Genome Biology

METHOD Open Access

Tree inference for single-cell data

Katharina Jahn[1,2†], Jack Kuipers[1,2†] and Niko Beerenwinkel[1,2*]

Abstract

Understanding the mutational heterogeneity within tumors is a keystone for the development of efficient cancer therapies. Here, we present SCITE, a stochastic search algorithm to identify the evolutionary history of a tumor from noisy and incomplete mutation profiles of single cells. SCITE comprises a flexible Markov chain Monte Carlo sampling scheme that allows the user to compute the maximum-likelihood mutation history, to sample from the posterior probability distribution, and to estimate the error rates of the underlying sequencing experiments. Evaluation on real cancer data and on simulation studies shows the scalability of SCITE to present-day single-cell sequencing data and improved reconstruction accuracy compared to existing approaches.

Background

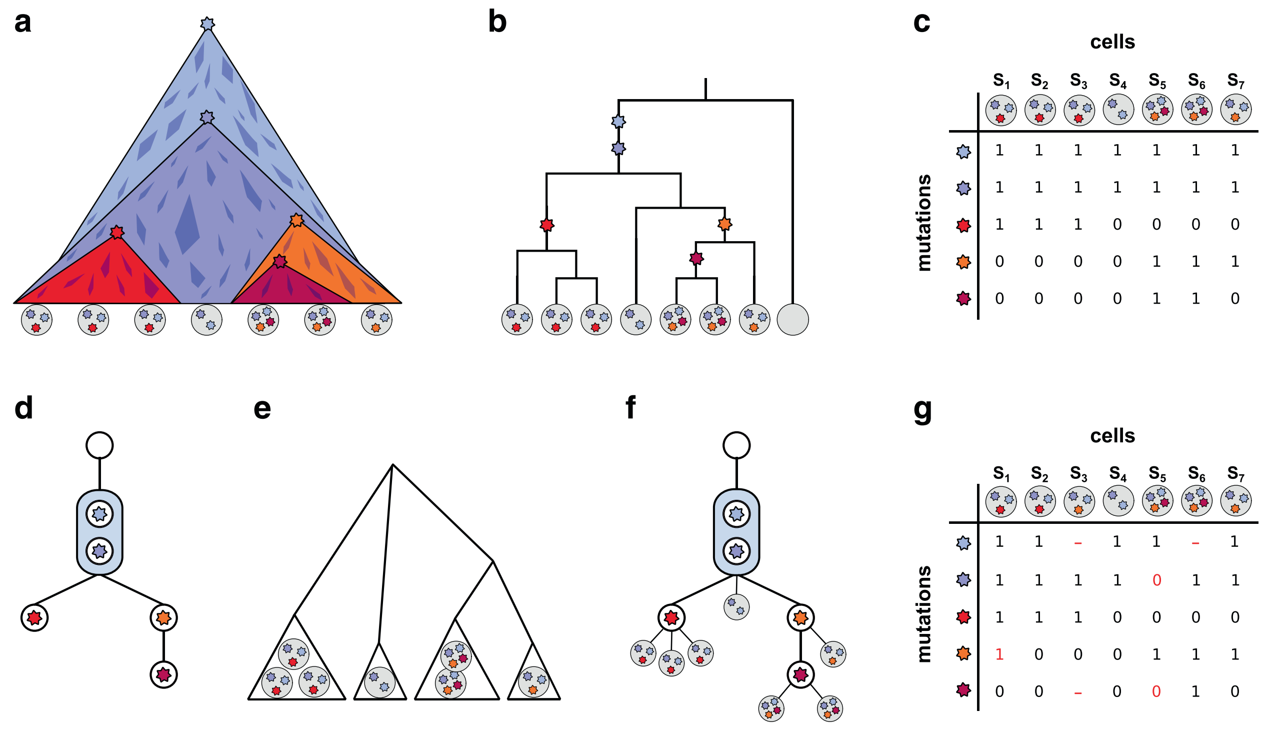
Tumor progression can be described as a dynamic evolutionary process acting at the level of individual cells [1–3]. A tumor typically arises from a single founder cell whose distinct set of genetic (and epigenetic) lesions gives it a growth advantage over the surrounding cells and helps it to evade the patient's immune response. As a consequence, the clone arising from this cell expands and, over the course of time, the descendant cells develop further into subclones by acquiring additional somatic mutations [4]. The subclones compete against each other for resources in the tumor environment and the more successful ones will replace others until eventually they themselves are out-competed by new subclones [4, 5]; see also Fig. 1a.

The genetic diversity arising from this process, referred to as intra-tumor heterogeneity, is believed to be a major cause of relapse after cancer treatment [6, 7]. The common explanation is that drug therapy often targets the dominant subclone at the time of diagnosis, and upon its remission, either an expansion of previously suppressed subclones, non-susceptible to the treatment, or an emergence of new resistant subclones is likely to happen [8]. For monoclonal tumor progression, the temporal order in which specific mutations have occurred has been shown to be informative for disease progression and susceptibility to drug therapy [9]. Therefore, a more comprehensive understanding of the genetic diversity of individual tumors and their evolutionary history is likely to be a key component in the design of personalized cancer therapies that are more effective [6, 10, 11].

All cells in a tumor are related via a binary genealogical tree (Fig. 1b). To reconstruct their evolutionary history based on single-nucleotide variants (SNVs), the infinite sites assumption is typically made, which implies that the mutation profiles of the cells (Fig. 1c) form a perfect phylogeny. A perfect phylogeny exists if for all pairs of mutations i_1, i_2, the set of cells having mutation i_1 and the set of cells having mutation i_2 are either disjoint or one is a subset of the other [12]. Most approaches to reconstructing tumor phylogenies focus on the partial (temporal) order among the mutation events (Fig. 1d). This tree type implicitly defines the set of possible subclones via the mutation profiles that can be read from the tree by collecting the mutations on the path from the root to any other node in the tree. Not all possible subclones, in particular those at inner nodes, need to have surviving cells. Also, by chance, cells from surviving subclones may not be sampled.

The main challenge in obtaining knowledge on intra-tumor heterogeneity is that common bulk high-throughput sequencing admixes the DNA of millions of cells in a sample before sequencing. The mutation profiles obtained from the mixture constitute an average of an unknown number of unknown subclones each making up an unknown fraction of the mixture [13]. Therefore, tree

*Correspondence: niko.beerenwinkel@bsse.ethz.ch
This paper was selected for oral presentation at RECOMB 2016 and an abstract is published in the conference proceedings.
†Equal Contributors
[1] Department of Biosystems Science and Engineering, ETH Zurich, Basel, Switzerland
[2] SIB, Swiss Institute of Bioinformatics, Basel, Switzerland



	S_1	S_2	S_3	S_4	S_5	S_6	S_7
★	1	1	1	1	1	1	1
★	1	1	1	1	1	1	1
★	1	1	1	0	0	0	0
★	0	0	0	0	1	1	1
★	0	0	0	0	1	1	0

	S_1	S_2	S_3	S_4	S_5	S_6	S_7
★	1	1	–	1	1	–	1
★	1	1	1	1	0	1	1
★	1	1	1	0	0	0	0
★	1	0	0	0	1	1	1
★	0	0	–	0	0	1	0

Fig. 1 Tumor evolution and cell phylogeny. **a** Schematic representation of tumor evolution with time progressing downwards. *Stars* denote new mutations leading to subclone expansion. The *quadrangles* belong to minor extinct subclones with no traces in the present-day populations. The mutations founding these clones may not have induced a sufficient growth advantage to have surviving descendant cells or may have been lost by chance. The *gray discs* on the bottom denote single cells sequenced after tumor removal. The *stars* they contain indicate the mutations observed in the cell. **b** Binary genealogical tree of the sequenced cells. An *empty disc* represents a normal somatic cell, which is an outgroup for the tumor cells. **c** Binary mutation matrix representing the mutation status of the sequenced tumor cells. A *zero* entry denotes the absence of a mutation in the respective cell, while a *one* denotes its presence. **d** The perfect phylogeny represented as a mutation tree, the partial (temporal) order of the mutation events. Mutations are summarized in a single node when their order is unidentifiable from the sampled cells, as is the case here for the two top-most mutations with the matrix from (**c**). **e** Hierarchical subclone structure. Cells with identical mutation profiles cluster into subclones, which serve as taxa in this phylogenetic tree. **f** Mutation tree with single-cell samples attached. **g** Noisy mutation matrix with missing values. The *red numbers* indicate flipped mutation states with respect to the true mutation matrix in (**c**). For 0 → 1, a false positive, the mutation is called but not present in the cell. For 1 → 0, a false negative, the mutation is not called but present in the cell, most likely due to allelic dropout during the DNA amplification. The *red dash* indicates a missing value; it is unknown whether the site is mutated or in the normal state in this cell

reconstruction needs to be completed by a deconvolution of the mixed signal to identify the subclones, the taxa of the tree. In the past years, an abundance of tools has been developed to study subclone composition in mixed samples [14–19]. Among the approaches that additionally reconstruct the evolutionary relationships, the majority separates subclone estimation and tree reconstruction [20–24], while others combine both tasks into a single step [25–27]. The typical output of these tools would be one or several trees as in Fig. 1d, augmented with the estimated prevalence of the different subclones in the tumor. Signals from multiple samples from different locations in the tumor increase the statistical power [24–27]. Samples taken from different time points are useful as well [28] but are usually not available for solid tumors, as biopsies are typically taken only once, at the point when the tumor is removed from the patient.

Approaches using mixed samples provide valuable insights into intra-tumor heterogeneity. However, their resolution is inherently limited and inference of both complex subclone structures and low-frequency subclones remains difficult [13, 29]. The advent of single-nucleus sequencing techniques has started to change the situation. Here, the taxa are known in the form of the individual cells sequenced from a tumor. However, the data we obtain from single-cell sequencing experiments are notoriously error-prone, in particular the false negative rate can be extremely high (≥10 %) due to the high allelic dropout rate in the DNA amplification process. The false positive rate is also elevated in comparison to bulk sequencing. Lastly, unobserved sites can be a problem. For example, 58 % of the data points are reported as missing due to low quality in an early single-nucleus sequencing data set [30] thus giving no information on whether the site is mutated or not in the respective cell. This combination of error types prohibits the application of standard perfect phylogeny reconstruction approaches. While generalizations of the perfect phylogeny problem to deal with imperfect data exist, they are typically NP-hard, and modify the input data in the binary mutation matrix, either by finding the minimum number of entries that need to be changed to remove all inconsistencies [31], or by removing the minimum number of samples (taxa) to remove all the inconsistencies [32].

