standardization of Sequencing Coverage Depth in NGS: Recommendation for Detection of Clonal and Subclonal Mutations in Cancer Diagnostics. Front Oncol. 2019; 9. https://doi.org/10.3389/fonc.2019.00851 PMID: 31552176

24. Ma X, Shao Y, Tian L, Flasch DA, Mulder HL, Edmonson MN, et al. Analysis of error profiles in deep next-generation sequencing data. Genome Biol. 2019; 20(1):50. https://doi.org/10.1186/s13059-019-1659-6 PMID: 30867008

25. Stoler N, Nekrutenko A. Sequencing error profiles of Illumina sequencing instruments. NAR Genom Bioinform. 2021; 3(1):lqab019. https://doi.org/10.1093/nargab/lqab019 PMID: 33817639

26. Zafar H, Wang Y, Nakhleh L, Navin N, Chen K. Monovar: single-nucleotide variant detection in single cells. Nat Methods. 2016; 13(6):505–507. https://doi.org/10.1038/nmeth.3835 PMID: 27088313

27. Ewing B, Green P. Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities. Genome Res. 1998; 8(3):186–194. https://doi.org/10.1101/gr.8.3.175 PMID: 9521922

28. Ewing B, Hillier L, Wendl MC, Green P. Base-Calling of Automated Sequencer Traces Using Phred. I. Accuracy Assessment. Genome Res. 1998; 8(3):175–185. https://doi.org/10.1101/gr.8.3.175 PMID: 9521921

29. Bohrson CL, Barton AR, Lodato MA, Rodin RE, Luquette LJ, Viswanadham VV, et al. Linked-read analysis identifies mutations in single-cell DNA-sequencing data. Nat Genet. 2019; 51(4):749–754. https://doi.org/10.1038/s41588-019-0366-2 PMID: 30886424

30. Hård J, Al Hakim E, Kindblom M, Björklund ÅK, Sennblad B, Demirci I, et al. Conbase: a software for unsupervised discovery of clonal somatic mutations in single cells through read phasing. Genome Biol. 2019; 20(1):1–18. https://doi.org/10.1186/s13059-019-1673-8 PMID: 30935387

31. Yuan K, Sakoparnig T, Markowetz F, Beerenwinkel N. BitPhylogeny: a probabilistic framework for reconstructing intra-tumor phylogenies. Genome Biol. 2015; 16:36. https://doi.org/10.1186/s13059-015-0592-6 PMID: 25786108

32. Ross EM, Markowetz F. OncoNEM: inferring tumor evolution from single-cell sequencing data. Genome Biol. 2016; 17:69. https://doi.org/10.1186/s13059-016-0929-9 PMID: 27083415

33. Jahn K, Kuipers J, Beerenwinkel N. Tree inference for single-cell data. Genome Biol. 2016; 17:86. https://doi.org/10.1186/s13059-016-0936-x PMID: 27149953

34. Zafar H, Tzen A, Navin N, Chen K, Nakhleh L. SiFit: inferring tumor trees from single-cell sequencing data under finite-sites models. Genome Biol. 2017; 18(1):178. https://doi.org/10.1186/s13059-017-1311-2 PMID: 28927434

35. Zafar H, Navin N, Chen K, Nakhleh L. SiCloneFit: Bayesian inference of population structure, genotype, and phylogeny of tumor clones from single-cell genome sequencing data. Genome Res. 2019; 29 (11):1847–1859. https://doi.org/10.1101/gr.243121.118 PMID: 31628257

36. Kozlov A, Alves JM, Stamatakis A, Posada D. CellPhy: accurate and fast probabilistic inference of single-cell phylogenies from scDNA-seq data. Genome Biol. 2022; 23(1):37. https://doi.org/10.1186/s13059-021-02583-w PMID: 35081992

37. Singer J, Kuipers J, Jahn K, Beerenwinkel N. Single-cell mutation identification via phylogenetic inference. Nat Commun. 2018; 9(1):1–8. https://doi.org/10.1038/s41467-018-07627-7 PMID: 30514897

38. Kimura M. The number of heterozygous nucleotide sites maintained in a finite population due to steady flux of mutations. Genetics. 1969; 61(4):893–903. https://doi.org/10.1093/genetics/61.4.893 PMID: 5364968

39. Watterson GA. On the number of segregating sites in genetical models without recombination. Theor Popul Biol. 1975; 7(2):256–276. https://doi.org/10.1016/0040-5809(75)90020-9 PMID: 1145509

40. Tajima F. Infinite-allele model and infinite-site model in population genetics. J Genet. 1996; 75(1):27–31. https://doi.org/10.1007/BF02931749

41. Edrisi M, Valecha MV, Chowdary SBV, Robledo S, Ogilvie HA, Posada D, et al. Phylovar: toward scalable phylogeny-aware inference of single-nucleotide variations from single-cell DNA sequencing data. Bioinformatics. 2022; 38(Suppl 1):i195–i202. https://doi.org/10.1093/bioinformatics/btac254 PMID: 35758771

42. Saitou N, Nei M. The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol Biol Evol. 1987; 4(4):406–425. PMID: 3447015

43. Lemoine F, Entfellner JBD, Wilkinson E, Correia D, Felipe MD, De Oliveira T, et al. Renewing Felsenstein's phylogenetic bootstrap in the era of big data. Nature. 2018; 556(7702):452–456. https://doi.org/10.1038/s41586-018-0043-0 PMID: 29670290

44. Elias I, Lagergren J. Fast neighbor joining. Theor Comput Sci. 2009; 410(21):1993–2000. https://doi.org/10.1016/j.tcs.2008.12.040

45. Sukumaran J, Holder MT. DendroPy: a Python library for phylogenetic computing. Bioinformatics. 2010; 26(12):1569–1571. https://doi.org/10.1093/bioinformatics/btq228 PMID: 20421198

46. Garrison E, Marth G. Haplotype-based variant detection from short-read sequencing. arXiv. 2012;.

47. Robinson DF, Foulds LR. Comparison of phylogenetic trees. Math Biosci. 1981; 53(1-2):131–147. https://doi.org/10.1016/0025-5564(81)90043-2

48. Wang L, Bouchard-Côté A, Doucet A. Bayesian phylogenetic inference using a combinatorial sequential Monte Carlo method. J Am Stat Assoc. 2015; 110(512):1362–1374. https://doi.org/10.1080/01621459.2015.1054487

49. Felsenstein J. Confidence limits on phylogenies: an approach using the bootstrap. Evolution. 1985; 39 (4):783–791. https://doi.org/10.1111/j.1558-5646.1985.tb00420.x PMID: 28561359

50. Zhang C, Matsen IV FA. Variational Bayesian phylogenetic inference. In: Int. Conf. Learn. Represent.; 2018.

51. Zhang C. Improved variational bayesian phylogenetic inference with normalizing flows. "Adv Neural Inf Process Syst". 2020; 33:18760–18771.

52. Zhang C, Matsen IV FA. A Variational Approach to Bayesian Phylogenetic Inference. arXiv. 2022;.

53. Koptagel H, Kviman O, Melin H, Safinianaini N, Lagergren J. VaiPhy: A variational inference based algorithm for phylogeny. In: Advances in Neural Information Processing Systems 35; 2022.