Phertilizer: Growing a Clonal Tree from Ultra-low Coverage Single-cell DNA Sequencing of Tumors

Leah L. Weber[1,*], Chuanyi Zhang[2,*], Idoia Ochoa[2,3,†], and Mohammed El-Kebir[1,4,†]

[1]Department of Computer Science, University of Illinois at Urbana-Champaign, IL 61801, USA
[2]Department of Electrical & Computer Engineering, University of Illinois at Urbana-Champaign, IL 61801, USA
[3]Department of Electrical and Electronics Engineering, University of Navarra, Donostia, Spain
[4]Cancer Center at Illinois, University of Illinois Urbana-Champaign, Urbana, IL 61801, USA
[*]Joint first authorship
[†]Correspondence: {idoia,melkebir}@illinois.edu

Abstract

Emerging ultra-low coverage single-cell DNA sequencing (scDNA-seq) technologies have enabled high resolution evolutionary studies of copy number aberrations (CNAs) within tumors. While these sequencing technologies are well suited for identifying CNAs due to the uniformity of sequencing coverage, the sparsity of coverage poses challenges for the study of single-nucleotide variants (SNVs). In order to maximize the utility of increasingly available ultra-low coverage scDNA-seq data and obtain a comprehensive understanding of tumor evolution, it is important to analyze the evolution of SNVs alongside CNAs from the same set of tumor cells.

We present PHERTILIZER, a method to infer a clonal tree from ultra-low coverage scDNA-seq data of a tumor. Based on a probabilistic model, our method recursively partitions the data by identifying key evolutionary events in the history of the tumor. We demonstrate the performance of PHERTILIZER on simulated data as well as on two real datasets, finding that PHERTILIZER effectively utilizes the copy-number signal inherent in the data to more accurately uncover clonal structure and genotypes compared to previous methods.

Availability: `https://github.com/elkebir-group/phertilizer`

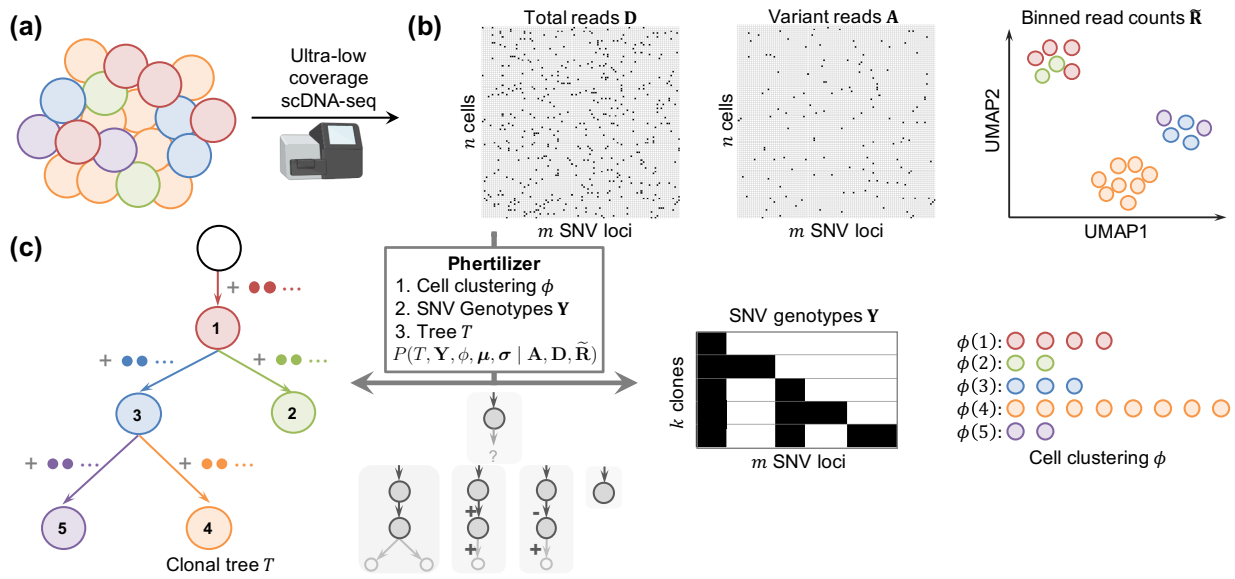


Figure 1: **PHERTILIZER infers a clonal tree T, SNV genotypes $\mathbf{Y}$ and a cell clustering ϕ given ultra-low coverage single-cell sequencing data**. (a) A tumor is composed of groups of cells, or clones with distinct genotypes. (b) Ultra-low coverage scDNA-seq produces total read counts $\mathbf{D} \in \mathbb{N}^{n \times m}$ and variant read counts $\mathbf{A} \in \mathbb{N}^{n \times m}$ for n cells and m SNV loci, and low dimension embedding of binned read counts $\tilde{\mathbf{R}} \in \mathbb{R}^{n \times \ell}$. (c) PHERTILIZER infers a clonal tree T, SNV genotypes $\mathbf{Y}$ and cell clustering ϕ with maximum posterior probability.

1 Introduction

Cancer results from an evolutionary process that yields a heterogeneous tumor composed of multiple subpopulations of cells, or *clones*, with distinct sets of somatic mutations [1]. These mutations include single-nucleotide variants (SNVs) that alter a single base and copy-number aberrations (CNAs) that amplify or delete large genomic regions. Over the last decade, new developments in single-cell DNA sequencing (scDNA-seq) methods have helped uncover a wealth of insights regarding intra-tumor heterogeneity and cancer evolution [2–5]. Importantly, in order to fully characterize the evolution of a tumor, both CNAs and SNVs should be studied and characterized by a single tumor phylogeny that depicts their coevolution. This is a challenging task due to limitations in current scDNA-seq technologies and corresponding tumor phylogeny algorithms.

One important tradeoff of all scDNA-seq technologies is the depth versus uniformity of sequencing coverage. Some technologies like direct library preparation (DLP+) [6] and acoustic cell tagmentation (ACT) [7], which we refer to as ultra-low coverage scDNA-seq, provide extremely low coverage at each position $(0.01 - 0.1\times)$ but offer close to uniform sampling of the whole genome. As a result, these technologies have paved the way for enriching our understanding of the role CNAs play in cancer progression and tumor evolution [6–8]. The drawback of these ultra-low coverage sequencing techniques is the sparsity of coverage, which make it difficult to identify and characterize the evolution of SNVs. On the other hand, scDNA-seq technologies such as Mission Bio Tapestry [9] and MDA have higher coverage ($> 1\times$), due to preamplification of the DNA, thus making them well suited for the identification of point mutations and accurate reconstruction of clonal genotypes [10] and tumor phylogenies [11–14] based only on SNVs. However, these higher coverage scDNA-seq methods are not ideal for identifying CNAs as a result of preamplification, which either provides a biased sampling of the genome or intentionally targets only loci that have a known impact in similar cancers.

Current methods for tumor phylogeny and/or clone inference naturally choose to focus on the features (SNV or CNA events) for which the data is ideally suited [10–19]. SBMCLONE [20] took the first step of using ultra-low coverage sequencing data to infer SNV clones via stochastic block modeling. While SBMCLONE performed well on simulated data, especially with higher coverage ($> 0.2\times$), it was unable to identify clear structure in a 10X genomics breast cancer dataset without *ad hoc* use of additional copy number clone information. Moreover, it is non-trivial convert the inferred parameters of SBMCLONE's stochastic block model to clonal genotypes, which may impact

downstream analyses. Similarly, [21] developed a cell clustering method that focuses only on a set of informative loci. While both of these clustering methods capitalize on the ever increasing throughput of ultra-low coverage scDNA-seq methods, neither method constrains the output by a tree and CNA features are used only in an *ad hoc* manner or for orthogonal validation. As a result, both methods imply that CNA and SNV data features should be segregated and analyzed in separate bioinformatics pipelines. The other emerging trend from the analysis of ultra-low coverage scDNA-seq is pseudobulk analysis [6]. This approach begins by identifying copy number clones using existing methods, followed by pooling cells that belong to the same CNA clones into pseudobulk samples, which are then independently analyzed to identify SNVs. Finally, phylogeny inference is performed with the copy number clones as the leaves of the tree. By forcing the leaves to correspond to the copy number clones, this method does not allow for further refinement of these clones based on SNV evolution.

Here we introduce PHERTILIZER, the first method to grow an SNV clonal tree from ultra-low single-cell DNA sequencing of tumors. The key insight of PHERTILIZER is that to overcome SNV coverage sparsity, we rely on both the high throughput of ultra-low coverage scDNA-seq and the strong copy number signal inherent in the data to guide the building of the clonal tree. By analogy to the planting and growing of trees, our method PHERTILIZER takes as input the variant and total read counts at a given set of loci and the total number of reads in binned regions of the genome and seeks to grow a clonal tree by recursively inferring a set of elementary clonal trees that are used as building blocks to construct the final output clonal tree with maximum posterior probability. As opposed to current methods, our simulations demonstrate that PHERTILIZER accurately infers phylogenies and cell clusters when the number of cells matches current practice. On real data, we find that PHERTILIZER effectively utilizes the copy-number signal inherent in the data to uncover clonal structure, yielding high-fidelity SNV genotypes.

2 Problem Statement

Our goal is to infer a phylogeny from ultra-low coverage sequencing data from n cells from which m single-nucleotide variants (SNVs) have been identified. More precisely, we are given variant reads $\mathbf{A} = [a_{iq}]$ and total reads $\mathbf{D} = [d_{iq}]$, where a_{iq} and d_{iq} are the variant and total read counts for SNV locus $q \in [m]$ in cell $i \in [n]$, respectively (Fig. 1b). While the number of cells is large with the latest generation of ultra-low coverage single-cell DNA sequencing technology ($n \approx 1000$ cells), the *coverage*, or the average number of reads that span a single locus is uniform but low ($0.01\times$ to $0.1\times$). For example with a coverage of $0.01\times$, we would on average observe $a_{iq} = d_{iq} = 0$ reads for 99 out of every 100 loci q for each cell i. This sparsity renders phylogeny inference using SNVs extremely challenging. We propose to overcome this challenge using the following three key ideas.

First, similarly to current methods, we leverage the clonal structure present in tumors, i.e., cells typically cluster into a small number of clones. Thus, we seek to group the n observed cells into k clones such that $k \ll n$ via a cell clustering and corresponding clonal genotypes are defined as follows.

Definition 1. Function $\phi : [k] \to 2^{[n]}$ is a *cell clustering* provided its image encodes a partition of cells $[n]$ into k (disjoint and non-empty) parts.

Definition 2. Matrix $\mathbf{Y} \in \{0,1\}^{k\times m}$ encodes *clonal genotypes* where $y_{jq} = 1$ ($y_{jq} = 0$) indicates that SNV q is present (absent) in clone j.

Second, because ultra-low coverage scDNA-seq is sequenced uniformly, we may leverage the copy number signal inherent in the data to improve cell clustering performance and guide tree inference. More specifically, we expect that all cells in a clone have identical copy number profiles. As we do not observe copy number directly, we will use observed reads counts $\mathbf{R} \in \mathbb{N}^{n\times b}$, where b is the number of genomic bins, as a proxy for copy number. More precisely, we will derive distances that reflect copy-number similarity between pairs of cells on a low-dimensional embedding of binned read counts $\tilde{\mathbf{R}} \in \mathbb{R}^{n\times \ell}$ of $\mathbf{R}$ (i.e., $\ell \ll b$) — see Section 3.1.

Third, similarly to methods such as SCITE [11], SciCloneFit [14] and SPhyR [13], which operate on medium-to-high coverage scDNA-seq data, we consider that the observed cells are generated as the result of a tree-like evolutionary process that constrains the order of SNV clusters. In particular, we use the infinite sites model [22] defined as follows.

Definition 3. A tree T with nodes $V(T) = \{v_1, \ldots, v_k\}$ rooted at node v_1 is a *clonal tree* with clonal genotypes $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_k]^\top$ provided (i) each node v_j is labeled by clonal genotype $\mathbf{y}_j$ and (ii) each SNV q is gained exactly once and subsequently never lost, i.e., there is either exactly one directed edge $(v_{j'}, v_j)$ such that $y_{j'q} = 0$ and $y_{jq} = 1$ or $y_{1q} = 1$, and no edge $(v_{j'}, v_j)$ exists such that $y_{jq} = 1$ and $y_{j'q} = 0$.

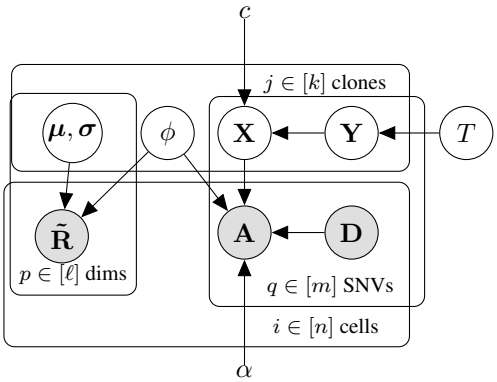


Figure 2: **Plate diagram.** The observed data consists of variant and total read counts $\mathbf{A}, \mathbf{D}$ and binned read counts $\tilde{\mathbf{R}}$, generated from an unobserved cell clustering ϕ and clonal tree T with SNV genotypes $\mathbf{Y}$. Given hyperparameters $\theta = (c, \alpha)$ and observed data $(\mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}})$, we seek $(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma})$ with maximum posterior probability $P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta)$, marginalizing over latent VAFs $\mathbf{X}$.

To relate the data to our latent variables of interest $(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma})$, we introduce a generative model in Section 3.1 that takes sequencing errors into account, as well as the effect of copy number on SNV variant read counts $\mathbf{A}$ and the binned read count embedding $\tilde{\mathbf{R}}$. This enables us to define the posterior probability $P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta)$ that we use in the following problem.

Problem 1 (CLONAL TREE INFERENCE (CTI))**.** Given variant reads $\mathbf{A} \in \mathbb{N}^{n \times m}$, total reads $\mathbf{D} \in \mathbb{N}^{n \times m}$ and binned read count embedding $\tilde{\mathbf{R}} \in \mathbb{R}^{n \times \ell}$, find a clonal tree T with genotypes $\mathbf{Y}$ and cell clustering ϕ with maximum posterior probability $P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta)$.

3 Methods

3.1 Generative model

Our generative model captures the evolution and the subsequent process of ultra-low coverage sequencing of n cells with m SNVs yielding our data $\mathbf{A}, \mathbf{D} \in \mathbb{N}^{n \times m}$ and $\tilde{\mathbf{R}} \in \mathbb{R}^{n \times \ell}$. Given a number $k \in \mathbb{N}$ of clones, we generate a rooted tree T with k nodes with a uniform prior $P(T \mid k)$. We generate clonal genotypes $\mathbf{Y}$ for tree T under the infinite sites model with $P(\mathbf{Y} \mid T)$ uniformly distributed. We model the cell clustering $\phi : [k] \to 2^{[n]}$ with $P(\phi)$ distributed uniformly over all possible partitions of n cells into k clones/parts. The prior probability $P(\mathbf{D})$ of total read counts $\mathbf{D}$ is also uniform. The generative model has three remaining components: the (i) latent variant allele frequency model, (ii) the variant read count model and (iii) the binned read count embedding model (Fig. 2).

Latent variant allele frequency (VAF) model. Let $\bar{d}_{jq}$ be the number of chromosomal copies of the SNV locus q in clone j, and let $\bar{a}_{jq} \leq \bar{d}_{jq}$ be the number of chromosomal copies that harbor the SNV. We call the ratio $\bar{a}_{jq}/\bar{d}_{jq}$ the *latent variant allele frequency* (VAF), which we model as the random variable x_{jq}. Latent VAFs $\mathbf{X} \in \mathbb{Q}^{k \times m}$ depend on the clonal genotypes $\mathbf{Y}$. If clonal genotype $y_{jq} = 0$, then $P(x_{jq} = 0 \mid y_{jq} = 0) = 1$. Otherwise, given a maximum copy number $c \in \mathbb{N}$ $(c > 0)$, we define the multiset $S = \{s/t \mid 1 \leq s \leq t \leq c\}$ of latent VAFs. Then, for each SNV locus q with $y_{jq} = 1$, we have that $P(x_{iq} \mid y_{jq} = 1)$ equals the proportion of occurrences of x_{iq} in S. For example, if we have maximum copy number $c = 2$, the SNV is present (i.e., $y_{jq} = 1$), we have that $S = \{1/1, 1/2, 2/2\}$ and that $P(x_{iq} = 1 \mid y_{jq} = 1) = 2/3$. Appendix A.2.1 contains the precise definition of $P(\mathbf{X} \mid \mathbf{Y}, \theta)$.