Probabilistic approaches are an alternative to make use of all information contained in the (inconsistent) data. In addition, using a Bayesian scheme, the whole posterior tree distribution instead of just a single tree can be obtained and model parameters such as the error rates of the sequencing experiments can be learned. Bayesian approaches typically use polynomial-time Markov chain Monte Carlo (MCMC) sampling heuristics to explore a (super-)exponential search space.

A fully Bayesian approach is BitPhylogeny [33], which uses non-parametric clustering in combination with a tree-structured stick-breaking process to identify subclones and their evolutionary relationships. Unlike tree-based approaches for mixed samples, BitPhylogeny clusters samples into subclones and sets these in a phylogenetic relation (Fig. 1e).

Kim and Simon [34] introduced a pairwise ordering test for mutations in an attempt to find the best fitting tree from noisy and incomplete single-cell data [30]. Their approach reconstructs a mutation history as in Fig. 1d, also referred to as *mutation tree*. The restriction to pairwise tests results in an efficient polynomial-time algorithm but comes at the cost of a potential loss in reconstruction quality, as all information from more complex relations than pairwise order is discarded. Instead of using pairwise orders, one could consider testing the ordering of triplets of nodes and then higher groupings.

Here we propose a likelihood-based approach to test the entire mutation tree at once and perform a stochastic search to find the best fitting tree. We introduce SCITE (Single Cell Inference of Tumor Evolution), a flexible MCMC sampling scheme that allows us to compute the maximum likelihood (ML) tree plus attachment points of the samples, sample from their posterior, or treat mutation trees with the attachment points marginalized out. These can be combined with learning the error rates of the sequencing experiments. We evaluate SCITE on real cancer data, showing its scalability to present-day single-cell sequencing data and its improved results over BitPhylogeny [33], the approach of [34], classic perfect phylogeny reconstruction, and methods designed for bulk-sequencing data. In addition, we estimate from simulation studies the number of cells necessary for reliable mutation tree reconstruction, which could inform the design of future single-cell sequencing projects.

Results and discussion

Tree inference from single-cell mutation profiles

We first provide a brief description of our approach to tree inference from single-cell mutation profiles. We start with a model for representing single-cell mutation histories and the likelihood-based approach to deal with sequencing errors. Then we give an overview on the different variants of the MCMC sampling scheme implemented in SCITE. A more technical description of SCITE is in the "Methods" section.

Model of tumor evolution and tree representation

We restrict the evolutionary model to point mutations in this work and make the infinite sites assumption, which states that every genome position mutates at most once in the evolutionary history of a tumor. No further constraints are necessary, in particular no assumption on a monoclonal origin of the tumor is made, a core assumption in tree reconstruction from mixed samples.

We represent the mutation status of m single cells at n different loci in a binary $n \times m$ *mutation matrix* E where a 1, respectively a 0, at entry (i, j) denotes the presence, respectively the absence, of mutation i in cell j (Fig. 1c). With the exclusion of convergent evolution due to the infinite sites assumption, this matrix defines a perfect phylogeny of the single cells. This means that there exists a rooted binary tree with the cells as leaves in which every mutation can be placed on one edge such that the mutation status of every leaf equals the set of mutations on its path to the root (Fig. 1b). Mutations present in all cells can be removed from the data as their location in the tree is known. The same is true for mutations observed only in a single cell. These are directly associated with the cell and non-informative in the tree reconstruction. For example, the mutation matrix from Fig. 1c reduces to:

$$E = \begin{array}{c} \\ M_1 \\ M_2 \\ M_3 \end{array} \begin{array}{c} \begin{array}{ccccccc} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & s_7 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \end{array}, \qquad (1)$$

where we now represent the remaining three mutations as M_1, M_2, and M_3. In general, the binary tree defined by the matrix E will not be unique. In the example in Fig. 1b, since the three left-most leaves all have the same mutation status, their branching order in the tree is, therefore, arbitrary. Also the correct placement of the fourth leaf is not unique, as it has no mutation other than the ones shared by all samples. It could equally well branch off in the left subtree after the two ubiquitous mutations instead of the right one. A more compact tree representation of E is a *mutation tree* T, which represents the mutations as nodes and connects the nodes according to their order in the evolutionary history. An empty node is used to indicate the root (Fig. 1d). The mutation tree can be seen as the perfect phylogeny tree, where instead of placing the mutations along the edges we encapsulate them inside internal nodes. This slight change in representation facilitates our inference later. The mutation tree can be augmented with the sequenced cells by attaching them to the node that matches their mutation state (Fig. 1f). The order of mutations shared by the exact same set of cells is unidentifiable in the mutation tree, as is the case for the two top-most mutations in Fig. 1f. Such subsets of mutations are summarized in a single node, here highlighted as a shaded box.

Observational errors

In real data, we do not observe a perfect mutation matrix (Fig. 1c) but a noisy version of it (Fig. 1g), which we denote by D in the following. If the true mutation value is 0, we may observe a 1 with probability α (false positive), and if the true mutation value is 1, we may observe a 0 with probability β (false negative) such that

$$\begin{aligned} &P(D_{ij}=0|E_{ij}=0)=1-\alpha, \quad &&P(D_{ij}=0|E_{ij}=1)=\beta,\\ &P(D_{ij}=1|E_{ij}=0)=\alpha, &&P(D_{ij}=1|E_{ij}=1)=1-\beta. \end{aligned} \tag{2}$$

Assuming the observational errors are independent of each other, the likelihood of the data given a mutation tree T, knowledge of the attachment of the samples $\boldsymbol{\sigma}$, and the error rates $\boldsymbol{\theta}=(\alpha,\beta)$ is then

$$P(D|T,\boldsymbol{\sigma},\boldsymbol{\theta})=\prod_{i=1}^{n}\prod_{j=1}^{m}P(D_{ij}|E_{ij}), \tag{3}$$

where E is the mutation matrix defined by T and $\boldsymbol{\sigma}$.

For the posterior,

$$P(T,\boldsymbol{\sigma},\boldsymbol{\theta}|D)\propto P(D|T,\boldsymbol{\sigma},\boldsymbol{\theta})P(T,\boldsymbol{\sigma},\boldsymbol{\theta}), \tag{4}$$

we can factorize the prior, $P(T,\boldsymbol{\sigma},\boldsymbol{\theta})=P(\boldsymbol{\sigma}|T,\boldsymbol{\theta})P(T,\boldsymbol{\theta})$, and we assume independence of the error rates to set $P(T,\boldsymbol{\sigma},\boldsymbol{\theta})=P(\boldsymbol{\sigma}|T)P(T)P(\boldsymbol{\theta})$ so that the attachment prior $P(\boldsymbol{\sigma}|T)$ depends on T. Such a prior might be useful if one suspects that cells are more likely to be sampled from later stages in tumor development and lower down in the tree. Here though we use a uniform attachment prior.

MCMC sampling

Our model for learning mutation histories from single-cell mutation profiles consists of three parts: the mutation tree T, the sample attachment vector $\boldsymbol{\sigma}$, and the error rates of the sequencing experiment $\boldsymbol{\theta}$. The resulting search space has a continuous component for $\boldsymbol{\theta}$ and a discrete component of size $(n+1)^{(n-1)}(n+1)^m$ for $(T,\boldsymbol{\sigma})$, which prohibits an exhaustive search. Instead, with Eqs. 3 and 4 we built SCITE, a MCMC scheme to sample from the joint posterior given the data. From the current state $(T,\boldsymbol{\sigma},\boldsymbol{\theta})$, we propose a new state $(T',\boldsymbol{\sigma}',\boldsymbol{\theta}')$ with an ergodic mixture of moves where we change one component at a time. With properly defined transition probabilities and acceptance ratio, our chain converges to the posterior. In practice, we marginalize out the sample attachments in our model not only to speed up convergence but to focus on the mutation tree T as the informative part for understanding the mutation history. Thus,

$$P(T,\boldsymbol{\theta}|D)=\sum_{\boldsymbol{\sigma}}P(T,\boldsymbol{\sigma},\boldsymbol{\theta}|D). \tag{5}$$

We then only need to consider moves in the joint $(T,\boldsymbol{\theta})$ space, thereby reducing the search space by a factor of $(n+1)^m$. It is still possible to augment the tree with the samples in a post-processing step by sampling them conditionally on the tree.

After convergence, the MCMC chain can be used to sample trees and error rates proportionally to the joint posterior distribution in Eq. 4. In addition, it is possible to obtain a single best fitting combination of mutation tree and error rates via point estimates of the model parameters. One way of doing this is via maximum a posteriori (MAP) estimates:

$$(T,\boldsymbol{\theta})_{\mathrm{MAP}}=\arg\max_{(T,\boldsymbol{\theta})}P(T,\boldsymbol{\theta}|D). \tag{6}$$

Another possibility is to use ML estimates. Since the likelihood depends on the full set of model parameters $(T,\boldsymbol{\sigma},\boldsymbol{\theta})$, it is more natural to optimize them all jointly rather than marginalizing out the sample attachment:

$$(T,\boldsymbol{\sigma},\boldsymbol{\theta})_{\mathrm{ML}}=\arg\max_{(T,\boldsymbol{\sigma},\boldsymbol{\theta})}P(D|T,\boldsymbol{\sigma},\boldsymbol{\theta}). \tag{7}$$

In the ML framework, SCITE includes a parameter γ that amplifies the likelihood and which can speed up discovery of the ML tree.

Finally, SCITE provides an option to skip the learning of error rates when fixed error rates are provided. Since these are often available for sequencing data, they can be used instead to reduce the search space size.

Reconstructing mutation histories from real tumor data

For a first evaluation of SCITE, we applied it to three real single-cell tumor data sets of different data quality.

JAK2-negative myeloproliferative neoplasm

The first tumor data is single-cell exome sequencing data from a JAK2-negative myeloproliferative neoplasm (essential thrombocythemia) [30]. It originally consists of 712 SNVs detected in the exomes of 58 tumor cells. In our evaluation, we focus on the 18 mutation sites selected as cancer-related by [30]. The error rates of the sequencing were estimated as $\alpha=6.04\times10^{-6}$ (false positives) and $\beta=0.4309$ (false negatives, allelic dropout). In addition, the reduced set has 45 % missing data points (compared to 58 % in the full data set). The mutation matrix (Additional file 1: Figure S1a) is taken from [34]. It distinguishes three observed states: normal, heterozygous, and homozygous mutation. This means only that a homozygous mutation is observed, not that it is actually present in the data. The latter would contradict the infinite sites model that each site mutates at most once. Explanations consistent with infinite sites are that we either have a false negative for the normal copy of a heterozygous site, or less likely, a combination of a false positive and an allelic dropout for a site whose true state is homozygous normal. Another explanation for observing a homozygous mutation could be a loss of heterozygosity. We adapted our approach to integrate the third mutation state by using the same error probabilities as [34]. They assume that an allelic dropout is equally likely to cause a heterozygous mutation to be recorded as a normal state or as homozygous. Denoting heterozygous sites by 1 and homozygous sites by 2, this assumption results in the error probabilities:

$$\begin{aligned}
&P(D_{ij}=0|E_{ij}=0)=1-\alpha-\frac{\alpha\beta}{2}, &&P(D_{ij}=0|E_{ij}=1)=\frac{\beta}{2},\\
&P(D_{ij}=1|E_{ij}=0)=\alpha, &&P(D_{ij}=1|E_{ij}=1)=1-\beta,\\
&P(D_{ij}=2|E_{ij}=0)=\frac{\alpha\beta}{2}, &&P(D_{ij}=2|E_{ij}=1)=\frac{\beta}{2}.
\end{aligned} \tag{8}$$

Mutation tree reconstruction We computed the ML tree for the 18 mutation sites with SCITE. When optimizing tree and sample attachment, we obtain a mostly linear mutation tree with a single branching in the lower part of the tree (Additional file 1: Figure S2a) with a ML log score of −378.4.

We observe that quite a few samples are placed at nodes high up in the tree (Additional file 1: Figure S3), though many of these placements are uncertain, as indicated by the multiple co-optimal attachments. Taking into account the uncertainties due to the high error rates and the large number of missing values (45 %), it is not unexpected that many cells fit equally well to several neighboring nodes. The linear nature of the tree matches a sequential monoclonal development. The subclone expansion starting towards the bottom of the tree indicates the co-existence of multiple subclones at the point of sampling. However, from the single time point data, it is not possible to decide whether the more recent subclones are on the verge of replacing the more ancestral clones, or will coexist for longer.

Along with finding the ML tree with attachments, we performed a fully Bayesian sampling of trees and attachments from the posterior. To summarize such a sample, we consider as an example the number of branches the trees possess. The distribution for the data from [30] (Fig. 2a) shows that the trees mostly have a single branching point (with two branches) like the ML tree and often occur as a simple linear chain with a single branch.

Comparison to trees found with other approaches The same data have previously been analyzed with two competing methods [33, 34].

Kim and Simon [34] employ the same underlying likelihood with errors as in Eq. 8 but they use the data to learn ancestral relations between each pair of mutation nodes instead of the whole tree at once. They also use the data to learn a parameter representing how quickly the mutation tree branches. This parameter is then used to calculate the prior probability of ancestral relations, which is fed into their pairwise test and subsequent tree reconstruction.

With the data from [30] (on the same 18 selected mutations), [34] estimate that 92 % of the evolutionary time of the phylogenetic tree should be before the first binary split. In their model, this translates into expecting over 80 % of the mutations to occur before any branching in the mutation tree. Despite this very linear tree estimate, their algorithm to turn the pairwise ancestral relations into a mutation tree leads to the very branched tree in Additional file 1: Figure S2c, which has a much lower log-likelihood of −1059.7 than the ML tree found with SCITE (with a log-likelihood of −378.4). This may be due to the use of the minimum spanning tree algorithm by Kim and Simon. The method effectively needs to turn ancestral relations into strict parent–child relations and thereby, it essentially discounts the deeper history embedded in their pairwise tests.

We cannot compare directly to the tree found by BitPhylogeny [33] since their algorithm aims to find the phylogenetic connection between the samples themselves rather than the mutation tree. Furthermore, the algorithm groups samples into clones according to the data and a stick-breaking prior. For example, using all the mutation data from [30], as well as a bulk normal and bulk cancer sequence, and with a particular stick-breaking tree prior they find one large clone accounting for over half the samples and eight further smaller clones arranged in a tree structure [33]. However, we can view their result as a mutation tree with attachments where the mutations themselves have been censored. This leaves just the sample attachment information as well as the global tree structure between their groupings.

To build a complete mutation tree we allow each mutation to be placed before any one of the clonal groupings of samples (or completely afterwards). For each mutation, we find its ML position and hence find the ML tree (with attachments), which respects the result of [33]. The resulting tree (Additional file 1: Figure S2b) is a mostly linear chain like the ML tree SCITE finds and involves some of the same genes at the branches although one

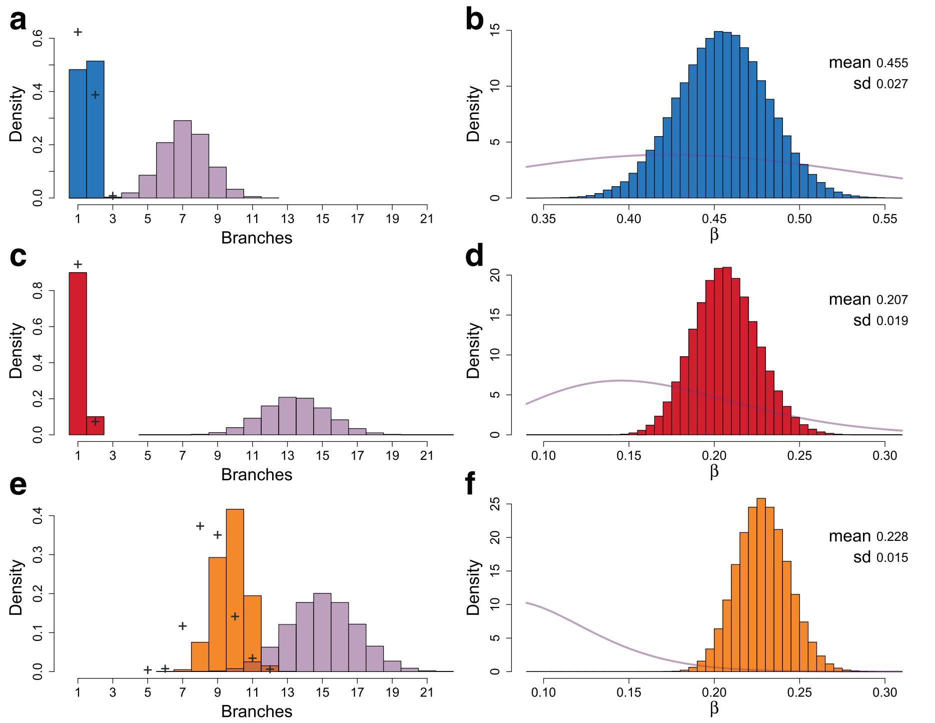


Fig. 2 The posterior tree branch and error distributions. The posterior distribution for the number of tree branches for the data from [30] in (**a**), for the data from [35] in (**c**), and for the data from [36] in (**e**), all with fixed false negative error rate β. The prior distributions from uniformly sampled trees are in *light purple*. The posterior distributions for β for the same data sets are given in (**b**), (**d**), and (**f**) with the priors included as *light purple lines*. When β is learned, the posterior distribution of the number of tree branches shifts slightly as indicated by the *black crosses* in (**a**), (**c**), and (**e**). *SD* standard deviation

of our branches is lost. The log-likelihood of −642.3 for this tree is substantially better than the tree of [34] but worse than the tree SCITE finds (with a log-likelihood of −378.4). With single-cell sequencing we can, as we do here, simply treat each cell as its own clone and discover the phylogeny directly. BitPhylogeny [33] instead focuses on clustering samples into subclones during tree inference thereby reducing the resolution of the reconstruction.

Error rate learning Within our Bayesian MCMC approach, we can also sample error rates from the posterior. Focusing on the false negative error rate β while keeping the false positive α fixed, for the beta prior on β with mean 0.4309, we chose a large standard deviation of 0.1. In the MCMC chain, with probability 10 % a new β' is proposed following a Gaussian random walk with standard deviation equal to one third of the prior's. Running the chain for 10 million steps, throwing away the first quarter, and plotting the resulting posterior of β we arrive at Fig. 2b. The posterior mean is 0.455 with standard deviation 0.027 so that the data indicates that the measured value of 0.4309 is a little underestimated but well within tolerances.

More interesting for our purposes is how these error rates affect the tree inference. The MAP β is 0.455 while the MAP tree (with attachments marginalized out) is a simple chain (Additional file 1: Figure S4). The mutation order is similar to the ML tree (Additional file 1: Figure S2a) up to the branching point suggesting a monoclonal tumor development. Keeping the error rate fixed at 0.4309 instead, we find an identical MAP tree giving us confidence that the inference is robust against minor differences in the error rates.

Mutation tree inference for a larger set of mutations

We also considered a larger set of mutations comprising all 78 non-synonymous mutations from the full data set. For this number of mutations, with only 58 sampled cells and high levels of missing data (48 %), the posterior is rather flat making discovering a global optimum rather than a local optimum more difficult. Increasing the parameter γ to 2–3 to amplify the likelihood landscape helped in discovering high-scoring trees. We also tested that the alternative tree representation (see "Methods") designed for instances with more mutations than samples aided in finding the ML tree (Additional file 1: Figure S5). The ML tree is again highly linear but the order especially of some of the 18 mutations varies compared to the ML tree inferred for that subset of the data

(Additional file 1: Figure S3). With missing data, the mutations may fit equally well along several edges and they were placed in their earliest position, which may explain some of the variation. More generally though, the high levels of missing data allow mutations and samples to move without affecting the likelihood while high error rates allow further rearrangements with only a small effect. For example, the mutation in the gene PDE4DIP that changes most between the two data sets has 59 % missing data. Also the order is essentially determined by the smaller number of samples that attach higher up the trees. This smaller number is effectively reduced further by the missing data, limiting the accuracy of any tree reconstruction, as explored later with the simulations.