Variant read count model. Given latent VAFs $\mathbf{X}$, total read counts $\mathbf{D}$ and cell clustering ϕ, we model alternate read counts $\mathbf{A}$ as a binomial distribution accounting for sequencing errors. While previous models for medium-to-high coverage scDNA-seq data specifically account for errors introduced during PCR amplification [23, 24], the lack of preamplification in the latest generation of ultra-low scDNA-seq technology restricts errors to base calling errors that occur during sequencing [6, 7]. Let α be the probability of misreading a single nucleotide during sequencing, then we model the variant read count a_{iq} of SNV locus q in cell i as

$$a_{iq} \mid \phi, \mathbf{x}_{\cdot q}, d_{iq} \sim \text{Binom}\left(d_{iq},\, x_{jq}(1-\alpha) + \frac{(1-x_{jq})\alpha}{3}\right),$$

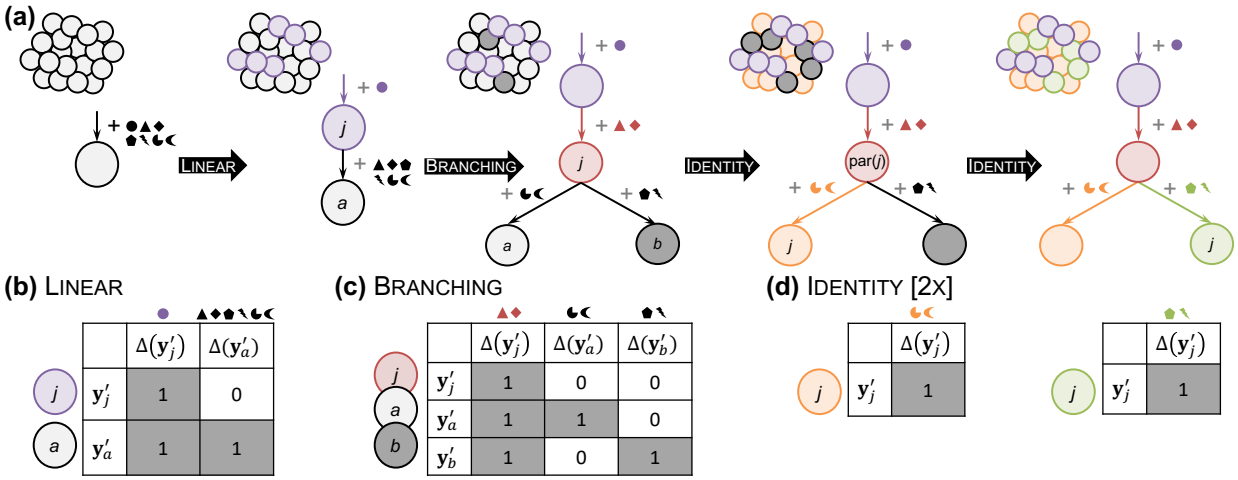


Figure 3: **PHERTILIZER solves the CTI problem by enumerating clonal trees using three elementary operations in a recursive fashion.** (a) Each operation yields a new clonal tree T' by extending a leaf v_j of the previous clonal tree T and reassigning its SNVs $\Delta(\mathbf{y}_j)$ and cells $\phi(j)$. The resulting clonal genotypes $\mathbf{Y}'$ and cell clustering ϕ' are constrained as depicted: (b) `Linear`, (c) `Branching`, and (d) `Identity`.

where j is the unique clone corresponding to cell i, i.e., $i \in \phi(j)$. Appendix A.2.2 provides the derivation of $P(\mathbf{A} \mid \mathbf{D}, \mathbf{X}, \phi, \theta)$.

Binned read count embedding model. The projection of binned reads counts $\mathbf{R}$ into a low dimension embedding $\tilde{\mathbf{R}}$ has two advantages. First, it allows us to identify sets of cells with similar copy number profiles, without the overhead of copy number calling. Second, distance functions behave counter-intuitively in high dimensions [25, 26] and therefore dimensionality reduction provides us with a useful proxy for the distance between the copy number profile of pairs of cells. We model the generation of the binned read counts embedding $\tilde{\mathbf{R}}$ using a Gaussian mixture model where $\tilde{r}_{ip} \mid \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \sim \mathcal{N}(\mu_p, \sigma_p^2)$. Latent variables $\boldsymbol{\mu} \in \mathbb{R}^{k \times \ell}$ and $\boldsymbol{\sigma} \in \mathbb{R}_{\geq 0}^{k \times \ell}$ are independent for each clone $j \in [k]$ and dimension $p \in [\ell]$.

Posterior probability. Given hyperparameters $\theta = (c, \alpha)$ for the maximum copy number c and sequencing error rate α and observed data $(\mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}})$, we seek to identify latent variables $(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma})$ with maximum posterior probability $P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta)$. Using Bayes' theorem and further simplification, we show in Appendix A.2.4 that this is equivalent to

$$\underset{T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}}{\arg\max} \{\log P(\tilde{\mathbf{R}} \mid \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}) + \log \sum_{\mathbf{X}} P(\mathbf{A} \mid \mathbf{D}, \mathbf{X}, \phi, \theta) P(\mathbf{X} \mid \mathbf{Y}, \theta)\}. \tag{1}$$

3.2 PHERTILIZER

At the root of PHERTILIZER are three elementary tree operations that break down the CTI problem into smaller subproblems, which we seek to solve to optimality. Specifically, each operation takes as input a clonal tree T with genotypes $\mathbf{Y}$ and cell clustering ϕ and yields a new clonal tree T' with updated genotypes $\mathbf{Y}'$ and cell clustering ϕ' by transforming exactly one leaf v_j of T (Fig. 3). To define these operations, we introduce some notations. Given node v_j not equal to the root v_1, the function $\mathrm{par}(j)$ returns its parent node $v_{\mathrm{par}(j)}$ in clonal tree T. Recall that $\mathbf{y}_j \in \{0, 1\}^m$ is the clonal genotype of clone j corresponding to node v_j in T. With slight overloading of notation, $\mathbf{y}_j$ also indicates the set of SNVs present in clone j. We denote the set of SNVs introduced on the incoming edge $(v_{\mathrm{par}(j)}, v_j)$ of node v_j by $\Delta(\mathbf{y}_j) = \mathbf{y}_j \setminus \mathbf{y}_{\mathrm{par}(j)}$.

A `Linear` operation transforms node v_j of clonal tree T with cells $\phi(j)$ and mutations $\Delta(\mathbf{y}_j)$ into a clonal tree T' with a new node v_a and directed edge (v_j, v_a), partitioning original cells $\phi(j)$ and mutations $\Delta(\mathbf{y}_j)$ among nodes v_j and v_a such that the resulting clonal genotypes $\mathbf{y}'_j, \mathbf{y}'_a$ adhere to the infinite sites model, i.e. $\mathbf{y}'_j \subseteq \mathbf{y}'_a$ and

$\mathbf{y}'_j \cup \mathbf{y}'_a = \Delta(\mathbf{y}_j)$ (Fig. 3b). Similarly, a `Branching` operation results in a clonal tree T' with two new nodes v_a, v_b that are children of v_j such that the original cells $\phi(j)$ are partitioned between child nodes v_a, v_b, the mutations $\Delta(\mathbf{y}_j)$ are partitioned between all three nodes v_j, v_a, v_b, and the genotypes $\mathbf{y}'_a, \mathbf{y}'_b, \mathbf{y}'_c$ also adhere to the infinite sites model, i.e. $\mathbf{y}'_j \subseteq \mathbf{y}'_a$, $\mathbf{y}'_j \subseteq \mathbf{y}'_b$, $\Delta(\mathbf{y}'_a) \cap \Delta(\mathbf{y}'_b) = \emptyset$ and $\mathbf{y}'_j \cup \mathbf{y}'_a \cup \mathbf{y}'_b = \Delta(\mathbf{y}_j)$. We note that this operation may not assign cells to v_j, thus modeling an extinct clone with SNVs that are common to its children but absent from its parent (Fig. 3c). Lastly, an `Identity` operation on v_j does not alter the tree and prevents the application of other elementary operations on that node (Fig. 3d).

While each elementary tree algorithm varies out of necessity for the specific constrained CTI problem being solved, the two key ideas overlap. First, we partition cells ϕ_j and SNVs $\mathbf{y}_j$ in an iterative fashion. That is, we fix one partition (*e.g.* SNVs) and solve for the other partition (*e.g.* cells), and alternate until convergence or until we reach a specified number of iterations. Drawing parallels between image segmentation, where utilizing pixel location features results in better clustering [27], we incorporate the binned read count embedding $\tilde{\mathbf{R}}$ and variant read counts $\mathbf{A}$ into a single feature as opposed to only using sparse SNV features when finding a cell partition. We implement this via the well known normalized cut algorithm [27]. Since our method is iterative, improved cell partitions enables us to subsequently obtain better quality SNV partitions. Given a fixed cell partition, we use our probabilistic model to assign each SNV being partitioned to the edge with maximum likelihood. Second, to prevent overfitting, we introduce additional regularization via a number of parameters that allows us to enforce the `Identity` operation on leaf nodes without sufficient information content and to avoid extensions that poorly fit the data.

Staring from a tree T with a single node v_1, we use these elementary operations to enumerate a set of valid candidate trees and select the solution $(T^*, \mathbf{Y}^*, \phi^*)$ with maximum posterior probability $P(T^*, \mathbf{Y}^*, \phi^* \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta)$. Details are provided in Appendix A.3. PHERTILIZER is implemented in Python 3, is open source (BSD-3-Clause license), and is available at `https://github.com/elkebir-group/phertilizer`.

4 Results

4.1 Simulation study

Simulation overview. To assess the performance of PHERTILIZER, we performed a simulation study with known ground truth clonal trees, evaluating the folowing four questions: (i) How accurate are the inferred clonal trees? (ii) How well is each method able to identify clusters of cells with similar clonal genotypes? (iii) How accurate are the inferred clonal genotypes? (iv) How sensitive is each method to violations of the infinite sites assumption? We aimed to design our simulation study to match the characteristics of current datasets generated from ultra-low coverage scDNA-seq. To achieve this, we generated simulation instances with varying number $k \in \{5, 9\}$ of nodes, number $n \in \{1000, 2000\}$ of sequenced cells and the number $m \in \{5000, 10000, 15000\}$ of SNVs with a mean sequencing coverage $g \in \{0.01\times, 0.05\times, 0.1\times\}$. We replicated each of these combination 10 times for a total of 360 instances. See Appendix B.1 for details on the generation of simulation instances.

We assessed the quality of an inferred solution $(T, \phi, \mathbf{Y})$ against a ground-truth tree T^*, cell clustering ϕ^* and clonal genotypes $\mathbf{Y}^*$ using ancestral pair recall (APR), incomparable pair recall (IPR), clustered pair recall (CPR) metrics for cells and SNVs [28] as well as SNV genotype similarity. Briefly, for any two SNVs $q \neq q'$ there are three possible placements in the tree: (i) *ancestral*: q is gained on a node that is distinct and ancestral to the node where q' is gained; (ii) *clustered*: q and q' are both gained on the same node; and (iii) *incomparable*: q and q' are gained on distinct nodes that occur on distinct branches of the tree. Similarly, two distinct cells $i \neq i'$ have the same three possible placements. The APR assesses how many ancestral pairs from T^* are recalled in T, whereas CPR and IPR do so for clustered and incomparable pairs, respectively. Thus, if $\mathrm{APR} = \mathrm{CPR} = \mathrm{IPR} = 1$ for both cells and SNVs then the inferred solution $(T, \phi, \mathbf{Y})$ is identical to the ground truth $(T^*, \phi^*, \mathbf{Y}^*)$. We compute a single *accuracy* value composed of the weighted average of APR, IPR and CPR — where the weights are the number of ancestral, incomparable and clustered pairs, respectively. We define the *SNV genotype similarity* as $1/(nm) \sum_{i=1}^{n} \sum_{q=1}^{m} \mathbb{1}(y_{ij} = y^*_{ij})$, given an inferred genotype $\hat{\mathbf{y}}_i \in \{0, 1\}^m$ and a ground truth genotype $\mathbf{y}^*_i$. Thus, a similarity of 1 implies that every sequenced cell was correctly genotyped. We refer to Fig. 4a and Fig. S2 for examples of these metrics.

We benchmarked our method against two clustering/genotyping methods, SBMCLONE [20] and a commonly adopted *ad hoc* practice which we refer to as BASELINE [6, 7]. Specifically, the BASELINE infers SNV genotypes for cells utilizing copy-number information embedded in the binned read counts $\mathbf{R}$ (details in Appendix A.4). Since there exists no standalone method that performs cell clustering, genotyping and tree inference for ultra-low coverage scDNA-seq, we paired SBMCLONE and BASELINE with SCITE [11]. For ease of exposition, we focus our discussion

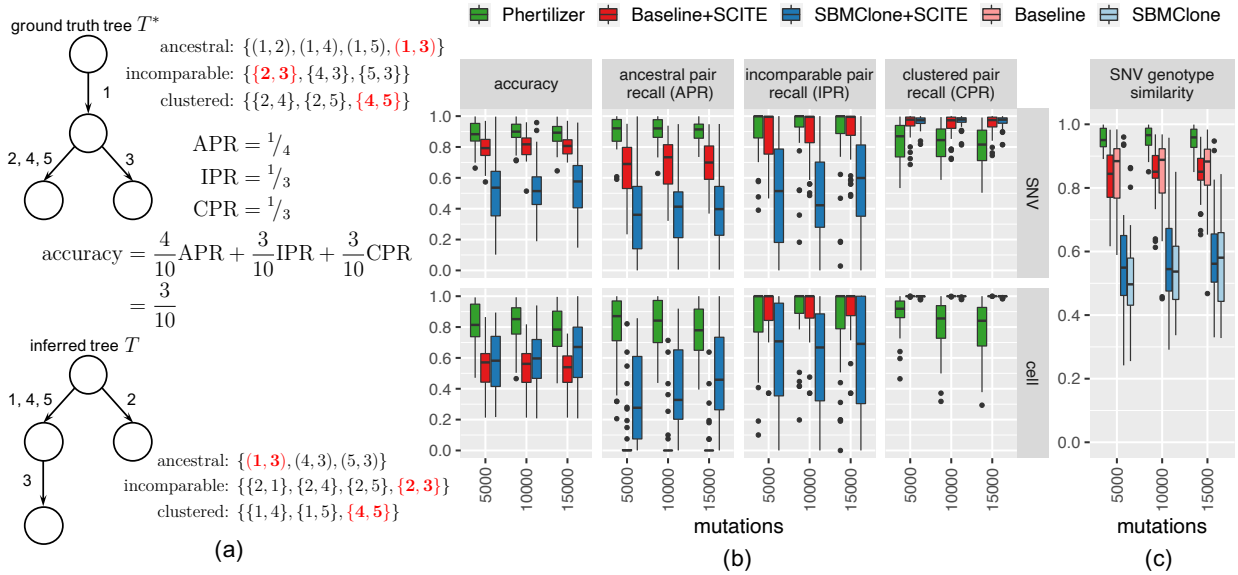


Figure 4: **PHERTILIZER outperforms BASELINE and SBMCLONE on simulated data.** This plot shows aggregated results for $n \in \{1000, 2000\}$ cells, $k \in \{5, 9\}$ clones and coverage $g = 0.05\times$. (a) Example of SNV accuracy, ancestral pair recall (APR), clustered pair recall (CPR), and incomparable pair recall (IPR) metrics. See Fig. S2 for the corresponding example for cell metrics and SNV genotype similarity. (b) PHERTILIZER outperforms BASELINE +SCITE and SBMCLONE +SCITE in APR and IPR for both SNVs and cells. Although competing methods rank higher in CPR, overall PHERTILIZER performs the best considering the accuracy. (c) PHERTILIZER more accurately recovers SNV genotypes than competing methods.

on a coverage g of $0.05\times$ and show aggregated results for $n \in \{1000, 2000\}$ cells and $k \in \{5, 9\}$ clones. We report the median for all performance metrics and include the interquartile range (IQR), *i.e.* the difference between the 75th and 25th percentiles of the data, when appropriate. We note deviations in trends where relevant, and refer to Appendix B for results for coverage $g \in \{0.01\times, 0.1\times\}$.

Results. We first evaluated the accuracy of SNV placement on the inferred tree by assessing APR, CPR, IPR and accuracy for SNVs (Fig. 4b). Overall, PHERTILIZER (median: 0.90) achieved the highest SNV accuracy in terms of SNV placement of all three methods (BASELINE+SCITE median: 0.81, SBMCLONE+SCITE median: 0.54). This result can be explained by further investigating APR, CPR and IPR metrics for SNVs. For SNV APR, PHERTILIZER (median: 0.92) outperformed both BASELINE+SCITE (median: 0.71) and SBMCLONE+SCITE (median 0.38). This implies that our `Linear` operation reliably partitions SNVs accurately and that our `Branching` operations performs well at identifying the SNVs that should be placed at the parent node. Moving on to SNV IPR, PHERTILIZER (median: 1.0, IQR: 0.11) outperformed SBMCLONE+SCITE (median: 0.48, IQR: 0.5) with lower variability than BASELINE+SCITE (median 1.0, IQR: 0.18). Thus, in addition to correctly identifying the SNVs in the parent node, the `Branching` operation also successfully partitions the SNVs in the children nodes. However, for SNV CPR, PHERTILIZER performed worse compared to SBMCLONE+SCITE (median: 0.98) and BASELINE+SCITE (median: 0.97) but still maintained good performance (median 0.84). It is important to note that an SNV CPR of 1 is achievable by clustering all SNVs into a single cluster. Thus, the cost of underfitting the data, or grouping the SNVs into a few very large clusters, will be reflected in decreased APR and IPR. Indeed, we observed this to be the case for both BASELINE+SCITE, which performed relatively poorly on APR, and SBMCLONE, which is the worst performing on both APR and IPR.