Clear-cell renal-cell carcinoma

The second data set is from single-cell exome sequencing data of a clear-cell renal-cell carcinoma [35]. The mutation statuses of 50 sites in 17 tumor cells are detailed in the supplementary material of [35]. We marked the presence of an SNV when the call was different from the consensus of five normal tissue cells (in line with the totals provided in their supplementary material). As for the data from [30, 35] distinguish between heterozygous and homozygous mutations so we again use Eq. 8. Of the 50 sites, only 35 were not mutated in at least one cell. Only those were selected since the remaining 15 would simply be placed at the top of the mutation tree. The error rates were estimated by [35] as $\alpha = 2.67 \times 10^{-5}$ (false positives) and $\beta = 0.1643$ (false negatives) and the data also has 22 % missing entries (Additional file 1: Figure S1b).

Mutation tree reconstruction The ML and MAP trees both possess a completely linear accumulation of mutations (Additional file 1: Figures S6 and S7a), which is consistent with a series of monoclonal expansions and the conclusions of [35]. The linearity is confirmed in the full posterior distribution of trees with a linear chain being dominant (Fig. 2c). In addition, we observe that almost all of the samples are placed towards the end of the tree. Again a larger value of the parameter γ and the alternative tree representation sped up discovery of ML trees.

Error rate learning Fixing a beta prior for β with mean 0.1643 and standard deviation of 0.06 the posterior distribution of β was obtained by averaging over ten runs of 10 million steps (with a quarter as burn-in) (Fig. 2d). The posterior mean is a little larger at 0.207 with a standard deviation of 0.019 so the stated value is just within the uncertainties. The MAP value of β instead is a little closer at 0.198 while the MAP tree (Additional file 1: Figure S7b) is essentially identical to that with a fixed value of $\beta = 0.1643$ (Additional file 1: Figure S7a). The order of some of the higher mutations varies, however, since their exact placement hardly affects the posterior probability.

Estrogen-receptor positive (ER$^+$) breast cancer

The third data set is from single-nucleus exome sequencing of 47 tumor cells from an estrogen-receptor positive (ER$^+$) breast cancer [36]. Only two states are called for each site: the presence or absence of a SNV. Estimated error rates from [36] are 9.72 % for allelic dropout, and 1.24×10^{-6} for false discovery. In our analysis, we use the 40 mutations present in at least two tumor cells (Additional file 1: Figure S1c).

Mutation tree reconstruction The MAP tree computed for this data set is shown in Fig. 3. In the Supplement, we additionally show the ML tree (Additional file 1: Figure S8) and a version of the MAP tree with attached samples (Additional file 1: Figure S9a). In both the MAP and the ML trees, we observe a linear accumulation of mutations in the early stages of the tumor, suggesting that the development was through a sequential replacement of subclones with no surviving side branches and only a few cells with ancestral states surviving until present. In the later stages of the tumor, we observe a complex branching into co-existing subclones. This branching is exhibited more generally in the full posterior distribution of trees as summarized in Fig. 2e.

From the single time point data available for this tumor, it cannot be inferred whether there will be a long-term coexistence of subclones, or if we observe a transient state that will eventually lead to a single surviving subclone. For initial cancer treatment, however, the status quo, whatever mutations co-occur in cells, is already informative for jointly targeting the present subclones and therefore, minimizing the risk of further differentiation into therapy-resistant subclones.

Error rate learning Using a beta prior for β with mean 0.0972 and standard deviation of 0.04, we averaged over 20 runs of 10 million steps (with a quarter as burn-in) to obtain the posterior distribution of β (Fig. 2f). The posterior mean is more than double at 0.228 (with a standard deviation of 0.015), which disagrees with the stated value. This result is in contrast to our later simulations on learning the error rate (Fig. 4) that show that the MAP value is close to the true one. A possible explanation for the discrepancy is that allelic dropout only comprises one part of the false negative rate. Other contributing factors could include inaccuracies in calling heterozygous mutations at low coverage.

The MAP value of β is 0.226 with a MAP tree (Additional file 1: Figure S9b), which shares many feature with the MAP tree at fixed $\beta = 0.0972$ (Additional file 1: Figure S9a) but has some rearrangements of the branches

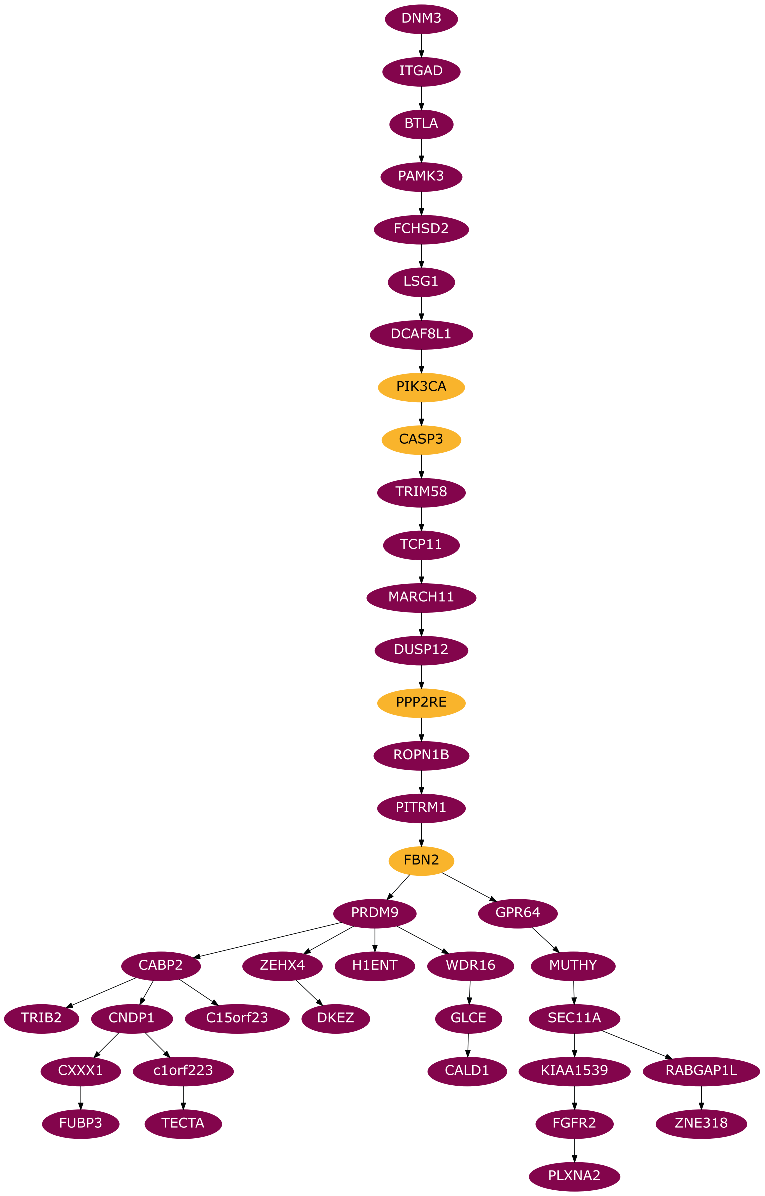


Fig. 3 MAP tree for the (ER$^+$) breast cancer for the [36] data. See Additional file 1: Figure S9a for a version with samples attached. *Yellow* genes indicate non-synonymous mutations in known cancer genes [36]

lower down and some reordering of the mutations higher up. Learning the error rate also leads to slightly fewer branches in the posterior distribution, as indicated by the black crosses in Fig. 2e.

Systematic evaluation of SCITE on simulated data

With the limited availability of single-cell sequencing data at this point and the lack of the ground truth in real data, we performed a more systematic evaluation of SCITE on simulated data sets. Our analysis focuses on the accuracy of tree inference and error rate learning, the effect of data quality, and the practical run times of SCITE.

Accuracy of tree inference

To check the consistency of our approach, we simulated random mutation trees with attachments uniformly, which allows for poly-clonal tree topologies. First, for $n = 20$ and $\alpha = 10^{-5}$, we generated 100 such trees with up to 100 attachments. For error rates $100\beta \in \{5, 15, 25\}$, for each tree we sampled from a lognormal with standard deviation 0.1 and multiplied it by β to obtain β^*. Then we added noise to the perfect data with rates (α, β^*) and removed 1 % of the data. Taking subsets of the data of size m, we learned the ML and MAP trees for the error rates β. This gives us

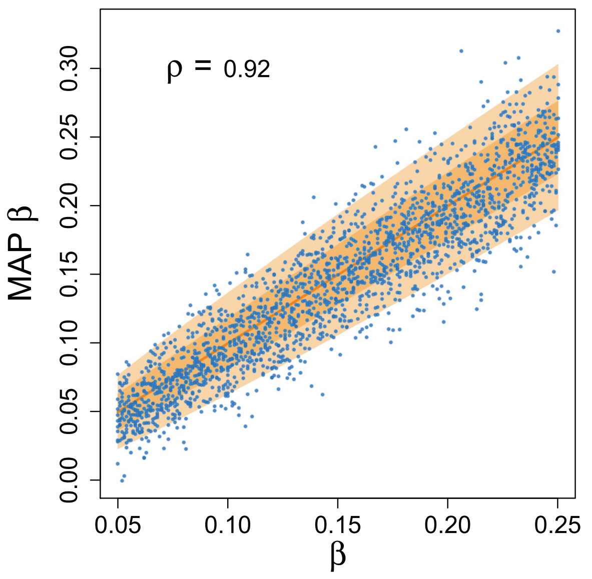


Fig. 4 Learning error rates. Comparison of the MAP false negative rate β learned using SCITE for $n = 20$ against the β used to generate the data. The *solid blocks* are one and two standard deviations of inferring β if the tree was known. *MAP* maximum a posteriori probability

a random misspecification of around 10 % compared to β^*.

We quantified the difference between the inferred trees and the true tree by counting how often a node has the wrong parent (Fig. 5 and the top row of Additional file 1: Figure S10). In the ML setting, if no samples are attached to a chain of mutations, then any ordering of those mutations has the same likelihood. Here, in the score we do not penalize this non-identifiability and take the ordering that minimizes the distance to the generating tree. The non-identifiability will, however, tend to decrease as the number of samples m increases. The MAP tree does select an ordering (roughly following the frequencies) and hence has higher distances than the ML tree. In general, MAP inference should be more robust and less prone to overfitting, but can have a higher bias. To compare the ML and MAP inference fairly, we chose a random ordering of the mutations in non-identifiable regions in the ML trees and recomputed the distances to the generating tree. We do observe a marginal improvement in the tree reconstruction with the MAP tree (Additional file 1: Figure S11).

The errors, however, are not a result of the inference method, since SCITE indeed finds the ML tree (Additional file 1: Figure S12). Instead these errors are inherent in noisy data where another tree might happen to fit the data better than the generating tree. The discrepancy can only be resolved by reducing errors or increasing the sample size and Additional file 1: Figure S10 gives an indication of how this occurs. To put the errors in scale, a value of two would refer to adjacent mutations in a chain being swapped. Since samples contain the mutations along their entire history in the mutation tree, we have a greater consensus about the mutation structure higher up the tree than lower down. The exact placement of mutations near the bottom of the tree may be determined by only a couple of samples so that the errors we typically see with larger m are mutations near the bottom of the tree being shifted, or two adjacent mutations being swapped. With this in mind, we obtain very good trees with about 60 samples, depending on the error rate.

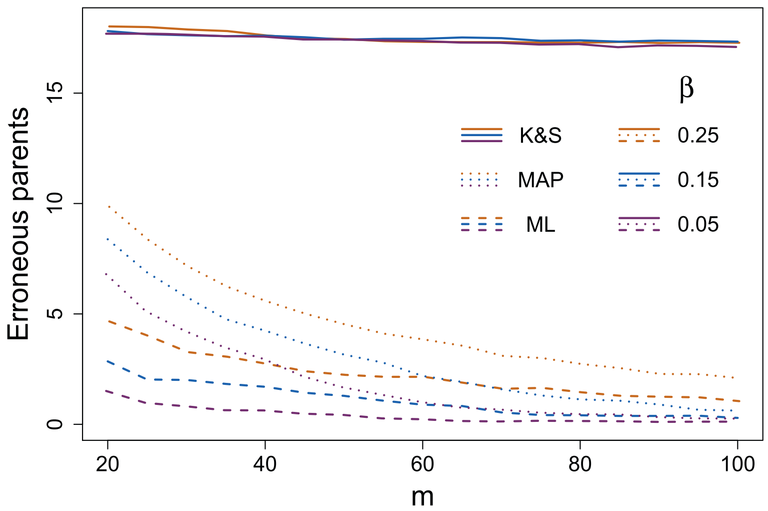


Fig. 5 Comparison of different methods. Comparison of the tree learning for $n = 20$ using SCITE for the ML tree (*dashed*) and MAP tree (*dotted*) against results from [34] (*solid lines*). The ML tree distances do not include non-identifiable regions. *K&S* Kim and Simon [34], *MAP* maximum a posteriori, *ML* maximum likelihood

We repeated the simulations for $n = 40$ and up to 200 attachments as depicted along the bottom row of Additional file 1: Figure S10 and again find good reconstruction when we have several samples per mutation.

Learning the error rates

Since SCITE can also perform fully Bayesian tree inference, we examined its ability to infer the false negative rate from data. For 2000 random trees with 60 attachments, we generated data with a range of β from 5 to 25 %, $\alpha = 10^{-5}$, and 1 % missing data. We further fixed a uniform prior for learning β so that no information is passed to SCITE apart from the noisy and incomplete mutation matrix.

There is very high correlation between the generating β and the MAP value learned (Fig. 4). To put this in context, we consider the theoretical distribution if the tree was known. From the random trees and attachments, around 22 % of the entries in the perfect mutation matrix are ones. They are randomly changed with the rate β, leading to a binomial distribution and a standard deviation of

$$\sqrt{\frac{100\beta(1-\beta)}{22mn}}$$

when inferring β from the result. One and two standard deviation intervals are included in Fig. 4, showing again that SCITE performs very well as it must also infer the tree structure and handle the missing data.

Similar plots for $m = 40$ and $m = 80$ (Additional file 1: Figure S13) show also a tightening of the β inference as m increases.

The effect of missing data

High rates of missing data points due to unobserved mutation states are typical for present-day single-cell sequencing data. We performed simulation experiments to test how this feature affects the accuracy of mutation tree reconstruction. With an error rate of $\beta = 10$ % and the same misspecification as before, we generated up to 400 random trees with up to 80 attachments. Keeping $\alpha = 10^{-5}$, we varied the amount of missing data from 1 to 20 % to see the effect on the tree reconstruction for $m = \{40, 60, 80\}$. We see a very weak increase in reconstruction errors as the missing data rate increases (top row of Additional file 1: Figure S14). Since SCITE treats the inference probabilistically, missing data is akin to effectively reducing the number of samples m, so the behavior in Additional file 1: Figure S14 is in line with changing m slightly in Additional file 1: Figure S10. The behavior also shows that SCITE is robust even against high missing data rates.

Looking back to the even higher missing data rates in the earliest data sets, we simulated up to 60 % missing data with 400 trees and the same settings as before. The reconstruction progressively gets worse with increasing missing data (bottom row of Additional file 1: Figure S14). At around 30–40 % missing data with 80 attachments, we have similar performance as for 40 cells attached with no missing data, and so have effectively halved our sample size. With 60 % missing data, the reconstruction is notably poorer again, although SCITE does find about half the parents correctly for the MAP solution and a large majority with the ML approach. This difference is because the optimal order is chosen for the ML solutions in case of non-identifiability.

Doublet samples

Rarely, instead of isolating a single cell for sequencing, a pair of cells is captured instead. We checked how robust SCITE is to these sorts of perturbations by again simulating data from 400 random trees with 20 nodes and up to 100 attachments. To represent the sequences of doublet samples, we took up to 20 pairs of attached samples and combined them by recording a mutation whenever it was present in either of the original single cells. Errors were added with a rate of $\beta = 10$ % (misspecified as previously), $\alpha = 10^{-5}$, and 1 % missing data. We ran SCITE with $m = \{40, 60, 80\}$ total samples, including up to 20 doublets, to see their effect on the tree reconstruction.

We observe a linear increase in reconstruction errors as the number of doublets increases (Additional file 1: Figure S15) with decreasing gradient as m increases since then the doublets represent a smaller proportion of the total sample. Unlike missing data, which reduces the effective sample size, doublets add confounding mutations, which could disagree with the tree topology. However, since SCITE employs probabilistic inference, and at the level of the mutation tree rather than the sample tree, the consensus of the single-cell samples moderates the negative effects of the doublets. Even at high rates of doublet sampling, like 10 or 20 %, the tree reconstruction, therefore, performs well.

Run times

To uncover the complexity of the stochastic search and MCMC scheme, we simulated data from 400 uniformly sampled trees with up to 100 nodes and 400 attached samples. We set $\alpha = 10^{-5}$ and $\beta = 0.1$ (with the same misspecification as before), included 1 % missing data and set the parameter $\gamma = 1$ as for the MCMC case. For each tree, we ran SCITE 100 times and recorded how many steps the algorithm took to first hit the highest likelihood tree uncovered by that run, as well as the time of the run. The lengths of the chains were chosen so that nearly all of the runs would share the same highest likelihood. The average number of steps needed to first find the consensus ML tree can then be calculated (for those runs with a lower likelihood, we add the length of the chain and then assumed they would find the ML tree in an additional average number of steps). This can then be multiplied by the average time per step to give a measure of how long SCITE takes to find a ML tree on average, and repeated for all 400 trees to provide Fig. 6.

On the theoretical side, arguments analogous to those in [37] indicate that the MCMC chain requires $O(n^2 \ln(n))$ steps to converge or find ML trees. The likelihood landscape may also depend on n and m in non-trivial ways, which can further affect the convergence. With each MCMC step taking $O(mn)$ to score the tree, we get an overall estimate of $O(mn^3 \ln(n))$ for convergence.

Compared to the numerical results in Fig. 6, the gradients in the log–log plots are 4.5, 4.5, and 4.2 for $m = \{n, 2n, 4n\}$ respectively. Since $m \sim n$ in the simulations, these are a little higher than the power of 4 suggested by the estimate, but roughly in line with it. To check the linear scaling with m, we take the fit lines at $n = 60$ in the middle of the simulation and we find that doubling m from n to $2n$ and then $4n$ increases the time by a factor of 1.9 and then 1.95, slightly less than double and in line with linear scaling. With linear scaling in m, and for a reasonable number of mutations, SCITE will, therefore, be able to handle large numbers of sampled cells efficiently.

Further parameters with influence on the practical performance of SCITE are the move probabilities and for ML tree discovery, additionally the parameter γ. We performed a systematic search for the optimal parameters, which is described in Additional file 1. Our observation is that an optimal choice of move probabilities gives a constant factor speed-up compared to default values. Similar results were observed for γ, for which the optimum for finding a ML tree quickly is just below 1, the value required for the MCMC sampling.

Comparison with competing approaches

To assess further the performance of SCITE, we compared it to a simple perfect phylogeny approach, two methods designed for single-cell data, and two recent methods for tree inference from bulk-sequencing data.

Perfect phylogeny

We first compared SCITE against a simple algorithm for solving the perfect phylogeny problem (i.e. testing whether the data defines a phylogeny, and if it does to construct one [12]). A mutation matrix has a perfect phylogeny if a tree can be constructed such that the leaves are the samples and the mutations are each placed at exactly one edge, such that for every leaf the mutations on the path leading to it from the root reflect its mutation status. Such a tree exists only if there are no contradictions in the data due to noise or recurring mutations. But if it exists, it can be represented as a mutation tree by labeling nodes instead of edges. To test for perfect phylogeny, we use a version of the data with no missing values. From our simulated trees and data, only very few are free of contradictions, which limits the tree comparison to a few instances. The perfect phylogenies on average deviate more from the true tree than both ML and MAP trees and none is found for instances with more than 45 samples. The differences between the perfect phylogeny and the true tree are due to both the errors introduced and insufficient information to fully reconstruct the tree. Details of the comparison are given in Additional file 1: Table S1.