Next, we evaluated the accuracy of the cell clustering and placement on the inferred tree. We observed similar trends across accuracy, APR, IPR and CPR cell performance metrics to those identified for their SNV counterparts (Fig. 4b). PHERTILIZER achieved the highest overall cell accuracy (median: 0.82) in comparison to SBMCLONE+SCITE (median: 0.60) and BASELINE+SCITE (median: 0.56). Likewise, PHERTILIZER (median: 0.84)

outperformed all other methods on cell APR (BASELINE+SCITE median: 0.0, SBMCLONE+SCITE median: 0.32). On cell IPR, both PHERTILIZER (median: 1.0) and BASELINE+SCITE (median: 1.0, IQR: 0.13) significantly outperform SBMCLONE+SCITE (median: 0.67, IQR: 0.68) but BASELINE+SCITE (IQR: 0.13) has slightly lower variability than PHERTILIZER (IQR: 0.2). Similarly to SNV CPR, BASELINE+SCITE (median: 1.0) and SBMCLONE+SCITE (median: 1.0) outperformed PHERTILIZER on cell CPR (median: 0.87), but the corresponding decreased performance in cell APR and IPR is indicative of inferring too few cell clusters. In addition to providing more supporting evidence for the validity of our `Linear` and `Branching` operations, these cell placement performance metrics also highlight the advantages of utilizing both copy number and SNV signal for tree inference, which is inherent within PHERTILIZER. In contrast, BASELINE+SCITE prioritizes copy number signal and is unable to further refine a cluster of cells with the same copy number profile but contains SNV evolution. Conversely, SBMCLONE+SCITE ignores copy number signal and struggles to infer the nodes in the clonal tree with sparse SNV signal.

Finally, we assessed the SNV genotype similarity and include SBMCLONE and BASELINE as this is obtained prior to tree inference. Since SNV genotype similarity compares the inferred genotype or each simulated cell against its ground truth genotype, it captures the interplay between of cell clustering and clonal genotyping. Given that PHERTILIZER achieved the highest performance on the clonal tree inference and cell clustering metrics, we would expect PHERTILIZER to have the highest SNV genotype similarity. Indeed, Fig. 4c demonstrates that to be true since PHERTILIZER is the only method to have a median SNV genotype similarity above 0.95. BASELINE is the next highest (median: 0.88), closely followed by BASELINE+SCITE (median: 0.85), while SBMCLONE+SCITE (median: 0.55) and SBMCLONE (median: 0.53) have the worst performance.

When evaluating these metrics at the lowest coverage $g = 0.01\times$, we find PHERTILIZER maintained top performance on both cell placement and SNV genotype similarity but that BASELINE+SCITE was competitive with PHERTILIZER on SNV placement (Fig. S3). For the highest coverage $g = 0.1\times$ regime, PHERTILIZER achieved the highest accuracy on SNV placement and cell placement and had a median SNV genotype similarity of 0.97 while the next closest competitor (BASELINE) had a median similarity of 0.89 (Fig. S5).

We performed an additional set of simulations under a Dollo [29] evolutionary model with $k = 9$, $m = 15000$, coverage $g \in \{0.01\times, 0.05\times, 0.01\times\}$ to assess robustness to violations of infinite sites. We find that PHERTILIZER still outperforms BASELINE+SCITE and SBMCLONE+SCITE while maintaining high scores on all performance metrics (Fig. S6), with the exception of cell APR for the lowest coverage of $g = 0.01\times$ (Fig. S6a). In terms of running time for the case of $n = 1000$ cells, $m = 15000$ SNVs, and a coverage of $g = 0.01\times$, the median running time of PHERTILIZER is 460s, 45.9s for BASELINE+SCITE and 101s for SBMCLONE+SCITE. In summary, we conclude that given the high level of accuracy obtained by PHERTILIZER on these performance metrics, not only are the elementary operations successful in isolation but also that our probabilistic model is useful in guiding the growth of our clonal trees. Additionally, we find that utilizing copy number signal, whenever available, is necessary when working with ultra-low coverage scDNA-seq data but not sufficient for accurate clonal tree reconstruction and/or SNV genotyping.

4.2 High-grade serous ovarian cancer patient

We ran PHERTILIZER on $n = 890$ sequenced cells from three clonally-related cancer cells lines sourced from the same high-grade serous ovarian cancer patient sequenced with DLP+ [6]. We used the variant and total read counts $\mathbf{A}, \mathbf{D}$ for $m = 14{,}068$ SNVs reported in the paper. Similarly, we obtained binned read counts $\mathbf{R}$ for $b = 6{,}207$ bins (bin width of 500 KB) from the paper. The average coverage for these data is approximately $0.1\times$. Utilizing an approach similar to the BASELINE method described above, Laks et al. [6] identified 9 copy number clones (labeled A-I) through dimensionality reduction and density-based clustering, which were reconciled in a single phylogeny with the copy number clones as leaves (Fig. 5a). We used a grid search to select the input parameters to PHERTILIZER, selecting the clonal tree that had maximum posterior probability. Details on parameter choices are provided in Appendix B.

As shown in Fig. 5b, PHERTILIZER inferred a clonal tree with 13 nodes and 8 clones. We find that PHERTILIZER's tree closely aligns with the tree inferred by Laks et al. [6], with both approaches correctly identifying three major clades corresponding to the three distinct cell lines of origin (Fig. 5c). To assess the quality of each inferred clade, we developed a performance metric called *cell mutational burden* (CMB) defined as $\mathrm{CMB}(i, M) = \sum_{q \in M} \mathbf{1}\{a_{iq} > 0\} / \sum_{q \in M} \mathbf{1}\{d_{iq} > 0\}$. In words, $\mathrm{CMB}(i, M)$ is the fraction of mapped SNV loci M with mapped variant reads in cell i. For a specified *clade* j or subtree rooted at node v_j, SNVs M_j are the SNVs gained at node v_j. For a cell i placed within clade j, we expect $\mathrm{CMB}(i, M_j)$ to be high, although the value will depend on copy number. By contrast, for cells placed outside of clade j, we expect $\mathrm{CMB}(i, M_j)$ to be low.

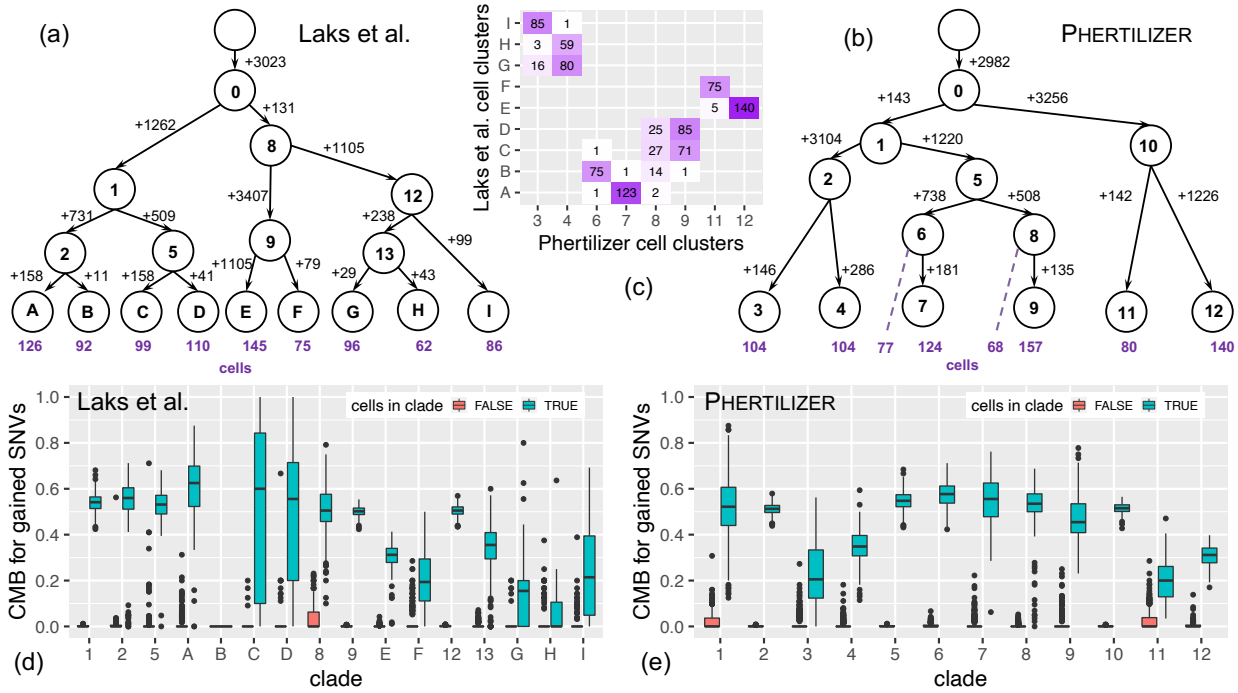


Figure 5: **PHERTILIZER improves upon the SNV phylogeny previously inferred by Laks et al. [6]** (a) The clonal tree inferred by Laks et al. [6] with edges labeled by SNV gains. The number of cells placed in each leaf node is shown below. (b) The clonal tree inferred by PHERTILIZER with edges labeled by SNV gains. The number of cells placed in each leaf node is shown below. (c) Mapping between the Laks et al. cell placement and PHERTILIZER cell placement. (d-e) A comparison of the distribution of cell mutational burden (CMB) for each clade in, respectively, the inferred Laks et al. [6] and PHERTILIZER clonal tree between cells within the clade and cells outside of it.

Fig. 5d,e depicts the comparison of the distributions of CMB for all clades for cells placed within and outside that clade for Laks et al. [6] and PHERTILIZER, respectively. For cells placed outside of a specified clade, the reported tree by Laks et al. [6] as well as the tree inferred by PHERTILIZER have a median CMB of 0 for all clades. This is indicative of high specificity of both methods, *i.e.* SNVs are not assigned to a clade if there are observations of that SNV outside that clade. However, for cells placed within the clades, we observe greater variability for the Laks et al. [6] inferred clades than PHERTILIZER. This variability is most pronounced for the leaf nodes, especially C, D, G, F, H and I, where most have a small number of SNVs gained. We will further analyze clusters G and H, where the 25th percentile of the CMB for cells within the clade equals 0, and clusters C and D that have large variability (IQR of 0.74 and 0.51 respectively).

The location of clusters G and H in the embedding space suggests overfitting during density-based clustering, with an arbitrary split of a larger cohesive cluster containing G and H (Fig. S8a). In comparison, PHERTILIZER uses both copy number and SNV signal, resulting in the Laks et al. [6] G and H cells being clustered together to node 4 in the inferred PHERTILIZER clonal tree. Comparing CMB distributions for cells within clades G and H (Fig. 5d) with PHERTILIZER's inferred clade 4 (Fig. 5e), we observed a higher 25th percentile (0.31) for node 4 than for G (0.00) and H (0.00). This resulted in large separation between the CMB distributions of cells within and outside of clade 4 for PHERTILIZER but not for Laks et al. [6] G and H clades, implying better SNV placement in PHERTILIZER's tree.

The last major difference of note between the two inferred trees is the clustering of the cells in Laks et al. [6] nodes C and D versus PHERTILIZER's nodes 8 and 9. Similarly to nodes G and H, we did not observe a clear separation of C and D in the embedding space (Fig. S8a), making it difficult to define clusters of these cells without more rigorous copy number profiling and clustering. However, as we saw in the simulation study, PHERTILIZER was able to detect further SNV evolution within a set of cells having the same copy number profile. Cells in these clusters would not be split in half in the embedding space but instead should be scattered randomly throughout a single cluster in the embedding space. In addition to observing cells in nodes 8 and 9 randomly scattered in a cluster in the embedding

(a)

Tumor	Cells	SNVS	Coverage	Phertilizer clones	Minussi et al. super clones	Minussi et al. subclones	SBMclone clones
TN1	1100	13,934	0.036×	6	4	17	1
TN2	1024	8,474	0.049×	6	4	15	1
TN3	1101	13,064	0.021×	1	4	9	1
TN4	1307	14,333	0.031×	5	4	22	1
TN5	1238	10,271	0.021×	1	4	7	1
TN6	1205	11,827	0.020×	1	4	15	1
TN7	915	7,873	0.024×	1	3	18	1
TN8	1224	73,359	0.021×	2	4	15	1

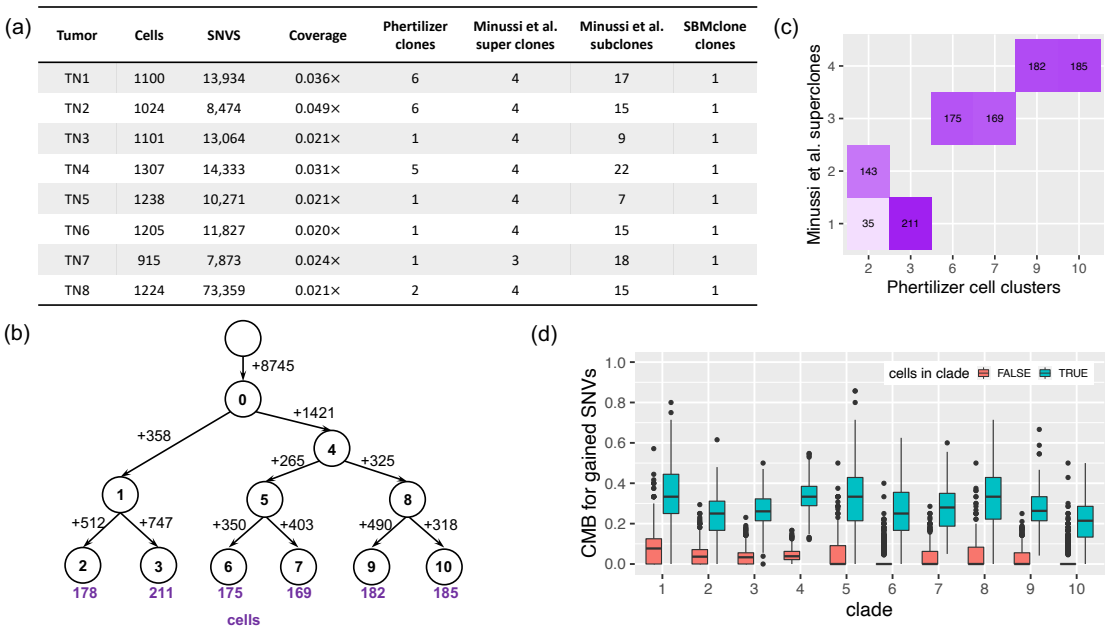


Figure 6: **PHERTILIZER infers clonal tree for breast cancer tumor TN1.** (a) Table depicting the tumor, number of cells and SNVS, the average coverage, and the number of clones inferred by different methods for the eight analyzed breast cancer tumors. (b) The tree inferred by PHERTILIZER with numbers of SNVs labeled beside the edges, and numbers of cells labeled beneath the leaves. (c) A mapping between PHERTILIZER's cell clusters and the Minussi et al. [7] superclones. d) The CMB distribution for the inferred clades for TN1.

space (Fig. S8b), we also observed a clear separation between the CMB distributions for cells placed within and outside of clades 8 and 9 (Fig. 5e). For these clades, we also observed low variability with IQR's of 0.08 and 0.13, respectively, whereas clusters C and D have very high variability in the inferred Laks et al. [6] clonal tree.

Overall, both inferred clonal trees are very similar but since PHERTILIZER simultaneously uses both CNA and SNV information, we obtained a slight improvement in terms of SNV phylogeny inference. Additionally, we note that the small number of SNVs gained at most of the leaf nodes is a direct result of the bottom up approach taken by Laks et al. [6], which performed pseudobulk SNV calling individually on each cell cluster. When SNVs are present in multiple cell clusters but at low prevalence in each cluster, they may not pass filtering of current somatic SNV callers for a cell cluster. This gives the appearance that SNVs are unique to a single clone, when in actuality they are present in multiple clones but not called due sequencing/variant calling artefacts. When sequencing coverage is closer to the $0.01\times$ as opposed to the $0.1\times$ that we have with these data, this challenge of correctly genotyping a cell cluster is amplified further. In contrast, the top down approach of PHERTILIZER is better suited to detect SNVs present in multiple cell clusters at low prevalence.

4.3 Eight triple negative breast tumors

We applied PHERTILIZER to eight triple negative breast tumors sequenced via ACT [7], labeled TN1 to TN8. After dimensionality reduction of normalized and GC-bias corrected binned read counts $\mathbf{R}$, Minussi et al. [7] identified for each tumor two sets of cell clusters with varying granularity, denoted as superclones and subclones. To obtain the input set of SNVs for each patient, we performed SNV calling of a pseudobulk sample of pooled sequenced cells using MuTect2 in tumor-only mode [30]. See Appendix A.5 for further details on data processing. Fig 6a displays the breakdown of each tumor in terms of the number n of cells, the number m of SNVs and average coverage g, and depicts the number of inferred clones by PHERTILIZER, Minussi et al. [7] superclones and subclones, as well as the number of inferred clones by SBMCLONE. We additionally ran BASELINE+SCITE with the cell clusters fixed to the Minussi et al. [7] subclones but all instances except TN3 and TN5 timed out after 10 hours. Furthermore, the CMB distribution for the inferred trees for TN3 and TN5 (Fig. S11) provide no evidence in support of these trees. SBMCLONE only inferred a single clone for all patients, while PHERTILIZER was able to infer a tree with more than

one clone for 4 out of the 8 tumors (TN1: 6, TN2: 6, TN4: 4 and TN8: 2). With the exception of TN8, the other three tumors have the highest average coverage of the eight patients. We will focus our discussion on the clonal trees inferred by PHERTILIZER for tumors TN1 (Fig. 6) and TN2 (Fig. S12).