The approach of Kim and Simon [34]

The method in [34] reconstructs the same type of mutation trees as our approach. However, in their approach, a parameter representing how quickly the mutation tree branches is first learned from the data. This parameter is then used to calculate the prior probability of ancestral relations, which informs a pairwise ordering test and subsequent tree reconstruction. Instead of learning the parameter from the data, we give their method the exact value from the tree that was actually used to generate the data since this simplifies running the simulation test. Of

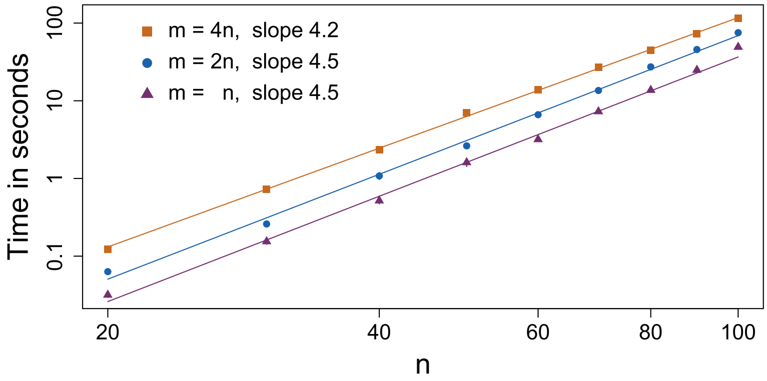


Fig. 6 Scaling behavior. The average time taken for SCITE to first find a ML tree as the number of mutations n in the tree is varied along with the number of attached samples $m = \{n, 2n, 4n\}$

course, in practice, this piece of information would not be available so the results from their algorithm are over optimistic. Nevertheless, the pairwise approximation performs comparatively poorly (Fig. 5). In particular, there is little improvement as the number of samples increases. Although the pairwise ancestral tests will become more accurate, this additional information appears to have little impact on the conversion to a mutation tree.

Comparison with BitPhylogeny

More advanced probabilistic inference is provided by BitPhylogeny [33]. This method, however, reconstructs a hierarchical subclone structure rather than a mutation tree thereby precluding a direct comparison to SCITE and the approach of [34]. Therefore, we convert the outcome of each method into a complete mutation tree with samples attached. For SCITE, this means finding the ML tree with attachments. For the approach of [34], we place the samples at their best fitting position on the tree found. For BitPhylogeny instead, we place the mutations along the branches of their clonal tree in the position that maximizes the likelihood. Since the mutations and samples may be grouped together, as a measure of fit we use the consensus node-based shortest path distance (as defined in [33]) between the (completed) inferred tree and the generating tree. In particular, for each tree, the pairwise shortest distance between any two samples is their number of differing mutations. We then normalize by averaging over the absolute differences between the pairwise distances in the inferred and generating trees, rather than taking the sum.

For $n = 20$, $\alpha = 10^{-5}$, and $\beta = 0.1$ (with the same misspecification as before), we generated 400 such trees with 1 % missing data. For simplicity and giving BitPhylogeny a slight advantage, we passed it the complete data. The results for $m \in \{40, 60, 80\}$ are presented in Fig. 7. The methods compared perform significantly more poorly than SCITE, with BitPhylogeny [33] performing better than the algorithm of [34], but with neither approaching the performance of SCITE.

We can also compare the performance of the different methods in terms of the difference in log-likelihoods between the inferred and generating trees, normalized by dividing by the number of data matrix elements (Additional file 1: Figure S12). This shows similar behavior to Fig. 7 and we observe that SCITE always provides a non-negative difference. SCITE, therefore, always found either the generating tree or one with a slightly higher likelihood than the generating tree.

Comparison with bulk-sequencing methods

Finally, we compared SCITE to methods designed for deconvolution and tree reconstruction from mixed bulk sequences. We chose PhyloWGS [24] and AncesTree [22] as two recent high-performing methods that allow samples to be treated separately as well as combined. PhyloWGS employs a stick-breaking tree prior (like BitPhylogeny) while AncesTree solves the deconvolution and ancestry as a matrix factorization. When passing the simulated single-cell mutations as individual samples to both methods, neither returned anything other than a single grouping of mutations. A possible explanation for this result is that the two methods interpret the binary mutation states as cellular prevalence in mixed samples, which likely causes trouble in the deconvolution step. Better performance was obtained when combining the single cells into a bulk mixture, with both methods returning mutation trees with the mutations possibly grouped together at the nodes. To compare with the other methods, we again placed the samples at their best positions in the inferred trees to obtain the results in Fig. 7. AncesTree performs slightly worse than PhyloWGS and both are notably worse than BitPhylogeny and SCITE. This is not unexpected, as only the latter two are designed to handle single-cell

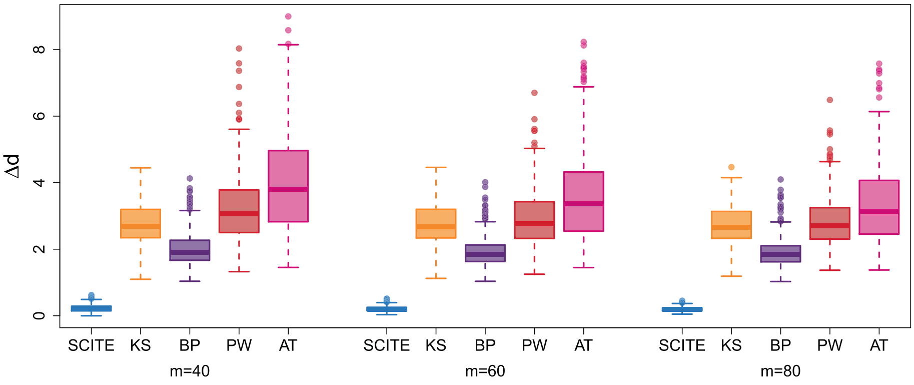


Fig. 7 Comparison of additional methods. Comparison of the tree inference of SCITE, the algorithm of [34], BitPhylogeny [33], PhyloWGS [24], and AncesTree [22]. The quantity Δd is the normalized consensus node-based shortest path distance (as defined in [33]) between the inferred and generating trees. *AT* AncesTree, *BP* BitPhylogeny, *KS* Kim and Simon [34], *PW* PhyloWGS

data. The main conclusion here is that specialized methods are necessary for single-cell data as approaches for mixed samples are not readily applicable.

Conclusions

Single-cell sequencing data is giving unprecedented insights into intra-tumor heterogeneity, a major obstacle to permanent remission in cancer treatment. In this paper, we introduced SCITE, a likelihood-based reconstruction of tumor genealogies from noisy and incomplete mutation profiles of single cells. The approach provides a flexible MCMC sampling scheme that allows us to either find the best fitting tree or sample from the posterior distribution, and it can be combined with learning the error rates of the sequencing experiments. We have shown that the probability model underlying SCITE is highly adaptable. It performs well in the presence of various types of noise, including types that were not explicitly modeled, such as doublet samples (the inadvertent sequencing of two instead of a single cell). The model also lends itself to some straightforward extensions, such as the incorporation of position-specific error rates, or the introduction of further mutation and error types that would maintain the independence of genome positions. Besides its flexibility, the key advantage of SCITE is its linear scaling with the number of samples. While this feature is negligible for present data sets, it will become essential as soon as hundreds or even thousands of cells of a tumor are routinely sequenced.

Using SCITE, we reconstructed the monoclonal origin of an ET tumor and a clear-cell renal-cell carcinoma, as well as a complex subclonal structure in an ER$^+$ breast cancer. The consistency of SCITE is shown in simulation studies, which we also used to estimate the number of cells necessary to obtain reliable tree reconstructions, a piece of information that could be useful in the design of future sequencing experiments.

SCITE differs from earlier approaches, in particular BitPhylogeny [33], in its use of single cells as taxonomic units, giving it the highest possible resolution in the tree reconstruction. Because each cell provides information about all the mutations, and all this detail is used, this approach allows a more robust reconstruction of the mutation tree. This in turn aids the identification of driver mutations. The placement of the individual cells is, however, less certain. Clustering cells into clones instead, as done in BitPhylogeny [33], and placing these as the taxa means we can use the consensus of single-cell information in each clone to deduce more robustly the ancestral relationships between the clones themselves, but at the expense of reduced accuracy in the reconstruction of the mutational history.

Further improvements in tree reconstruction could be achieved by considering copy number alterations along with point mutations. For one, copy number information could be used to understand point mutations states better, e.g. a seemingly homozygous mutation may in fact be loss of heterozygosity, but more importantly it can be used as a feature in tree reconstruction itself, as has been done previously for bulk-sequencing data [24]. The main challenges here will be that for large-scale copy number events, the independence of mutation sites is no longer given, and that the infinite sites assumption would no longer hold.

The knowledge of individual mutation histories is a promising source of information for personalized cancer treatment. Once single-cell sequencing has become more prevalent, the unprecedented resolution of mutation histories reconstructed from single cells will likely be valuable in many more respects. One direction is the identification of recurrent mutation patterns by comparing high-resolution mutation trees from patients with the same and/or different tumor types. Another direction could be to combine single-cell data from different time points and different locations in the tumor to obtain a better understanding of the temporal and spatial organization of subclonal populations of tumor cells, again at a higher resolution than would be possible with bulk-sequencing data. When sampling cells from the primary tumor and metastasis, the attachment point of metastatic cells to the mutation tree could help to answer the open question of whether subclones with the potential to metastasize arise early or late in tumor development.

Methods

Mutation trees

In SCITE, we represent a rooted mutation tree T over n mutations as an augmented ancestor matrix $A(T)$ where every node is considered an ancestor of itself:

$$A_{ik} = \begin{cases} 1, & \text{if } i = k \text{ or } i \text{ is an ancestor of } k, \\ 0, & \text{elsewise.} \end{cases} \tag{9}$$

For example, the augmented ancestor matrix for the tree in Fig. 1d, reduced to the mutation matrix given in 1 is

$$A = \begin{array}{c} \\ M_1 \\ M_2 \\ M_3 \end{array} \begin{array}{c} \begin{array}{cccc} M_1 & M_2 & M_3 & R \end{array} \\ \left(\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{array} \right) \end{array}, \tag{10}$$

where R represents the root of the mutation tree. The cells are attached to T such that the path to the root spells out their mutation status. This placement is denoted by a vector $\boldsymbol{\sigma}$, which records at the jth position the attachment point of sample j. Carrying on the example from Eq. 1 and Fig. 1d, we have $\boldsymbol{\sigma} = (1, 1, 1, 4, 3, 3, 2)$ where 4 represents the root. The connection between the mutation matrix and the mutation tree is

$$(E_{ij}|T,\boldsymbol{\sigma}) = A(T)_{i\sigma_j}. \tag{11}$$

This simply means that for a given tree and sample attachment, the mutation status of a sample is identical to the one observed in the node where the sample attaches to the tree. Therefore, the likelihood in Eq. 3 can be rewritten as

$$P(D|T,\boldsymbol{\sigma},\boldsymbol{\theta}) = \prod_{i=1}^{n}\prod_{j=1}^{m} P(D_{ij}|A(T)_{i\sigma_j}) \tag{12}$$

and thereby computed directly from T and $\boldsymbol{\sigma}$.