For tumor TN1, PHERTILIZER inferred a branching tree with 11 nodes and 6 clones (Fig. 6b). We note good concordance between the Minussi et al. [7] superclones and the cell clusters inferred by PHERTILIZER, with the exception of 35 cells that appear as outliers in superclone 1 (Fig. 6c, Fig. S9, Fig. S10). This suggests these cells might fit better in superclone 2 based on SNV signal. In addition, we identified 8745 of the 13934 SNVs as truncal. This large truncal distance and branching structure are in alignment with the truncal distance in the clonal lineage tree inferred by Minussi et al. [7] using bulk whole exome sequencing. We used CMB to assess the performance of SNV and cell placement (Fig. 6d). For clades 5 through 10, we observed that the median CMB for cells outside of the clade is 0. Clades 9 and 10 are particularly interesting because the embedding space depicts the occurrence of SNV evolution within superclone 4. For clades 2 through 4, we note the median CMB for cells outside of the clade is around 0.035 for each of these clades, while clade 1 is the highest at 0.077. Upon further investigation of these 358 SNVs, we found that the median number of mapped reads was 5 when aggregating all cells, making these SNVs especially challenging to place. This drop in performance on the median CMB for cells outside of a clade when compared to the ovarian cancer patient (Section 4.2) is expected due to the drop in sequencing coverage from $0.1\times$ to $0.036\times$. However, we still observed large separation between CMB distributions for the cells within the clade and cells outside the clade for all clades.

For tumor TN2, we inferred a branching a clonal tree with 11 nodes and 6 clones (Fig. S12). Two of PHERTILIZER's cell clusters directly agreed with Minussi et al. [7] superclones. However, PHERTILIZER split the remaining two superclones into four cell clusters $(3, 4, 5, 6)$ using SNV information. We observed low median CMB (0) for cells outside of the clade and a distinct separation between the cells within and outside of the clade distributions, providing evidence for this cell clustering and SNV placement. For tumor TN4, we identified an 8-node tree with 5 cell clusters, with trends similar to tumors TN1 and TN2 in terms of cell clustering concordance and CMB (Fig. S13). Finally, for tumor TN8, PHERTILIZER was only able to infer a 3-node branching tree with 2 cell clusters (Fig. S14). The low sequencing coverage of $0.021\times$ indeed made inference more challenging than the other three tumors, suggesting that PHERTILIZER is most reliable with a coverage of $0.03\times$ or greater. We also note that accurate variant calling remains an open problem for ultra-low scDNA-seq data, making it challenging to even identify all subclonal variants.

5 Discussion

The ability to infer clones using copy-number profiles from ultra-low coverage scDNA-seq data is well established [15]. Here, we showed that the copy-number signal inherent in these data, along with evolutionary constraints, can be used in a novel way to overcome the sparse SNV signal and uncover clonal structure, resulting in high-fidelity SNV and CNA genotypes. We introduced PHERTILIZER, which grows clonal trees in a recursive fashion using two new elementary tree operations. We demonstrated the effectiveness of our approach relative to current methods on both simulated and real data. Importantly, we found that for the current number of cells $(800 - 2000)$ used in practice, PHERTILIZER performs markedly better than current methods, yielding more accurate clonal trees, cell clusters, and SNV genotypes.

There are several limitations and directions for future research. First, the infinite sites model used in this work is often violated due to copy-number deletions. While we demonstrate robustness to such violations, A future direction is to use the Dollo evolutionary model [13, 29]. Second, our model lacks an explicit placement of CNA events on the tree. Tree reconciliation methods, such as PACTION [31], can be applied to integrate SNV clonal tree generated by PHERTILIZER and a CNA tree to obtain a joint tree. Third, beyond SNVs and CNAs, we plan to support structural variants and integrate other omics modalities such as methylation and transcription.

Acknowledgements. We thank the Navin Lab and Darlan Minussi for their assistance with the ACT data. Additionally, we thank the Shah Lab, including Daniel Lai, Robert Reinert, and Andrew McPherson, for their assistance with the DLP+ data. M.E-K. was supported by the National Science Foundation (CCF-2046488) as well as funding from the Cancer Center at Illinois. I.O. was supported by a Gipuzkoa Fellows grant from the Basque Government, a Ramon y Cajal Grant from Spain, and a grant from the Spanish Ministry of Science and Innovation (PID2021-126718OA-I00). This work used resources, services, and support provided via the Greg Gulick Honorary Research Award Opportunity supported by a gift from Amazon Web Services.

References

[1] Nowell, P. C. The clonal evolution of tumor cell populations: Acquired genetic lability permits stepwise selection of variant sublines and underlies tumor progression. *Science* **194**, 23–28 (1976).

[2] Morita, K. *et al.* Clonal evolution of acute myeloid leukemia revealed by high-throughput single-cell genomics. *Nature Communications* **11**, 1–17 (2020).

[3] Baslan, T. *et al.* Novel insights into breast cancer copy number genetic heterogeneity revealed by single-cell genome sequencing. *Elife* **9**, e51480 (2020).

[4] Kim, C. *et al.* Chemoresistance evolution in triple-negative breast cancer delineated by single-cell sequencing. *Cell* **173**, 879–893 (2018).

[5] Miles, L. A. *et al.* Single-cell mutation analysis of clonal evolution in myeloid malignancies. *Nature* **587**, 477–482 (2020).

[6] Laks, E. *et al.* Clonal decomposition and dna replication states defined by scaled single-cell genome sequencing. *Cell* **179**, 1207–1221 (2019).

[7] Minussi, D. C. *et al.* Breast tumours maintain a reservoir of subclonal diversity during expansion. *Nature* **592**, 302–308 (2021).

[8] Zahn, H. *et al.* Scalable whole-genome single-cell library preparation without preamplification. *Nature methods* **14**, 167–173 (2017).

[9] Pellegrino, M. *et al.* High-throughput single-cell DNA sequencing of acute myeloid leukemia tumors with droplet microfluidics. *Genome Research* **28**, 1345–1352 (2018).

[10] Roth, A. *et al.* Clonal genotype and population structure inference from single-cell tumor sequencing. *Nature Methods* **13**, 573–576 (2016).

[11] Jahn, K., Kuipers, J. & Beerenwinkel, N. Tree inference for single-cell data. *Genome biology* **17**, 1–17 (2016).

[12] Malikic, S. *et al.* PhISCS: a combinatorial approach for subperfect tumor phylogeny reconstruction via integrative use of single-cell and bulk sequencing data. *Genome Research* **29**, 1860–1877 (2019).

[13] El-Kebir, M. SPhyR: tumor phylogeny estimation from single-cell sequencing data under loss and error. *Bioinformatics* **34**, i671–i679 (2018).

[14] 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 Research* **29**, 1847–1859 (2019).

[15] Zaccaria, S. & Raphael, B. J. Characterizing allele-and haplotype-specific copy numbers in single cells with chisel. *Nature Biotechnology* **39**, 207–214 (2021).

[16] Markowska, M. *et al.* Conet: Copy number event tree model of evolutionary tumor history for single-cell data. *bioRxiv* (2021).

[17] Liu, Y., Edrisi, M., Ogilvie, H. & Nakhleh, L. Nestedbd: Bayesian inference of phylogenetic trees from single-cell dna copy number profile data under a birth-death model. *bioRxiv* (2022).

[18] Wang, F. *et al.* Medalt: single-cell copy number lineage tracing enabling gene discovery. *Genome Biology* **22**, 1–22 (2021).

[19] Petkovic, M. *et al.* Whole-genome doubling-aware copy number phylogenies for cancer evolution with MEDICC2. *bioRxiv* (2021).

[20] Myers, M. A., Zaccaria, S. & Raphael, B. J. Identifying tumor clones in sparse single-cell mutation data. *Bioinformatics* **36**, i186–i193 (2020).

[21] Rozhoňová, H. *et al.* Secedo: Snv-based subclone detection using ultra-low coverage single-cell dna sequencing. *bioRxiv* (2021).

[22] Kimura, M. The number of heterozygous nucleotide sites maintained in a finite population due to steady flux of mutations. *Genetics* **61**, 893 (1969).

[23] Weber, L. L., Sashittal, P. & El-Kebir, M. doubletD: detecting doublets in single-cell DNA sequencing data. *Bioinformatics* **37**, i214–i221 (2021).

[24] De Bourcy, C. F. *et al.* A quantitative comparison of single-cell whole genome amplification methods. *PloS one* **9**, e105585 (2014).

[25] Aggarwal, C. C., Hinneburg, A. & Keim, D. A. On the surprising behavior of distance metrics in high dimensional space. In *International Conference on Database Theory*, 420–434 (Springer, 2001).

[26] Kriegel, H.-P., Kröger, P. & Zimek, A. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data* **3**, 1–58 (2009).

[27] Shi, J. & Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* **22**, 888–905 (2000).

[28] El-Kebir, M., Oesper, L., Acheson-Field, H. & Raphael, B. J. Reconstruction of clonal trees and tumor composition from multi-sample sequencing data. *Bioinformatics* **31**, i62–i70 (2015).

[29] Dollo, L. Les lois de l'évolution. *Bulletin de la Société belge de géologie, de paléontologie et d'hydrologie* **7**, 164–166 (1893).

[30] Van der Auwera, G. A. & O'Connor, B. D. *Genomics in the cloud: using Docker, GATK, and WDL in Terra* (O'Reilly Media, 2020).

[31] Sashittal, P., Zaccaria, S. & El-Kebir, M. Parsimonious clone tree reconciliation in cancer. In *Leibniz International Proceedings in Informatics, LIPIcs*, vol. 201, 9 (Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2021).

[32] McInnes, L., Healy, J. & Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

[33] Pedregosa, F. *et al.* Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011).

[34] Damle, A., Minden, V. & Ying, L. Simple, direct and efficient multi-way spectral clustering. *Information and Inference: A Journal of the IMA* **8**, 181–203 (2019).

[35] Ankerst, M., Breunig, M. M., Kriegel, H.-P. & Sander, J. Optics: Ordering points to identify the clustering structure. *ACM Sigmod record* **28**, 49–60 (1999).

A Supplementary Methods

A.1 Data processing

PHERTILIZER either takes as input binned read counts $\mathbf{R} \in \mathbb{N}^{n \times b}$ of n cells with read counts grouped into b bins or alternatively a binned read count embedding $\tilde{\mathbf{R}} \in \mathbb{R}^{n \times \ell}$ of read counts of n cells projected into ℓ dimensional space. If provided the former, PHERTILIZER will project the binned read counts $\mathbf{R}$ into $\ell = 2$ dimensions as follows. First, we compute normalized binned read counts $R' = [r'_{ib}] \in \mathbb{R}^{n \times b}$ as

$$r'_{ib} = \frac{r_{ib}}{\sum_{p=1}^{b} r_{ip}}. \tag{2}$$

Next, we use UMAP [32] to project the normalized binned read counts $\mathbf{R}'$ to $\ell = 2$ dimensions with the following parameter settings:

```
n_neighbors=40, spread=1, n_components=2, min_dist=0.
```

A.2 Generative model

We have two hyperparameters $\theta = (c, \alpha)$ where $c \in \mathbb{N} \setminus \{0\}$ is the maximum copy number and $\alpha \in [0, 1]$ is the per-base sequencing error rate.

A.2.1 Latent variant allele frequency model

The latent variant allele frequency (VAF) x_{jq} denotes the fraction of chromosomal copies that harbor the SNV at locus q for clone j. The latent VAF $x_{jq} \in \mathbb{Q}$ is conditioned on the latent clonal genotype $y_{jq} \in \{0, 1\}$. Given a pre-specified maximum copy number $c \in \mathbb{N} \setminus \{0\}$, we define the multiset S of latent VAFs associated with SNV state $y_{jq} = 1$ as

$$S = \{s/t \mid 1 \leq s \leq t \leq c\}. \tag{3}$$

We denote with $|\{x \in S\}|$ the number of occurrences of latent VAF x in the multiset S. Then, we have

$$P(x_{jq} \mid y_{jq}, \theta) = \begin{cases} 1, & \text{if } y_{jq} = 0, x_{jq} = 0, \\ \frac{|\{x_{jq} \in S\}|}{|S|}, & \text{if } y_{jq} = 1, x_{jq} > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Using independence among clones $[k]$ and SNV loci $[m]$ when given clonal genotypes $\mathbf{Y}$, we have that

$$P(\mathbf{X} \mid \mathbf{Y}, \theta) = \prod_{j=1}^{k} \prod_{q=1}^{m} P(x_{jq} \mid y_{jq}, \theta). \tag{5}$$

A.2.2 Variant read count model

As discussed in Main Text, we have

$$P(a_{iq} \mid d_{iq}, \mathbf{x}_{\cdot q}, \phi, \theta) = \text{Binom}\left(a_{iq};\, d_{iq},\, x_{jq}(1-\alpha) + \frac{(1-x_{jq})\alpha}{3}\right), \tag{6}$$

where j is the unique clone corresponding to cell i, i.e. $i \in \phi(j)$. Using an indicator variable, this is equivalent to

$$P(a_{i,q} \mid d_{i,q}, \mathbf{x}_{\cdot q}, \phi, \theta) = \prod_{j=1}^{k} P(a_{iq} \mid d_{iq}, x_{jq}, \phi, \theta)^{\mathbf{1}\{i \in \phi(j)\}} \tag{7}$$

$$= \prod_{j=1}^{k} \left[\text{Binom}\left(a_{iq};\, d_{iq},\, x_{jq}(1-\alpha) + \frac{(1-x_{jq})\alpha}{3}\right)\right]^{\mathbf{1}\{i \in \phi(j)\}}. \tag{8}$$

Per the graphical model, we have independence among cells $[n]$ and SNV loci $[m]$ when conditioning on $\mathbf{D}, \mathbf{X}, \phi$ and θ, yielding

$$P(\mathbf{A} \mid \mathbf{D}, \mathbf{X}, \phi, \theta) = \prod_{j=1}^{k} \prod_{i=1}^{n} \prod_{q=1}^{m} P(a_{iq} \mid d_{iq}, x_{jq}, \phi, \theta)^{\mathbf{1}\{i \in \phi(j)\}} \tag{9}$$

$$= \prod_{j=1}^{k} \prod_{i=1}^{n} \prod_{q=1}^{m} \left[\text{Binom}\left(a_{iq};\, d_{iq},\, x_{jq}(1-\alpha) + \frac{(1-x_{jq})\alpha}{3} \right) \right]^{\mathbf{1}\{i \in \phi(j)\}}. \tag{10}$$

A.2.3 Binned read count embedding model

As discussed in the Main Text, we model the binned read count embedding $\tilde{R} \in \mathbb{R}^{n \times \ell}$ as Gaussian mixture model of k clones. Let $z_i \in [k]$ be a random variable depicting the assignment of cell i to one of the k clones. Given a cell clustering ϕ, we have $P(z_i = j \mid \phi) = \mathbf{1}\{i \in \phi(j)\}$ meaning ϕ specifies a hard clustering of cells into k clones. Given cell clustering ϕ and latent variables $\boldsymbol{\mu}_j = [\mu_{jp}]$ and $\boldsymbol{\sigma_j} = [\sigma_{jp}]$ for each clone $j \in [k]$, the probability of binned read count embedding for cell i and dimension p is thus

$$P(\tilde{r}_{ip}, \mid \phi, \mu_{jp}, \sigma_{jp}) = \sum_{j=1}^{k} P(z_i = j \mid \phi) \cdot \mathcal{N}(\mu_{jp}, \sigma_{jp}^2) \tag{11}$$

$$= \prod_{j=1}^{k} \mathcal{N}(\mu_{jp}, \sigma_{jp}^2)^{\mathbf{1}\{i \in \phi(j)\}}. \tag{12}$$

Thus, using independence among cells and dimensions, we have the following probability

$$P(\tilde{\mathbf{R}} \mid \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \prod_{j=1}^{k} \prod_{i=1}^{n} \prod_{p=1}^{\ell} \mathcal{N}(\mu_{jp}, \sigma_{jp}^2)^{\mathbf{1}\{i \in \phi(j)\}}. \tag{13}$$

A.2.4 Posterior probability

As discussed in Main Text, we seek to solve

$$\underset{T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}}{\arg\max}\, P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta). \tag{14}$$

We begin by applying Bayes' rule, obtaining

$$P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta) = \frac{P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \theta) P(\mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}} \mid T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, \theta)}{P(\mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}} \mid \theta)}. \tag{15}$$

We decompose $P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \theta)$ per the graphical model and obtain

$$P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta) = \frac{P(T \mid \theta) P(\mathbf{Y} \mid T, \theta) P(\phi \mid \theta) P(\boldsymbol{\mu}, \boldsymbol{\sigma} \mid \theta) P(\mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}} \mid T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, \theta)}{P(\mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}} \mid \theta)}. \tag{16}$$

Using uniformity of $P(T \mid \theta)$, $P(\mathbf{Y} \mid T, \theta)$, $P(\phi \mid \theta)$, $P(\boldsymbol{\mu}, \boldsymbol{\sigma} \mid \theta)$ and dropping $P(\mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}} \mid \theta)$ from the denominator, we obtain

$$P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta) \propto P(\mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}} \mid T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, \theta). \tag{17}$$

Using the law of conditional probability, we get

$$P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta) \propto P(\mathbf{A}, \tilde{\mathbf{R}} \mid \mathbf{D}, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, \theta) P(\mathbf{D}). \tag{18}$$

We have that $P(\mathbf{D})$ is uniform, and after simplification, yields

$$P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta) \propto P(\mathbf{A}, \tilde{\mathbf{R}} \mid \mathbf{D}, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, \theta) \tag{19}$$

$$= P(\mathbf{A} \mid \mathbf{D}, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, \theta) P(\tilde{\mathbf{R}} \mid \mathbf{D}, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, \theta) \tag{20}$$

$$= P(\mathbf{A} \mid \mathbf{D}, \mathbf{Y}, \phi, \theta) P(\tilde{\mathbf{R}} \mid \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}). \tag{21}$$

We now focus on $P(\mathbf{A} \mid \mathbf{D}, \mathbf{Y}, \phi, \theta)$, which equals

$$P(\mathbf{A} \mid \mathbf{D}, \mathbf{Y}, \phi, \theta) \tag{22}$$

$$= \prod_{j=1}^{k} \prod_{i=1}^{n} \prod_{q=1}^{m} P(a_{iq} \mid d_{iq}, y_{jq}, \phi, \theta)^{\mathbf{1}\{i \in \phi(j)\}} \tag{23}$$

$$= \prod_{j=1}^{k} \prod_{i=1}^{n} \prod_{q=1}^{m} \left[\sum_{x_{jq} \in S \cup \{0\}} P(a_{iq}, x_{jq} \mid d_{iq}, y_{jq}, \phi, \theta) \right]^{\mathbf{1}\{i \in \phi(j)\}} \tag{24}$$

$$= \prod_{j=1}^{k} \prod_{i=1}^{n} \prod_{q=1}^{m} \left[\sum_{x_{jq} \in S \cup \{0\}} P(a_{iq} \mid d_{iq}, x_{jq}, \phi, \theta) P(x_{jq} \mid y_{jq}, \theta) \right]^{\mathbf{1}\{i \in \phi(j)\}}. \tag{25}$$

Bringing everything together, we get

$$P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta) \tag{26}$$

$$\propto P(\mathbf{A} \mid \mathbf{D}, \mathbf{Y}, \phi, \theta) P(\tilde{\mathbf{R}} \mid \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}) \tag{27}$$

$$= \left(\prod_{j=1}^{k} \prod_{i=1}^{n} \prod_{q=1}^{m} \left[\sum_{x_{jq} \in S \cup \{0\}} P(a_{iq} \mid d_{iq}, x_{jq}, \phi, \theta) P(x_{jq} \mid y_{jq}, \theta) \right]^{\mathbf{1}\{i \in \phi(j)\}} \right) \cdot \left[\prod_{j=1}^{k} \prod_{i=1}^{n} \prod_{p=1}^{\ell} \mathcal{N}(\mu_{jp}, \sigma_{jp}^2)^{\mathbf{1}\{i \in \phi(j)\}} \right]. \tag{28}$$

Taking the logarithm, we obtain

$$P(T, \mathbf{Y}, \phi, \boldsymbol{\mu}, \boldsymbol{\sigma} \mid \mathbf{A}, \mathbf{D}, \tilde{\mathbf{R}}, \theta) \propto \log P(\mathbf{A} \mid \mathbf{D}, \mathbf{Y}, \phi, \theta) P(\tilde{\mathbf{R}} \mid \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}) \tag{29}$$

$$= \log P(\mathbf{A} \mid \mathbf{D}, \mathbf{Y}, \phi, \theta) + \log P(\tilde{\mathbf{R}} \mid \phi, \boldsymbol{\mu}, \boldsymbol{\sigma}) \tag{30}$$

$$= \sum_{j=1}^{k} \sum_{i=1}^{n} \sum_{q=1}^{m} \mathbf{1}\{i \in \phi(j)\} \log \sum_{x_{jq} \in S \cup 0} P(a_{iq} \mid d_{iq}, x_{jq}, \phi, \theta) P(x_{jq} \mid y_{jq}, \theta)$$

$$+ \sum_{j=1}^{k} \sum_{i=1}^{n} \sum_{p=1}^{\ell} \mathbf{1}\{i \in \phi(j)\} \log \mathcal{N}(\mu_{jp}, \sigma_{jp}^2). \tag{31}$$

A.3 PHERTILIZER

At the root of PHERTILIZER is the definition of three elementary tree operations, `Branching`, `Linear` and `Identity`. The input to each operation is a clonal tree T with genotypes $\mathbf{Y}$, cell clustering ϕ and a leaf node v_j. The output is a new clonal tree T' with new genotypes $\mathbf{Y}'$ and cell clustering ϕ'. As the definition of the `Identity` operation is trivial, we focus here on defining the `Branching` and `Linear` operations.

A.3.1 `Branching`

Recall that $\phi(j)$ corresponds to the cells attached to node v_j and $\Delta(\mathbf{y}_j)$ corresponds to the SNVs introduced on node v_j. Our starting point is a clonal tree T with a leaf node v_j with the set $\phi(j)$ of cells and the set $\Delta(\mathbf{y}_j)$ of SNVs. The goal is to obtain a new clonal tree T' by extending v_j with two children v_a and v_b with cells $\phi(a)$ and $\phi(b)$, respectively, and SNVs $\Delta(\mathbf{y}_a)$ and $\Delta(\mathbf{y}_b)$, respectively.

First, we initialize a `Branching` operation by partitioning uniformly at random the set $\Delta(\mathbf{y}_j)$ into three parts $\{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b)\}$. Next, we iteratively search for a cell partition $\{\phi(a), \phi(b)\}$ of $\phi(j)$, followed by an SNV partition $\{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b)\}$ of $\Delta(\mathbf{y}_j)$ that maximizes a normalized partial log likelihood for the new clonal tree T', with clonal genotypes $\mathbf{Y}'$ and cell clustering ϕ'. Note that during a `Branching` operation, no cells remain assigned to node j. Given an SNV partition $\{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b)\}$, we find a cell partition $\{\phi(a), \phi(b)\}$ by using the normalized cut algorithm [27]. The input to the normalized cut algorithm is a weighted undirected graph $G = (V, E)$ where specified edge weights relate the similarity of two vertices. Our input graph has vertices $V = \phi(j)$ and we set the edge weights as follows in order to incorporate both SNV features $\mathbf{f}$ and CNA features represented by the binned read count embedding $\tilde{\mathbf{R}}$,

$$w_{ii'} = e^{\frac{-\|\mathbf{f}_i - \mathbf{f}_{i'}\|_2^2}{s_f}} \cdot \begin{cases} e^{\frac{-\|\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_{i'}\|_2^2}{s_c}}, & \text{if } \|\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}_{i'}}\|_2^2 < r, C = 1, \\ 1, & \text{if } C = 0, \\ 0, & \text{otherwise,} \end{cases} \tag{32}$$

where $\mathbf{f}_i = \begin{bmatrix} f_{ia} & f_{ib} \end{bmatrix}^\top$ and $f_{ia} = \frac{\sum_{q \in \Delta(\mathbf{y}_a)} \mathbf{1}\{a_{iq} > 0\}}{\sum_{q \in \Delta(\mathbf{y}_a)} \mathbf{1}\{d_{iq} > 0\}}$. Intuitively, f_{ia} measures for each cell i the proportion of introduced SNV loci $\Delta(\mathbf{y}_a)$ with mapped reads for which we observe at least one variant read. SNV feature f_{ib} is similarly defined for SNVs $\Delta(\mathbf{y}_b)$. In case the denominator $\sum_{q \in \Delta(\mathbf{y}_a)} \mathbf{1}\{d_{iq} > 0\}$ is 0 for a cell i, we set $f_{ia} := f_{i'a}$ where i' is the closest cell to i in $\tilde{\mathbf{R}}$ with nonzero denominator. Parameter $C \in \{0, 1\}$ indicates if CNA features should be incorporated in the calculation of edge weights. Given a user-specified parameter q, we compute r as the qth percentile of $\|\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_{i'}\|_2^2$ over all $i, i' \in [n]$. We find the default value of $q = 1$, *i.e.* the maximum, works well in practice. Following Shi and Malik [27], parameters s_f and s_c are set to 20% of the total range of the L_2 norm for SNV and CNA features, respectively. We use the `scikit-learn` [33] spectral clustering implementation as it solves the normalized cut problem when the number of clusters is 2. We choose the `cluster-qr` [34] strategy to partition cells in the spectral embedding space.

Then, given the updated cell partition $\{\phi(a), \phi(b)\}$, we optimize the SNV partition $\{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b)\}$. For each SNV $q \in \Delta(\mathbf{y}_{j'})$, we compute the log-likelihood of SNV q belonging to each of $\Delta(\mathbf{y}_a)$,$\Delta(\mathbf{y}_b)$ and $\Delta(\mathbf{y}_{j'})$ as follows.

$$L(q \in \Delta(\mathbf{y}_a)) = \sum_{i \in \phi(a)} \log P(a_{iq} \mid d_{iq}, y_{aq} = 1, \theta) + \sum_{i \in \phi(b)} \log P(a_{iq} \mid d_{iq}, y_{bq} = 0, \theta), \tag{33}$$

$$L(q \in \Delta(\mathbf{y}_b))) = \sum_{i \in \phi(b)} \log P(a_{iq} \mid y_{bq} = 1) + \sum_{i \in \phi(a)} \log P(a_{iq} \mid d_{iq}, y_{aq} = 0, \theta), \tag{34}$$

$$L(q \in \Delta(\mathbf{y}_{j'})) = \sum_{i \in \phi(j)} \log P(a_{iq} \mid d_{iq}, y_{j'q} = 1, \theta). \tag{35}$$

We assign q to the set $J \in \{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b)\}$ where $L(q \in J)$ is maximum. We iterate until either the cell clusters are not altered from the previous iteration or a user-specified maximum number of iterations is reached.

Due to the random initialization of the SNV partition, we repeat this process for a number of restarts, alternating between $C = 0$ and $C = 1$ on each restart. After a restart terminates, we compute a normalized partial log likelihood based only clonal genotypes that will not change even if the leaf nodes of the output tree T' are later extended. Let $\eta(K) = \{i, j \mid i \in \phi(j), j \in K\}$, where K is a set of clones. The partial log likelihood term is defined as

$$\begin{aligned} L(\phi(a), \phi(b), \Delta(\mathbf{y}_{j'})\Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b)) = & \sum_{i,j \in \eta(\{a,b\})} \sum_{q \in \Delta(\mathbf{y}_{j'})} \log P(a_{iq} \mid d_{iq}, y_{jq} = 1, \theta) \\ & + \sum_{i \in \phi(a)} \sum_{q \in \Delta(\mathbf{y}_b)} \log P(a_{iq} \mid d_{iq}, y_{bq} = 0, \theta) \\ & + \sum_{i \in \phi(b)} \sum_{q \in \Delta(\mathbf{y}_a)} \log P(a_{iq} \mid d_{iq}, y_{aq} = 0, \theta). \end{aligned}$$

While the total number of observations under consideration is

$$O(\phi(a), \phi(b), \Delta(\mathbf{y}_{j'})\Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b)) = \sum_{i \in \phi(j)} \sum_{q \in \Delta(\mathbf{y}_{j'})} \mathbf{1}\{d_{iq} > 0\} \tag{36}$$

$$+ \sum_{i \in \phi(a)} \sum_{q \in \Delta(\mathbf{y}_b)} \mathbf{1}\{d_{iq} > 0\} \tag{37}$$

$$+ \sum_{i \in \phi(b)} \sum_{q \in \Delta(\mathbf{y}_a)} \mathbf{1}\{d_{iq} > 0\}, \tag{38}$$

and the normalized partial log likelihood is

$$L'(\phi(a), \phi(b), \Delta(\mathbf{y}_{j'})\Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b)) = \frac{L(\phi(a), \phi(b), \Delta(\mathbf{y}_{j'})\Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b))}{O(\phi(a), \phi(b), \Delta(\mathbf{y}_{j'})\Delta(\mathbf{y}_a), \Delta(\mathbf{y}_b))}. \tag{39}$$

Among all restarts, the `Branching` operations returns the extended tree T', clonal genotypes $\mathbf{Y}'$ and cell clustering ϕ' with maximum normalized partial log likelihood.

A.3.2 `Linear`

The `Linear` elementary operation is very similar to the `Branching` elementary operation described above. Recall that $\phi(j)$ corresponds to the cells attached to node v_j and $\Delta(\mathbf{y}_j)$ corresponds to the SNVs introduced on node v_j. Our starting point is a clonal tree T with a leaf node v_j with the set $\phi(j)$ of cells and the set $\Delta(\mathbf{y}_j)$ of SNVs. The goal is to obtain a new clonal tree T' by extending v_j with one child v_a with cells $\phi(a)$ and SNVs $\Delta(\mathbf{y}_a)$.

First, we initialize a `Linear` operation by partitioning uniformly at random the set $\Delta(\mathbf{y}_j)$ into two parts $\{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a)\}$. Next, we iteratively search for a cell partition $\{\phi(j'), \phi(a)\}$ of $\phi(j)$, followed by an SNV partition $\{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a)\}$ of $\Delta(\mathbf{y}_j)$ that maximizes a normalized partial log likelihood for the new clonal tree T', with clonal genotypes $\mathbf{Y}'$ and cell clustering ϕ'. Note that unlike `Branching`, for the `Linear` operation, some cells remain assigned to node j. Given an SNV partition $\{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a)\}$, we find a cell partition $\{\phi(j'), \phi(a)\}$ by using the normalized cut algorithm [27]. The input to the normalized cut algorithm is a weighted undirected graph $G = (V, E)$ where specified edge weights relate the similarity of two vertices. Our input graph has vertices $V = \phi(j)$ and we set the edge weights as follows in order to incorporate both SNV features $\mathbf{f}$ and CNA features represented by the binned read count embedding $\tilde{\mathbf{R}}$,

$$w_{ii'} = e^{\frac{-\|f_i - f_{i'}\|_2^2}{s_f}} \cdot \begin{cases} e^{\frac{-\|\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_{i'}\|_2^2}{s_c}}, & \text{if } \|\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_{i'}\|_2^2 < r, C = 1, \\ 1, & \text{if } C = 0, \\ 0, & \text{otherwise,} \end{cases} \tag{40}$$

where $f_i = \frac{\sum_{q \in \Delta(\mathbf{y}_a)} \mathbf{1}\{a_{iq} > 0\}}{\sum_{q \in \Delta(\mathbf{y}_a)} \mathbf{1}\{d_{iq} > 0\}}$. Intuitively, f_i measures for each cell i the proportion of introduced SNV loci $\Delta(\mathbf{y}_a)$ with mapped reads for which we observe at least one variant read.

In case the denominator $\sum_{q \in \Delta(\mathbf{y}_a)} \mathbf{1}\{d_{iq} > 0\}$ is 0 for a cell i, we set $f_i := f_{i'}$ where i' is the closest cell to i in $\tilde{\mathbf{R}}$ with nonzero denominator. Parameter $C \in \{0, 1\}$ indicates if CNA features should be incorporated in the calculation of edge weights. Given a user-specified parameter q, we compute r as the qth percentile of $\|\tilde{\mathbf{r}}_i - \tilde{\mathbf{r}}_{i'}\|_2^2$ over all $i, i' \in [n]$. We find the default value of $q = 1$, *i.e.* the maximum, works well in practice. Following Shi and Malik [27], parameters s_f and s_c are set to 20% of the total range of the L_2 norm for SNV and CNA features, respectively. We use the `scikit-learn` [33] spectral clustering implementation as it solves the normalized cut problem when the number of clusters is 2. We choose the `cluster-qr` [34] strategy to partition cells in the spectral embedding space.

Then, given the updated cell partition $\{\phi(j'), \phi(a)\}$, we optimize the SNV partition $\{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a)\}$. For each SNV $q \in \Delta(\mathbf{y}_j)$, we compute the log-likelihood of SNV q belonging to each of $\Delta(\mathbf{y}_{j'})$, and $\Delta(\mathbf{y}_a)$ as follows.

$$L(q \in \Delta(\mathbf{y}_{j'})) = \sum_{i \in \phi(j')} \log P(a_{iq} \mid y_{j'q} = 1) \tag{41}$$

$$L(q \in \Delta(\mathbf{y}_a)) = \sum_{i \in \phi(j')} \log P(a_{iq} \mid y_{aq} = 0) \tag{42}$$

We assign q to the set $J \in \{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a)\}$ where $L(q \in J)$ is maximum. We iterate until either the cell clusters are not altered from the previous iteration or a user-specified maximum number of iterations is reached.

Due to the random initialization of the SNV partition, we repeat this process for a number of restarts, alternating between $C = 0$ and $C = 1$ on each restart. After all restarts, we store the cell and SNV partition with the maximum normalized partial log likelihood based only clonal genotypes that will not change even if the leaf nodes of the output tree T' are later extended. Let $\eta(K) = \{i, j \mid i \in \phi(j), j \in K\}$, where K is a set of clones. The partial log likelihood term is defined as

$$\begin{aligned} L(\phi(j'), \phi(a), \Delta(\mathbf{y}_{j'})\Delta(\mathbf{y}_a))) = & \sum_{i,j \in \eta(\{j',a\})} \sum_{q \in \Delta(\mathbf{y}_{j'})} \log P(a_{iq} \mid d_{iq}, y_{jq} = 1, \theta) \\ & + \sum_{i \in \phi(a)} \sum_{q \in \Delta(\mathbf{y}_a)} \log P(a_{iq} \mid d_{iq}, y_{aq} = 0, \theta) \end{aligned}$$

While the total number of observations under consideration is

$$O(\phi(j'), \phi(a), \Delta(\mathbf{y}_{j'})\Delta(\mathbf{y}_a))) = \sum_{i \in \phi(j') \cup \phi(a)} \sum_{q \in \Delta(\mathbf{y}_{j'})} \mathbf{1}\{d_{iq} > 0\} \tag{43}$$

$$+ \sum_{i \in \phi(j')} \sum_{q \in \Delta(\mathbf{y}_a)} \mathbf{1}\{d_{iq} > 0\} \tag{44}$$

$$\tag{45}$$

and the normalized partial log likelihood is

$$L'(\phi(j'), \phi(a), \Delta(\mathbf{y}_{j'})\Delta(\mathbf{y}_a))) = \frac{L(\phi(j'), \phi(a), \Delta(\mathbf{y}_{j'})\Delta(\mathbf{y}_a)))}{O(\phi(j'), \phi(a), \Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a)}. \tag{46}$$

We then store the cell partition $\{\phi(j'), \phi(a)\}$ and SNV partition $\{\Delta(\mathbf{y}_{j'}), \Delta(\mathbf{y}_a)\}$ with maximum normalized partial log likelihood and and recurse on cells $\phi(j) := \phi(j')$ and SNVs $\Delta(\mathbf{y}_j) = \Delta(\mathbf{y}_{j'}$. The recursion terminates when only an `identity` operation can be performed. See Appendix A.3.3 for details on when this occurs. The recursion step is necessary because the normalized cut algorithm is a recursive partitioning algorithm. Unlike `Branching` where we later recurse and attempt to extend the children nodes,the cells and SNVs placed in node v_j after a `Linear` operation remain fixed. Therefore, we need to ensure that cell partition returned by the normalized cut algorithm is minimal. After the recursion terminates, the `Linear` operation returns the extended tree T', clonal genotypes $\mathbf{Y}'$ and cell clustering ϕ' with maximum normalized partial log likelihood.