MCMC sampling

In principle, the MCMC of SCITE needs three types of moves to separately alter the tree T, the attachment vector $\boldsymbol{\sigma}$, and the error rates. In fact, it is possible to marginalize out the $\boldsymbol{\sigma}$ component, such that we only need to consider moves in the joint $(T,\boldsymbol{\theta})$ space. We first focus on the marginalization and then describe the remaining move types.

Marginalization of the sample attachment

A move where we pick a sample and a new parent for it uniformly would satisfy the necessary properties for the MCMC chain on $\boldsymbol{\sigma}$ to converge, but we can achieve convergence much faster. This is because the likelihood in Eq. 12 factorizes into a product for each sample to be attached. As long as the prior $P(\boldsymbol{\sigma}|T,\boldsymbol{\theta})$ can also be factorized (so that the attachment for each sample is independent of the others), we can include the priors as in Eq. 4 and efficiently sum Eq. 12 over $\boldsymbol{\sigma}$ to marginalize it out:

$$\begin{aligned}\frac{P(T,\boldsymbol{\theta}|D)}{P(T,\boldsymbol{\theta})} &\propto \sum_{\boldsymbol{\sigma}}\prod_{j=1}^{m}\left[\prod_{i=1}^{n} P(D_{ij}|A(T)_{i\sigma_j})\right] P(\boldsymbol{\sigma}_j|T,\boldsymbol{\theta}) \\ &= \prod_{j=1}^{m}\sum_{\sigma_j=1}^{n+1}\left[\prod_{i=1}^{n} P(D_{ij}|A(T)_{i\sigma_j})\right] P(\boldsymbol{\sigma}_j|T,\boldsymbol{\theta}).\end{aligned} \tag{13}$$

Computation of Eq. 13 is in $O(mn)$ time due to the tree structure underlying $A(T)$. Along with efficient computation of Eq. 13, we now need only search over the $(n+1)^m$ times smaller space of trees T and error rates $\boldsymbol{\theta}$ leading to much faster MCMC convergence. This marginalization is equivalent to grouping all attachments to the same tree into a single object, which is responsible for the similarly large speed-up for sampling Bayesian networks in order MCMC [38] and more recently partition MCMC [39]. Analogously, nested effect models average over all effects with uniform prior [40].

With the attachments marginalized out, we need to consider only moves in the joint $(T,\boldsymbol{\theta})$ space. We can change one component at a time to propose a new pair $(T',\boldsymbol{\theta}')$ with transition probabilities $q(T',\boldsymbol{\theta}'|T,\boldsymbol{\theta})$ and accepting moves with the ratio

$$\rho = \min\left\{1, \frac{q(T,\boldsymbol{\theta}|T',\boldsymbol{\theta}')P(T',\boldsymbol{\theta}'|D)}{q(T',\boldsymbol{\theta}'|T,\boldsymbol{\theta})P(T,\boldsymbol{\theta}|D)}\right\} \tag{14}$$

to sample proportionally to $P(T,\boldsymbol{\theta}|D)$. Once we have sampled a tree, we can easily sample each attachment independently following Eq. 13.

Tree moves

Akin to standard MCMC approaches on graphical structures [41], we build a scheme on rooted mutation trees for fixed errors $\boldsymbol{\theta}$ as follows. Given the current tree T, we find the neighborhood of all trees reachable with the MCMC move from T. One then samples a tree T' from this neighborhood with some proposal probability $q(T',\boldsymbol{\theta}|T,\boldsymbol{\theta})$ and accepts the move with the probability in Eq. 14.

As long as the moves satisfy reversibility (that is, if the move from T to T' can be proposed with a non-zero probability, the reverse move from T' to T also has non-zero probability to be proposed), irreducibility (that is, a sequence of moves exists that leads from any tree to any other), and aperiodicity (which can be ensured by including the tree T in its neighborhood or adding a non-zero probability not to move), once the chain converges this scheme would allow us to sample trees proportionally to $P(T,\boldsymbol{\theta}|D)$.

The basic MCMC move we use is *prune and reattach*. We sample a node i uniformly from the n available and cut the edge leading to this node to remove the subtree from the tree. Then we sample one of the remaining nodes (including the root) uniformly and attach the subtree there instead. An example is illustrated in Fig. 8.

The reverse move, where we again sample i first but then pick its old parent, has the same proposal probability $q(T,\boldsymbol{\theta}|T',\boldsymbol{\theta}) = q(T',\boldsymbol{\theta}|T,\boldsymbol{\theta})$ since the non-descendant set has the same size each time i is removed. This term then drops from Eq. 14 and need not be calculated. Since we can also choose the old parent when sampling a new one, this move has a non-zero probability of proposing the same tree T, ensuring aperiodicity. There is also a path from any tree to a tree with all nodes attached to the root, by moving each node to the root step by step. Via reversibility, we can likewise move from there to any other tree ensuring irreducibility. The *prune and reattach* move, therefore, suffices to sample trees according to their posterior. To speed up the convergence of the chain, we use two additional moves in our MCMC scheme: *swap nodes* to swap the labels of two nodes and *swap subtrees* to swap two subtrees. (See Additional file 1 for more details.) One of the three moves is picked at each step of the chain with a fixed probability.

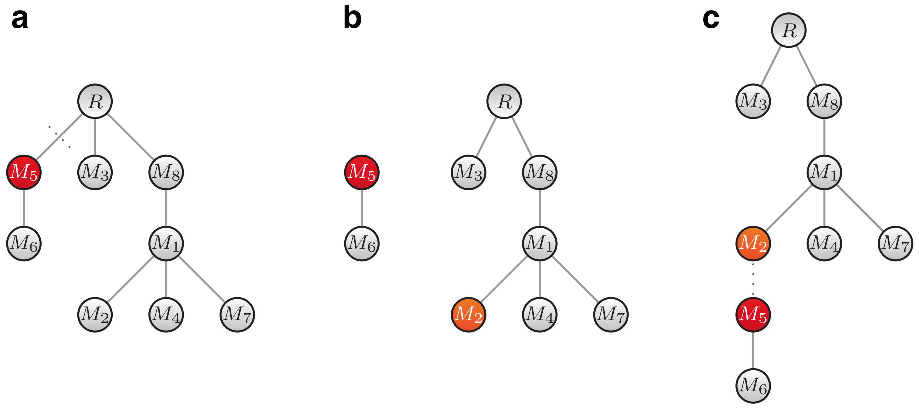


Fig. 8 Prune and reattach MCMC move. **a** From our starting tree *T*, we first select a node uniformly, here M_5, and detach it from the rest of the tree. **b** Then we sample one of the remaining nodes in the rooted tree; here M_2 is chosen. **c** Finally, we attach the detached subtree formed of M_5 and its descendants to the newly selected node M_2

Error learning

Where estimates of the error rates are known, we can input this information into the prior $P(\boldsymbol{\theta})$. Since $\boldsymbol{\theta}$ is between 0 and 1, we choose a beta prior with mean equal to the known estimates and a large standard deviation to be weakly informative. Although for a given $(T, \boldsymbol{\sigma})$ we can marginalize out $\boldsymbol{\theta}$ analytically, this interferes with the speed-up in Eq. 13. Instead, to move in the error space, we choose a simple Gaussian random walk with fixed standard deviations in each direction and centered on the current value $\boldsymbol{\theta}$.

Maximum likelihood tree

Along with utilizing our MCMC scheme to perform fully Bayesian tree inference, we can adapt the method to search for the ML tree as well. In the ML framework, we consider the full space of trees with attachments $(T, \boldsymbol{\sigma})$ and find the best scoring pair, rather than summing over the attachment points. Keeping the error rates $\boldsymbol{\theta}$ fixed for simplicity, when we wish to search for the ML tree with attachments, after maximizing over all possible placements, we can define the following score for each tree:

$$S(T) = P(D|T, \boldsymbol{\sigma}^*) , \qquad \boldsymbol{\sigma}^* = \arg\max_{\boldsymbol{\sigma}} P(D|T, \boldsymbol{\sigma}). \tag{15}$$

Since the likelihood in Eq. 12 factorizes, we can find the best attachment for each sample independently of the others,

$$\sigma_j^* = \arg\max_k \prod_{i=1}^{n} P(D_{ij}|A(T)_{ik}), \tag{16}$$

by running over the columns of $A(T)$ and comparing to the observed data with the error rates as in Eq. 2. If several placements provide the same maximum, any may be selected for calculating $S(T)$, which is then

$$S(T) = \max_{\boldsymbol{\sigma}} P(D|T, \boldsymbol{\sigma}) \tag{17}$$

and which again involves only $O(mn)$ simple operations.

Now we can turn our attention to finding the ML tree:

$$T^* = \arg\max_T S(T), \tag{18}$$

which because of Eq. 17 is the tree that maximizes the likelihood in Eq. 12. The number of rooted trees with $(n+1)$ nodes (including the root) grows factorially so an exhaustive search becomes infeasible for more than ten nodes or so.

Instead we can reuse our MCMC scheme on the space of rooted mutation trees where given the current tree T we propose a tree T' according to one of the three move types with the same proposal probability $q(T'|T)$ but now accept the move with probability

$$\rho = \min\left\{1, \frac{q(T|T')S(T')^\gamma}{q(T'|T)S(T)^\gamma}\right\}. \tag{19}$$

The power of γ here is a way to flatten the distribution (for $\gamma < 1$) or make it more pronounced (for $\gamma > 1$). Simulated annealing would involve running the chain while simultaneously increasing $\gamma \to \infty$ to end up in a local maximum. Here, instead, we chose a value (depending on the data) for this parameter that allows fast discovery of the maximally scoring tree and simply run many chains, recording the maximally scoring tree we encounter.

In the Bayesian framework, we can search for the MAP tree. Including the prior on all discrete components and updating $S(T)$ accordingly, we would find the joint MAP tree and attachments with the scheme here. Averaging out the attachments instead, we can search just for the MAP tree as well. In particular, we replace $S(T)$ by $P(T, \boldsymbol{\theta}|D)$ in

Eq. 19. We can also find jointly maximal trees and error rates by putting the error moves back in.