A.3.3 `Regularization`

To avoid overfitting the input data, we execute a number of regularization steps both before and after performing an elementary tree operation on a node v_j of clonal tree T.

Power to detect an extension. Recall that a tree operation partitions the cells $\phi(j)$ and SNVs $\Delta(\mathbf{y}_j)$ introduced on a leaf node v_j. Specifically, for a `Linear` operation there are four parts

$$\{(\phi(j), \Delta(\mathbf{y}_j)), (\phi(j), \Delta(\mathbf{y}_a)), (\phi(a), \Delta(\mathbf{y}_j)), (\phi(a), \Delta(\mathbf{y}_j))\} \tag{47}$$

of cells and SNVs of interest. For the `Branching` operation there are six such pairs:

$$\{(\phi(a), \Delta(\mathbf{y}_j)), (\phi(j), \Delta(\mathbf{y}_a)), (\phi(j), \Delta(\mathbf{y}_b)), (\phi(b), \Delta(\mathbf{y}_j)), (\phi(b), \Delta(\mathbf{y}_a)), (\phi(b), \Delta(\mathbf{y}_b))\}. \tag{48}$$

For each part, we wish to assess if there are a sufficient number of observations to support extending node v_j. We allow the user to specify the minimum number $\delta \in \mathbb{N}$ of observations. We define *observations* $O(N, M)$ for a subset $N \subseteq [n]$ of cells and a subset $M \subseteq [m]$ of SNVs as

$$O(N, M) = \sum_{i \in N} \sum_{q \in M} \mathbf{1}\{d_{iq} > 0\}. \tag{49}$$

Let X_i be a random variable representing the number of observations for part (N_i, M_i) of cells and SNVs. When we perform a `Branching` operation, our model specifies that we partition our data observations among six possible (cell, SNV) parts. Given the user-specified number $\delta \in \mathbb{N}$ representing the minimum number of observations per node and the partition of $(\phi(j), \Delta(\mathbf{y}_j))$ as in (48), we have that

$$P(X_1, X_2, \ldots, X_6 \geq \delta \mid \sum_{i=1}^{6} X_i = O(\phi(j), \Delta(\mathbf{y}_j)))$$

is the probability that each of the six parts has at least δ observations subject to $\sum_{i=1}^{6} X_i = O(\phi(j), \Delta(\mathbf{y}_j))$. This is a multinomial tail probability, *i.e.* $(X_1, \ldots, X_6) \sim \text{Multinomial}(O(\phi(j), \Delta(\mathbf{y}_j)), (1/6, \ldots, 1/6))$ where each of the $O(\phi(j), \Delta(\mathbf{y}_j))$ trials has six possible outcomes with uniform probabilities. We perform a power calculation, seeking the minimum number O^* of observations such that $P(X_1, X_2, \ldots, X_6 \geq \delta \mid \sum_{i=1}^{6} X_i = O^*)$ exceeds a user-specified probability γ. If $O(\phi(j), \Delta(\mathbf{y}_j)) < O^*$, we do not attempt to extend node v_j and opt to apply the `Identity` operation instead. Finally, we note that although the `Linear` operation results in four parts, we use six parts for the power calculation, resulting in a more conservative estimation.

Quality assessment of an extension. Even though we may have enough data observations to theoretically perform an elementary operation on a leaf node v_j in tree T, a high quality extension might not exist. We define the *cell mutational burden* (CMB) for a cell $i \in [N] \subseteq [n]$ and set $M \subseteq [m]$ of SNVs as

$$\text{CMB}(i, M) = \frac{\sum_{q \in M} \mathbf{1}\{a_{iq} > 0\}}{\sum_{q \in M} \mathbf{1}\{d_{iq} > 0\}}. \tag{50}$$

We define a quality check $\text{qc}(N, M)$ as $\text{median}\{\text{CMB}(i, M) \mid i \in [N]\}$. When performing a `Linear` operation, we expect $\text{qc}(\phi(v_j), \Delta(\mathbf{y}_a))$ to be high, *i.e.* at least 0.15, since we expect all $\Delta(\mathbf{y}_a)$ to be present in the set $\phi(v_j)$ of cells. Conversely, we expect $\text{qc}(\phi(v_a), \Delta(\mathbf{y}_b))$ to be low, *i.e.* at most 0.05, since we do not expect to see SNVs $\Delta(\mathbf{y}_b)$ in the set $\phi(v_a)$ of cells. For similar reasoning as above, we define the following quality checks for the `Branching` operation.

$$\begin{aligned}
&\text{qc}(\phi(j), \Delta(\mathbf{y}_{j'})) \geq 0.15, \\
&\text{qc}(\phi(a), \Delta(\mathbf{y}_b)) \leq 0.05, \\
&\text{qc}(\phi(b), \Delta(\mathbf{y}_a)) \leq 0.05.
\end{aligned}$$

We use 0.05 and 0.15 for illustrative purposes and because we set these as the default values. However, the user may specify different values for these parameters to be less conservative.

Sequencing artefact check. In order to avoid overfitting, we also assess whether the new nodes v_j, v_a, v_b for `Branching` and the nodes v_j, v_a for `Linear` contain a sufficient number of observations. To detect overfitting, we define the following two data checks (dc).

$$\begin{aligned}
\text{dc}_1(N, M) &= \text{median}\left\{\sum_{q \in M} \mathbf{1}(d_{iq} > 0) \mid i \in N\right\}, \\
\text{dc}_2(N, M) &= \text{median}\left\{\sum_{i \in N} \mathbf{1}(d_{iq} > 0) \mid q \in M\right\}.
\end{aligned}$$

Intuitively, dc_1 determines the median number of observations across cells, whereas dc_2 determines the median number of observations across SNVs. Given a user-specified threshold t, we require $\text{dc}_1(N, M), \text{dc}_2(N, M) \geq t$ for data check to pass. We set the default as $t = 4$. To accept a proposed extensions data checks $dc_1(N, M)$ and $dc_2(N, M)$ must pass for all parts $(N, M) \in \{(\phi(j), \Delta(\mathbf{y}_j)), (\phi(j), \Delta(\mathbf{y}_a)), (\phi(a), \Delta(\mathbf{y}_j)), (\phi(a), \Delta(\mathbf{y}_a))\}$ for a `Linear` extension. Conversely, we consider parts $(N, M) \in \{(\phi(a), \Delta(\mathbf{y}_a)), (\phi(a), \Delta(\mathbf{y}_b)), (\phi(b), \Delta(\mathbf{y}_a)), (\phi(b), \Delta(\mathbf{y}_b))\}$ to accept a `Branching` extension.

A.3.4 Postprocessing.

During tree inference, we remove all cells and SNVs from tree T with 0 variant reads and add them to a set N of cells and a set M of SNVs, respectively, prior to performing an elementary tree operation. After PHERTILIZER returns the clonal tree with maximum posterior probability, we then perform an optional post-processing phase that contains two stages: (i) the placement of cells and SNVs that were removed from the tree during inference and (ii) the reassignment of SNVs that fit poorly in the tree.

For post-processing, we are given a tree T, clonal genotypes $\mathbf{Y}$, cell clustering ϕ, set N of cells and a set M of SNVs. In the first stage, we first fix the cell clustering ϕ. Then, for each SNV $q \in M$ and each node $v_j \in V(T)$, we compute the likelihood that SNV q is gained at node v_j as follows.

$$L(q \in \Delta(\mathbf{y}_j)) = \sum_{v_{j'} \in \mathrm{desc}(v_j) \cup \{v_j\}} \sum_{i \in \phi(j')} \log P(a_{iq} \mid d_{iq}, y_{j'q} = 1, \theta) \tag{51}$$

$$+ \sum_{v_{j'} \in \mathrm{anc}(v_j)} \sum_{i \in \phi(j')} \log P(a_{iq} \mid d_{iq}, y_{j'q} = 0, \theta), \tag{52}$$

where $\mathrm{desc}(v_j)$ is the set of descendant nodes and $\mathrm{anc}(v_j)$ is the set of ancestor nodes of node v_j in tree T. We assign q to the node $v_j \in T$ where $L(q \in \Delta(\mathbf{y}_j))$ is maximized, resulting in updated clonal genotypes $\mathbf{Y}'$. We then set clonal genotypes $\mathbf{Y} := \mathbf{Y}'$.

Next, we repeat the above process but fix the clonal genotypes $\mathbf{Y}$ and update the cell clustering ϕ. For each cell $i \in N$ and for each node $v_j \in V(T)$ such that $\phi(j) \neq \emptyset$, we compute the log likelihood of cell i being placed at node v_j as follows.

$$L(i \in \phi(j)) = \sum_{v_{j'} \in \mathrm{anc}(v_j) \cup \{v_j\}} \sum_{q \in \Delta(\mathbf{y}_{j'})} \log P(a_{iq} \mid d_{iq}, y_{j'q} = 1, \theta) \tag{53}$$

$$+ \sum_{v_{j'} \in \mathrm{desc}(v_j)} \sum_{q \in \Delta(\mathbf{y}_{j'})} \log P(a_{iq} \mid d_{iq}, y_{j'q} = 0, \theta). \tag{54}$$

We assign cell i to the cell cluster $j \in [k]$, where $L(i \in \phi(j))$ is maximized, resulting in updated cell clustering ϕ'. We then set cell clusterings $\phi := \phi'$.

In the second stage, given a tree T and a cell clustering ϕ, we attempt to improve clonal genotypes $\mathbf{Y}$ by moving SNVs that fit poorly with the given cell clustering ϕ and tree T to a node in tree T with higher likelihood. We iterate over the nodes of tree T using a postorder traversal. At each node v_j, we identify poorly fitting SNVs by computing a metric we refer to as the binary variant alele frequency (BVAF), which is as follows for a given SNV q

$$\mathrm{BVAF}(q, v_j) = \frac{\sum_{v_{j'} \in \mathrm{desc}(v_j) \cup \{v_j\}} \sum_{i \in \phi(j')} \mathbf{1}\{a_{iq} > 0\}}{\sum_{v_{j'} \in \mathrm{desc}(v_j) \cup \{v_j\}} \sum_{i \in \phi(j')} \mathbf{1}\{d_{iq} > 0\}}. \tag{55}$$

In words, $\mathrm{BVAF}(q, v_j)$ is the fraction of cells in clade v_j with mapped variant reads at SNV q. We compute $\mathrm{BVAF}(q, v_j)$ for every SNV $q \in \Delta(\mathbf{y}_j)$. We identify a cutoff t at the 10th percentile of the distribution this distribution. Then, any SNV q where $\mathrm{BVAF}(q, v_j) \leq t$ is removed from the tree and placed in a candidate set M of SNVs to be reassigned. Reassignment then proceeds as described above.

We repeat this second stage of post-processing for either a fixed number of iterations or until no SNVs are reassigned, resulting in updated clonal genotypes $\mathbf{Y}'$. After termination, we update clonal genotypes $\mathbf{Y} := \mathbf{Y}'$ and return clonal genotypes $\mathbf{Y}$ and cell clustering ϕ.

A.4 BASELINE method

The BASELINE method seeks to infer SNV genotypes for cells utilizing copy-number information embedded in the binned read counts $\mathbf{R}$ via the following three steps. First, the method projects binned read counts $\mathbf{R}$ to a low-dimensional embedding $\tilde{\mathbf{R}}$. Second, it clusters cells into clones in the low-dimensional binned read count embedding $\tilde{\mathbf{R}}$. Third, each cluster of cells is treated as a pseudobulk sample and the cluster is genotyped. In our implementation, we used OPTICS [35] for the clustering step, since it is in the family of DBSCAN clustering methods that is a common choice used for the BASELINE in practice and the number of clusters does not need to be specified *a priori*. To genotype a cell cluster j, we set clonal genotype $y_{jq} = 1$ for each SNV $q \in [m]$ whenever the variant allele frequency, *i.e.* the total number of variant reads/total number of reads, was greater than 0.05 and $y_{jq} = 0$ otherwise. Fig. S1 shows good concordance of clonal genotypes using this approach with that of the inferred clonal genotypes on a DLP+ dataset [6].

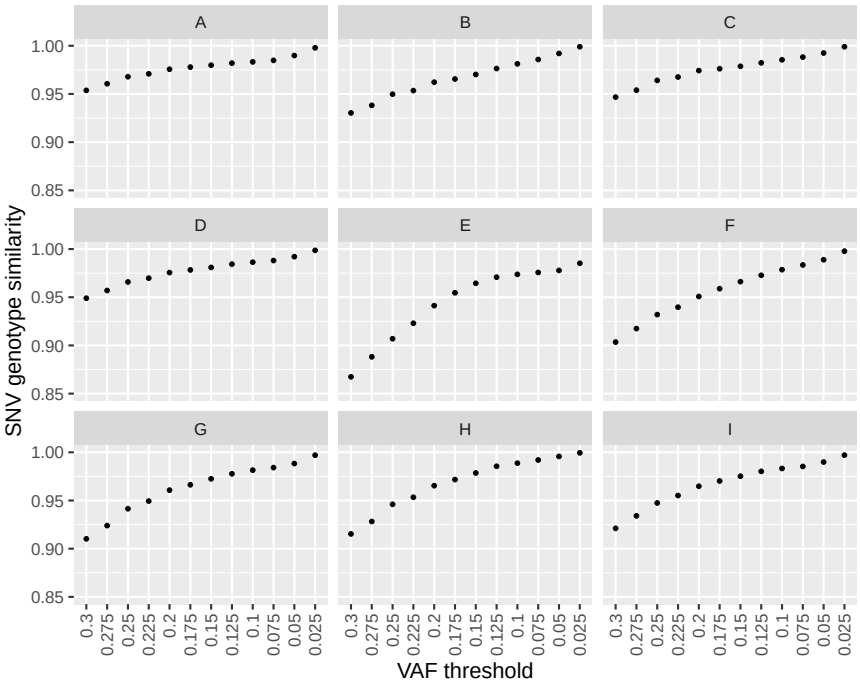


Figure S1: SNV genotype similarity between inferred genotypes per cell cluster using the BASELINE method and varying the variant allele frequency (VAF) threshold and inferred clonal genotypes per cell cluster by Laks et al. [6]

A.5 Processing the data from ACT

We merged reads from all cells into a single FASTQ file for each sample, aligned the merged reads to the Human reference genome hg19 (GRCh37) using bowtie2 (v.2.4.4), and sorted by samtools (v.1.15) forming pseudo-bulk samples. We then get a set of SNVs by running Mutect2 [30] in tumor-only mode on each pseudo-bulk sample, followed by FilterMutectCalls in GATK. We further select SNVs whose read depth is greater or equal to 11, VAF greater than 0.033, and have greater or equal to 4 variant reads in the pseudo-bulk sample. In order to match the bins in the original paper, we directly used the normalized read counts from ACT and find the low-dimensional embedding using UMAP [32].

B Supplementary Results

B.1 Simulation details

B.1.1 Simulation setup

We generated scDNA-seq data from *in silico* heterogeneous tumors with both CNAs and SNVs. We generated a tree T^* with $k \in \{5, 9\}$ nodes. We varied the number $n \in \{1000, 2000\}$ of sequenced cells and the number $m \in \{5000, 10000, 15000\}$ of SNVs. We generated read counts following a beta-binomial distribution with coverage of $0.01\times$, $0.05\times$, and $0.1\times$. We introduced 3 chromosome-level and 8 chromosome-arm level CNAs, affecting a total of $\ell = 577$ bins each of length 5 MB spanning all autosomes. We sampled a cell clustering ϕ^* as well as SNV and CNA placement on edges from a symmetric Dirichlet distribution with a concentration parameter of 2.

Each combination of simulation parameters was replicated with ten different random number generator seeds, amounting to a total of 180 experiments. In addition, we varied the SNV evolutionary model, using either a Dollo (SNV loss, [29]) or an infinite sites model (without SNV loss, [22]). We used a mean sequencing coverage of $0.01\times$ drawn from a Poisson distribution, to generate data matrices $\mathbf{D}, \mathbf{R}$ and sampled the variant count matrix $\mathbf{A}$ from a binomial distribution parameterized by the ground truth VAF.

B.2 Runtime parameters

PHERTILIZER. We used default parameters specified below for all simulation instances.

```
phertilizer -f  ${input_dataframe_tsv}$ \
    --bin_count_data ${input_reads_per_bin_file}$ \
    --radius 1 -a 0.001 \
    --min_obs 7 \
    -c 5 \
    --post_process \
    --use_copy_kernel \
    -d ${seed}$ \
    -j 50 \
    -s 16 \
    --low_cmb 0.05 \
    --high_cmb 0.15 \
    --nobs_per_cluster 3
```

SBMCLONE. We used the same seed for simulation as the seed for SBMClone.

```
python SBMClone/sbmclone.py -o ${out_dir} --seed $SEED \
    ${input_dataframe_tsv}

```

SCITE. We set the maximum MCMC chain length to be $900,000$, repetitions to 3, false positive rate to be 0.02, false negative rate to be 0.01, and seed to 42.

```
scite -i ${input} -n ${mut_clusters} -m ${cell_clusters} \
        -r 3 -l 900000 \
        -fd 0.02 -ad 0.01 \
        -names ${names} \
        -seed 42 \
        -a -o ${outprefix}
```

$$\begin{aligned}
\text{ancestral:}\ & \{(\mathbf{a},\mathbf{b}),(\mathbf{a},\mathbf{c}),(\mathbf{a},\mathbf{d}),(\mathbf{a},\mathbf{e}),(\mathrm{b},\mathrm{c}),(\mathrm{b},\mathrm{d}),(\mathbf{b},\mathbf{e})\} \\
\text{incomparable:}\ & \{\{\mathrm{c},\mathrm{e}\},\{\mathbf{d},\mathbf{e}\}\} \\
\text{clustered:}\ & \{\{\mathrm{c},\mathrm{d}\}\}
\end{aligned}$$

$$\begin{aligned}
\mathrm{APR} &= {}^{5}/_{7} \\
\mathrm{IPR} &= {}^{1}/_{2} \\
\mathrm{CPR} &= {}^{0}/_{1}
\end{aligned}$$

$$\begin{aligned}
\text{accuracy} &= \frac{7}{10}\mathrm{APR} + \frac{2}{10}\mathrm{IPR} + \frac{1}{10}\mathrm{CPR} \\
&= \frac{6}{10}
\end{aligned}$$

$$\begin{aligned}
\text{ancestral:}\ & \{(\mathbf{a},\mathbf{b}),(\mathbf{a},\mathbf{c}),(\mathbf{a},\mathbf{d}),(\mathbf{a},\mathbf{e}),(\mathbf{b},\mathbf{e}),(\mathrm{c},\mathrm{e})\} \\
\text{incomparable:}\ & \{\{\mathrm{b},\mathrm{d}\},\{\mathrm{c},\mathrm{d}\},\{\mathbf{d},\mathbf{e}\}\} \\
\text{clustered:}\ & \{\{\mathrm{b},\mathrm{c}\}\}
\end{aligned}$$

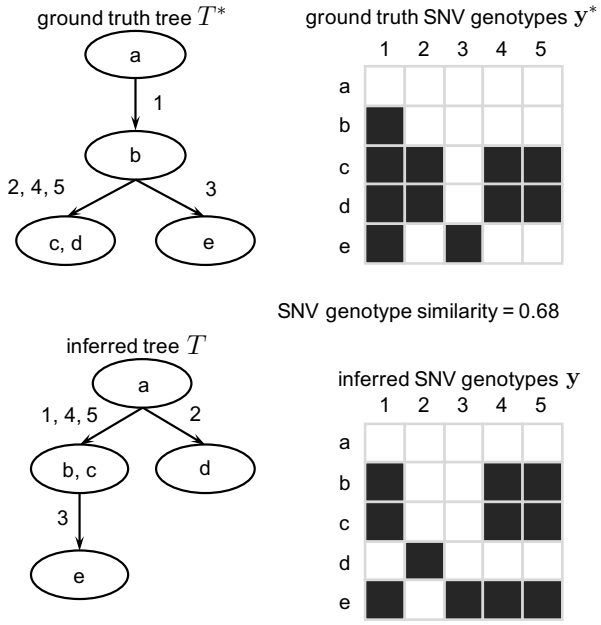
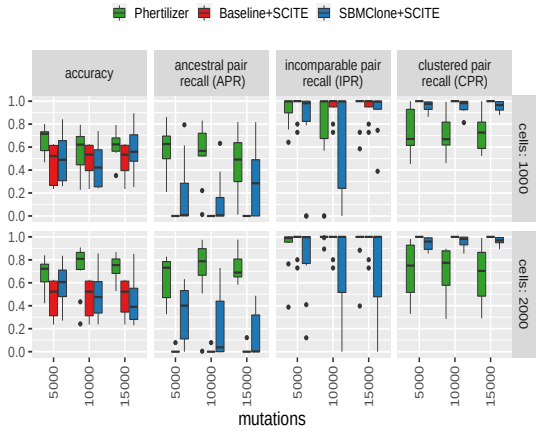
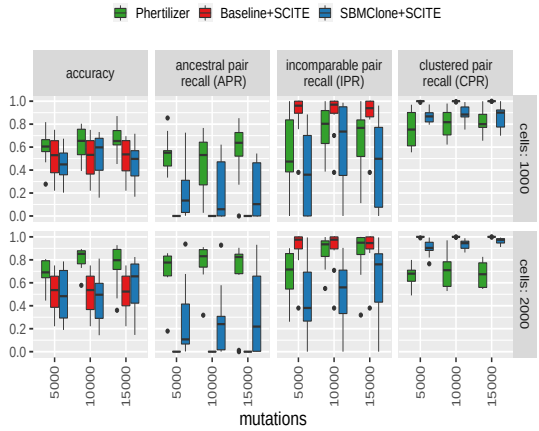


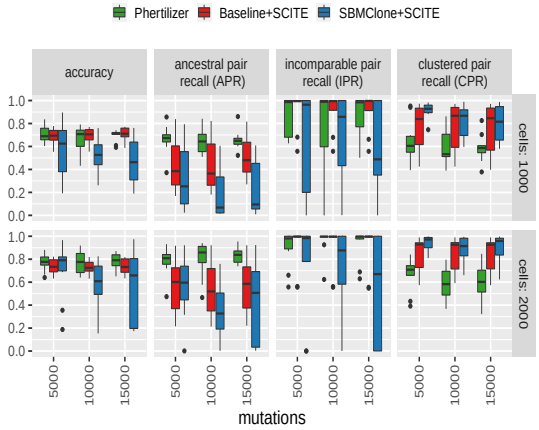
Figure S2: Example for cell APR, CPR, IPR, and accuracy. To the right of the trees, we show the ground truth and inferred SNV genotype $\mathbf{y}^*$ and $\mathbf{y}$ for the example.



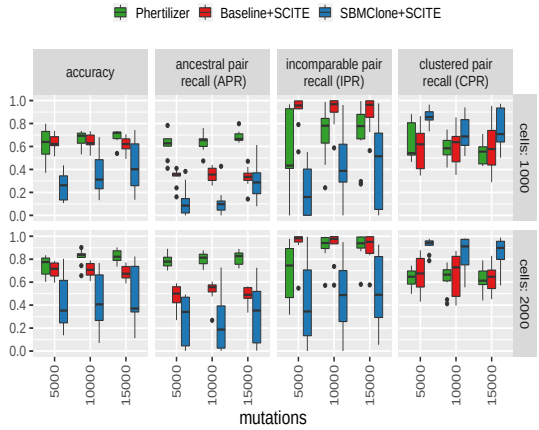
(a) cell placement metrics for $k = 5$ clones



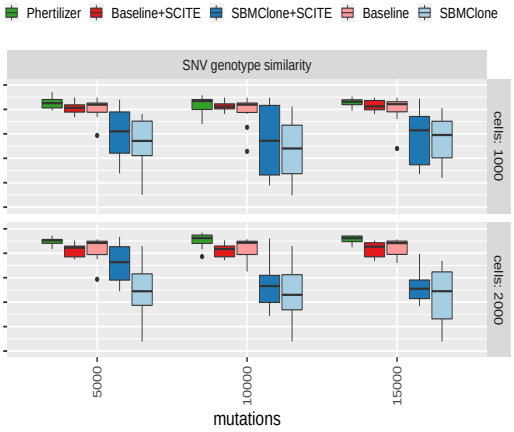
(b) cell placement metrics for $k = 9$ clones



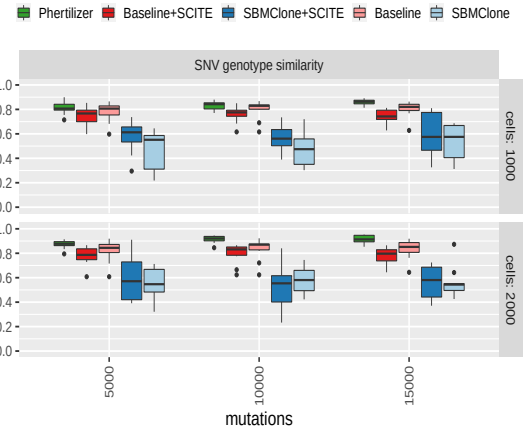
(c) SNV placement metrics for $k = 5$ clones



(d) SNV placement metrics for $k = 9$ clones

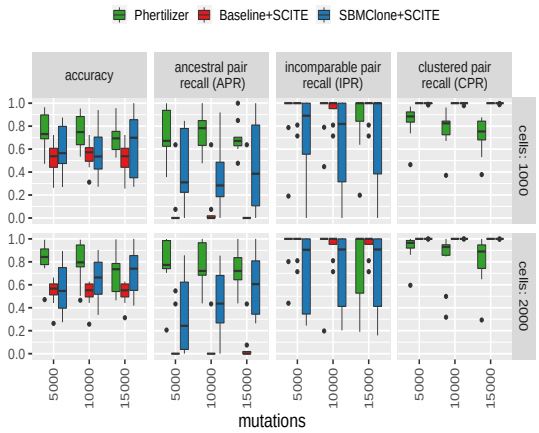


(e) SNV genotype similarity for $k = 5$ clones

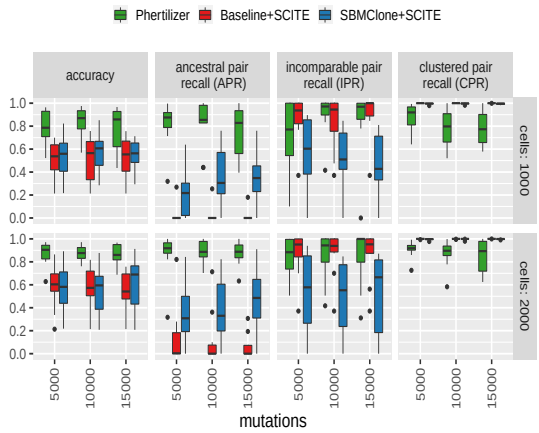


(f) SNV genotype similarity for $k = 9$ clones

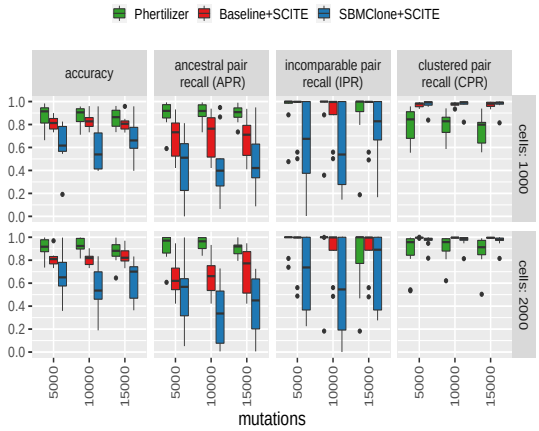
Figure S3: coverage $g = 0.01\times$



(a) cell placement metrics for $k = 5$ clones

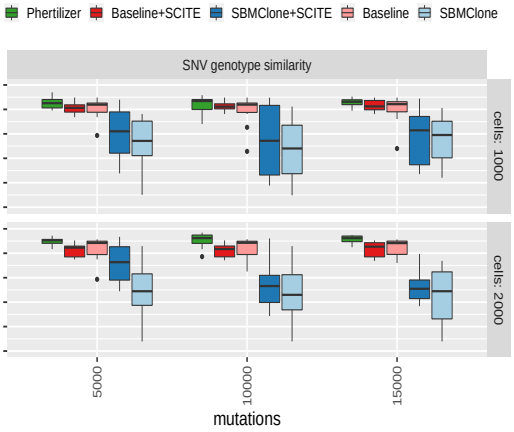


(b) cell placement metrics for $k = 9$ clones

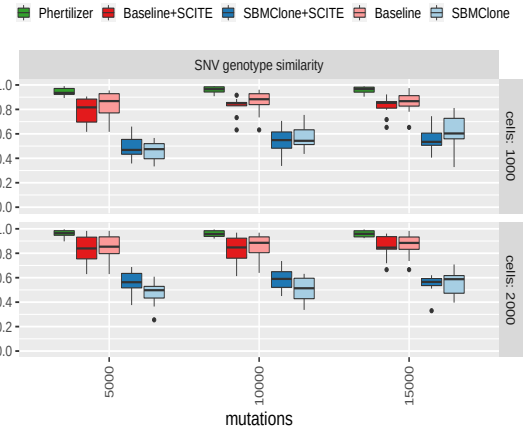


(c) SNV placement metrics for $k = 5$ clones

(d) SNV placement metrics for $k = 9$ clones

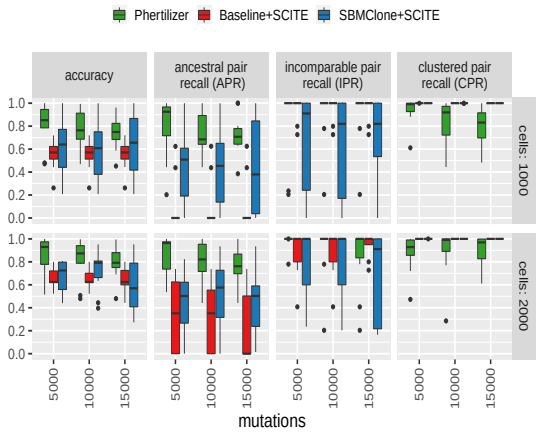


(e) SNV genotype similarity for $k = 5$ clones

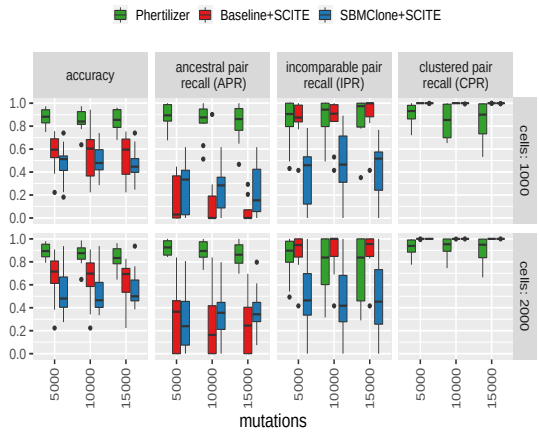


(f) SNV genotype similarity for $k = 9$ clones

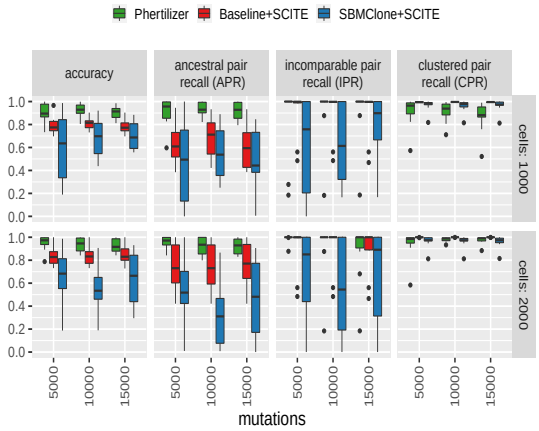
Figure S4: coverage $g = 0.05\times$



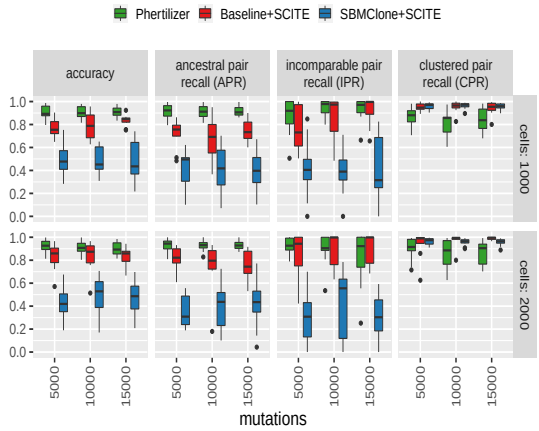
(a) cell placement metrics for $k = 5$ clones



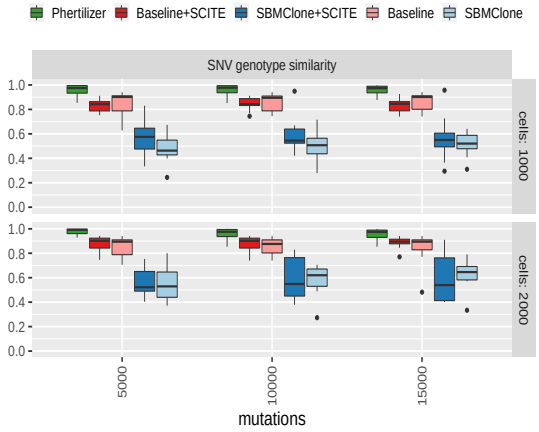
(b) cell placement metrics for $k = 9$ clones