Alternative representation for ML discovery

For the ML tree with attachments, since the optimal placement for each attachment can be easily found, we are left to search over all rooted trees with $(n+1)$ nodes. However, when $m \lesssim n$, we may return to the binary genealogical tree (Fig. 1b) with m sampled cells as leaves and $(m-1)$ internal binary divides. The mutations are placed along the edges and they are present in all cells further down that lineage. For a given genealogical tree with leaves, the optimal placement of every mutation along the edges is simple to compute. Each tree is assigned a score corresponding to the likelihood of the data given the optimal placement of the mutations, which can again be calculated in $O(mn)$ time. The binary tree with the highest score is then the ML binary genealogical tree that directly provides the ML mutation tree with attachments when we change the representation back to mutations trees (Fig. 1f).

We can search the binary tree space with analogous moves as for the mutation trees. A *prune and reattach* move can be performed by detaching one half of any internal binary divide (the remaining neighboring edges join together) and reinserting it into any of the edges then present. We can also *swap leaf labels*. The size of the relevant binary tree space is

$$\frac{m!\,(m-1)!}{2^{m-1}},$$

which may be smaller than the mutation tree space of $(n+1)^{(n-1)}$, or easier to search, allowing the ML tree to be discovered more quickly. This alternative representation is implemented in the SCITE software package.

In the binary tree space, one can further marginalize, but this is over the mutation placement rather than the sample attachments and the resulting posterior distribution does not translate directly into one over the mutation tree space.

Software availability

SCITE has been implemented in C/C++ and is freely available under a GPL3 license at https://github.com/cbg-ethz/SCITE.

Availability of data and materials

The published data sets analyzed are available from the supplementary material of [30, 35] and in Fig. 2f of [36]. The data matrices are also included with SCITE.

Ethics approval

Ethics approval is not applicable for this study.

Additional file

Additional file 1: Supplementary figures, a supplementary table, and a description of the additional MCMC moves and their effect on convergence. (PDF 970 kb)

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

KJ and JK designed and implemented the method. KJ, JK, and NB wrote the manuscript. All authors read and approved the final manuscript.

Funding

KJ was supported by SystemsX.ch RTD Grant 2013/150 (http://www.systemsx.ch/). JK was supported by ERC Synergy Grant 609883 (http://erc.europa.eu/).

Received: 11 November 2015 Accepted: 8 April 2016
Published online: 05 May 2016

References

1. Nowell PC. The clonal evolution of tumor cell populations. Science. 1976;194:23–8.
2. Merlo LM, Pepper JW, Reid BJ, Maley CC. Cancer as an evolutionary and ecological process. Nat Rev Cancer. 2006;6:924–35.
3. Pepper JW, Scott Findlay C, Kassen R, Spencer SL, Maley CC. Synthesis: cancer research meets evolutionary biology. Evol Appl. 2009;2:62–70.
4. Nik-Zainal S, Van Loo P, Wedge DC, Alexandrov LB, Greenman CD, Lau KW, Raine K, Jones D, Marshall J, Ramakrishna M, et al. The life history of 21 breast cancers. Cell. 2012;149:994–1007.
5. Yates LR, Campbell PJ. Evolution of the cancer genome. Nat Rev Genet. 2012;13:795–806.
6. Greaves M, Maley CC. Clonal evolution in cancer. Nature. 2012;481: 306–13.
7. Ding L, Ley TJ, Larson DE, Miller CA, Koboldt DC, Welch JS, Ritchey JK, Young MA, Lamprecht T, McLellan MD, et al. Clonal evolution in relapsed acute myeloid leukaemia revealed by whole-genome sequencing. Nature. 2012;481:506–10.
8. Gillies RJ, Verduzco D, Gatenby RA. Evolutionary dynamics of carcinogenesis and why targeted therapy does not work. Nat Rev Cancer. 2012;12:487–93.
9. Ortmann CA, Kent DG, Nangalia J, Silber Y, Wedge DC, Grinfeld J, Baxter EJ, Massie CE, Papaemmanuil E, Menon S, et al. Effect of mutation order on myeloproliferative neoplasms. N Engl J Med. 2015;372:601–12.
10. Stratton MR, Campbell PJ, Futreal PA. The cancer genome. Nature. 2009;458:719–24.
11. Swanton C. Intratumor heterogeneity: evolution through space and time. Cancer Res. 2012;72:4875–82.
12. Gusfield D. Algorithms on strings, trees and sequences: computer science and computational biology. Cambridge: Cambridge University Press; 1997.
13. Navin NE. Cancer genomics: one cell at a time. Genome Biol. 2014;15:452.
14. Gerstung M, Beisel C, Rechsteiner M, Wild P, Schraml P, Moch H, Beerenwinkel N. Reliable detection of subclonal single-nucleotide variants in tumour cell populations. Nat Commun. 2012;3:811.
15. Shah SP, Roth A, Goya R, Oloumi A, Ha G, Zhao Y, Turashvili G, Ding J, Tse K, Haffari G, et al. The clonal and mutational evolution spectrum of primary triple-negative breast cancers. Nature. 2012;486:395–9.
16. Roth A, Khattra J, Yap D, Wan A, Laks E, Biele J, Ha G, Aparicio S, Bouchard-Côté A, Shah SP. Pyclone: statistical inference of clonal population structure in cancer. Nat Methods. 2014;11:396–8.
17. Ha G, Roth A, Khattra J, Ho J, Yap D, Prentice LM, Melnyk N, McPherson A, Bashashati A, Laks E, et al. Titan: inference of copy number architectures in clonal cell populations from tumor whole-genome sequence data. Genome Res. 2014;24:1881–93.
18. Oesper L, Mahmoody A, Raphael BJ. Theta: inferring intra-tumor heterogeneity from high-throughput dna sequencing data. Genome Biol. 2013;14:80.

19. Zare H, Wang J, Hu A, Weber K, Smith J, Nickerson D, Song C, Witten D, Blau CA, Noble WS. Inferring clonal composition from multiple sections of a breast cancer. PLoS Comput Biol. 2014;10:003703.
20. Strino F, Parisi F, Micsinai M, Kluger Y. Trap: a tree approach for fingerprinting subclonal tumor composition. Nucleic Acids Res. 2013;41: 165–5.
21. Hajirasouliha I, Mahmoody A, Raphael BJ. A combinatorial approach for analyzing intra-tumor heterogeneity from high-throughput sequencing data. Bioinformatics. 2014;30:78–86.
22. El-Kebir M, Oesper L, Acheson-Field H, Raphael BJ. Reconstruction of clonal trees and tumor composition from multi-sample sequencing data. Bioinformatics. 2015;31:62–70.
23. Qiao Y, Quinlan AR, Jazaeri AA, Verhaak RG, Wheeler DA, Marth GT. Subcloneseeker: a computational framework for reconstructing tumor clone structure for cancer variant interpretation and prioritization. Genome Biol. 2014;15:443.
24. 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.
25. Jiao W, Vembu S, Deshwar AG, Stein L, Morris Q. Inferring clonal evolution of tumors from single nucleotide somatic mutations. BMC Bioinform. 2014;15:35.
26. Malikic S, McPherson AW, Donmez N, Sahinalp CS. Clonality inference in multiple tumor samples using phylogeny. Bioinformatics. 2015;31: 1349–56.
27. Popic V, Salari R, Hajirasouliha I, Kashef-Haghighi D, West RB, Batzoglou S. Fast and scalable inference of multi-sample cancer lineages. Genome Biol. 2015;16:91.
28. Schuh A, Becq J, Humphray S, Alexa A, Burns A, Clifford R, Feller SM, Grocock R, Henderson S, Khrebtukova I, et al. Monitoring chronic lymphocytic leukemia progression by whole genome sequencing reveals heterogeneous clonal evolution patterns. Blood. 2012;120:4191–6.
29. Van Loo P, Voet T. Single cell analysis of cancer genomes. Curr Opinion Genet Dev. 2014;24:82–91.
30. Hou Y, Song L, Zhu P, Zhang B, Tao Y, Xu X, Li F, Wu K, Liang J, Shao D, et al. Single-cell exome sequencing and monoclonal evolution of a JAK2-negative myeloproliferative neoplasm. Cell. 2012;148:873–85.
31. Chen D, Eulenstein O, Fernandez-Baca D, Sanderson M. Minimum-flip supertrees: complexity and algorithms. IEEE/ACM Trans Comput Biol Bioinform. 2006;3:165–73.
32. Gusfield D, Frid Y, Brown D. Integer programming formulations and computations solving phylogenetic and population genetic problems with missing or genotypic data. In: Computing and combinatorics. Berlin: Springer; 2007. p. 51–64.
33. Yuan K, Sakoparnig T, Markowetz F, Beerenwinkel N. BitPhylogeny: a probabilistic framework for reconstructing intra-tumor phylogenies. Genome Biol. 2015;16:36.
34. Kim KI, Simon R. Using single cell sequencing data to model the evolutionary history of a tumor. BMC Bioinform. 2014;15:27.
35. Xu X, Hou Y, Yin X, Bao L, Tang A, Song L, Li F, Tsang S, Wu K, Wu H, et al. Single-cell exome sequencing reveals single-nucleotide mutation characteristics of a kidney tumor. Cell. 2012;148:886–95.
36. Wang Y, Waters J, Leung ML, Unruh A, Roh W, Shi X, Chen K, Scheet P, Vattathil S, Liang H, et al. Clonal evolution in breast cancer revealed by single nucleus genome sequencing. Nature. 2014;512:155–60.
37. Kuipers J, Moffa G. Uniform random generation of large acyclic digraphs. Stat Comput. 2015;25:227–42.
38. Friedman N, Koller D. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. Mach Learn. 2003;50:95–125.
39. Kuipers J, Moffa G. Partition MCMC for inference on acyclic digraphs. J Am Stat Assoc. 2016. doi:10.1080/01621459.2015.1133426.
40. Markowetz F, Kostka D, Troyanskaya OG, Spang R. Nested effects models for high-dimensional phenotyping screens. Bioinformatics. 2007;23: 305–12.
41. Madigan D, York J. Bayesian graphical models for discrete data. Int Stat Rev. 1995;63:215–32.