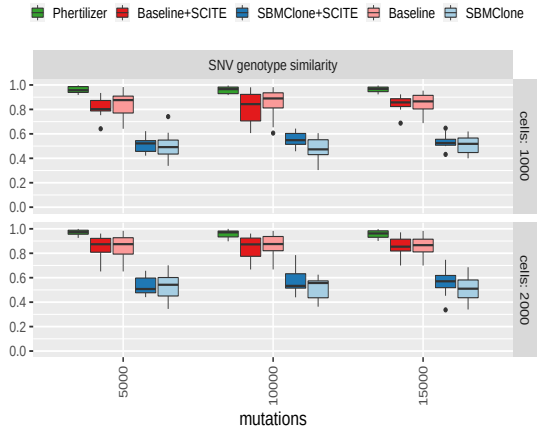
(c) SNV placement metrics for $k = 5$ clones



(d) SNV placement metrics for $k = 9$ clones

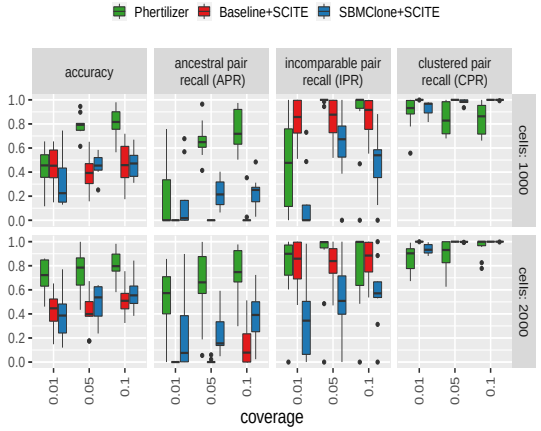


(e) SNV genotype similarity for $k = 5$ clones

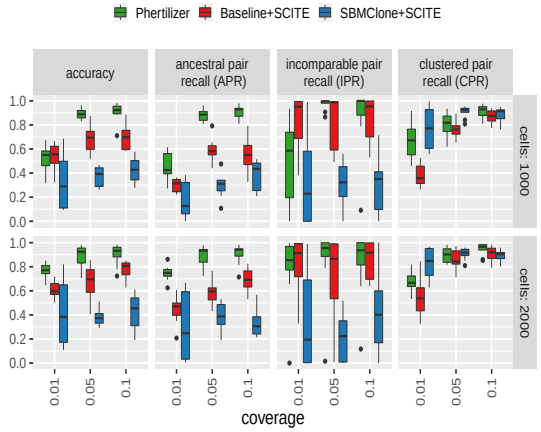


(f) SNV genotype similarity for $k = 9$ clones

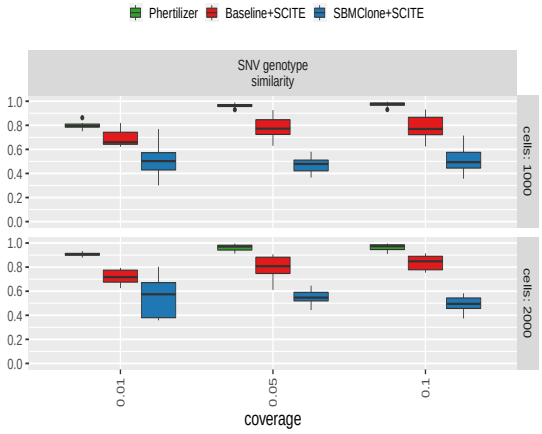
Figure S5: coverage $g = 0.1\times$



(a) cell placement metrics



(b) SNV placement metrics



(c) SNV genotype similarity

Figure S6: Dollo simulation results for $k = 9$ clones and $m = 15000$ SNVs

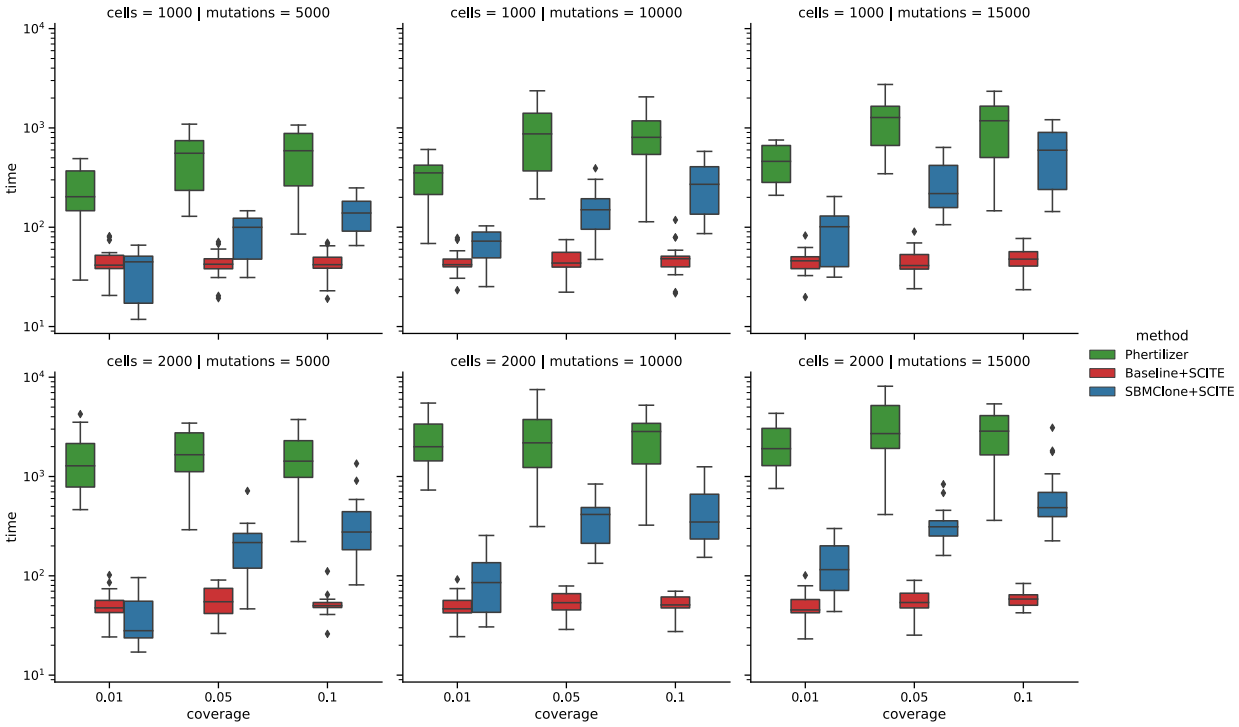
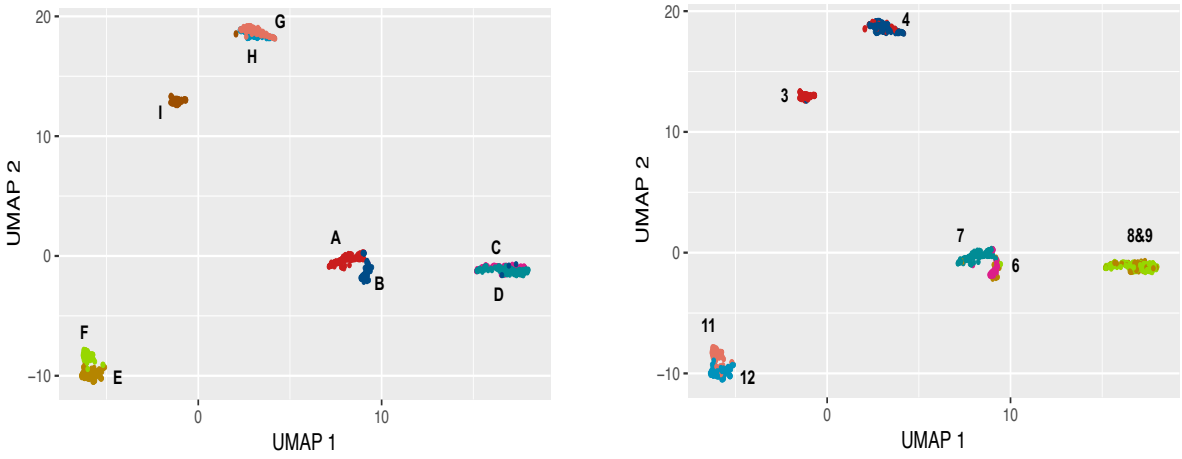


Figure S7: Running time (in seconds) on simulation data.



(a) Laks et al. [6] cell clusters

(b) Phertilizer cell clusters

Figure S8: UMAP for the high-grade serous ovarian cancer patient depicted with inferred cell clusterings.

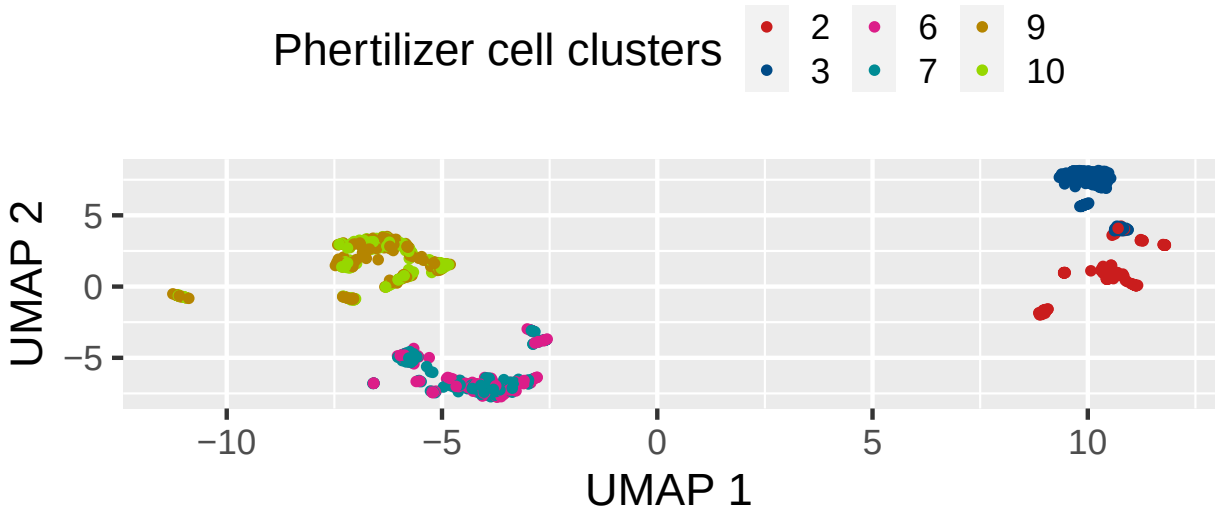


Figure S9: UMAP for the triple negative breast tumor TN1.

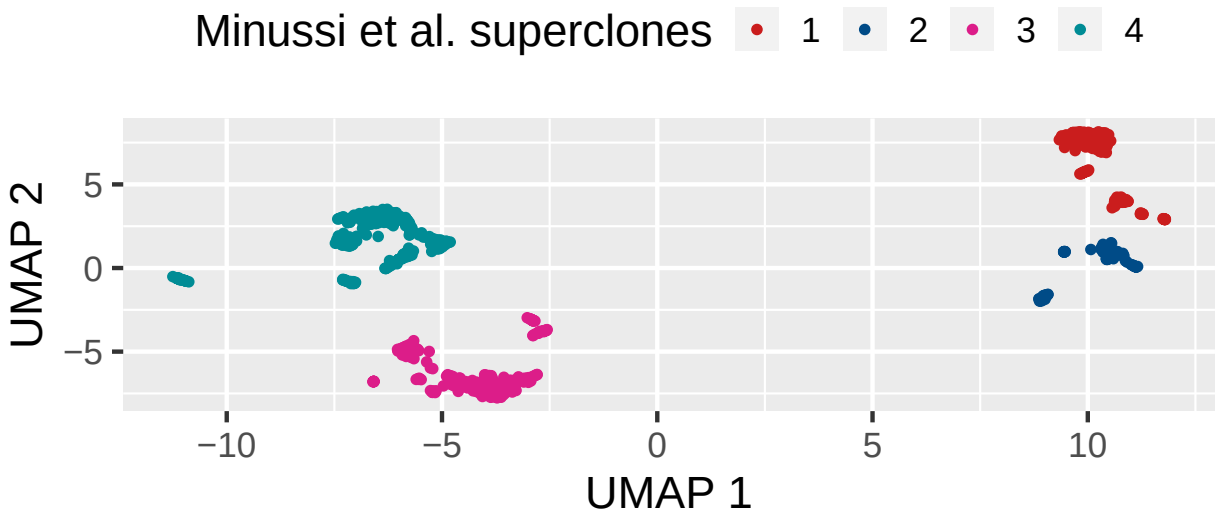
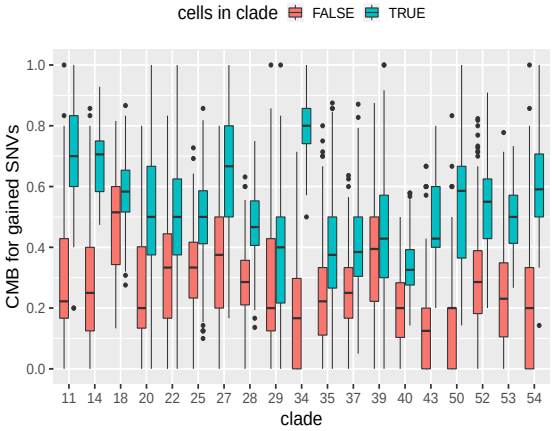
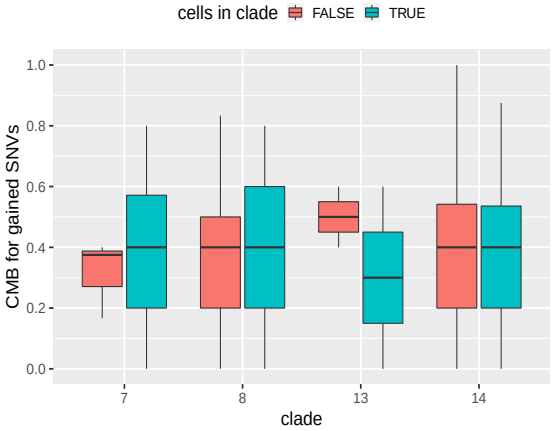


Figure S10: Minussi et al. [7] UMAP for the triple negative breast tumor TN1.



(a) Triple negative breast tumor TN3



(b) Triple negative breast tumor TN5

Figure S11: Cell mutational burden per clade in trees inferred by BASELINE+SCITE for tumors TN3 and TN5

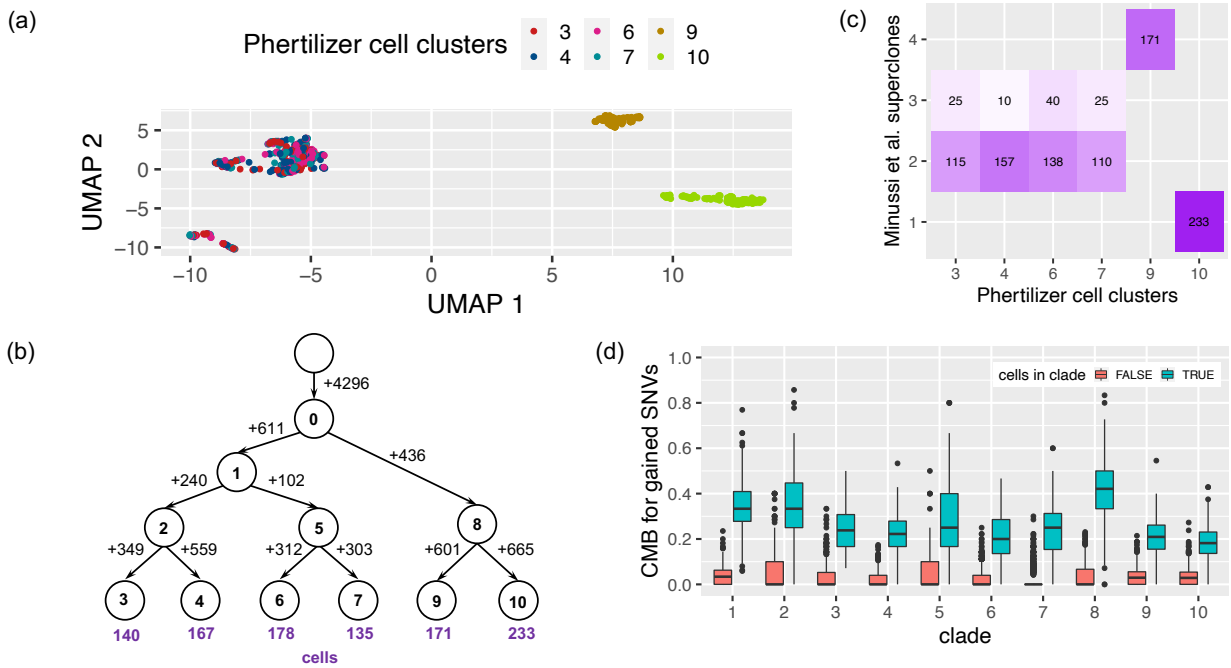


Figure S12: **PHERTILIZER infers clonal tree for breast cancer tumor TN2.** (a) UMAP for the triple negative breast tumor TN2. (b) The tree inferred by PHERTILIZER with numbers of SNVs labeled beside the edges, and numbers of cells labeled beneath the leaves. (c) A mapping between PHERTILIZER's cell clusters and the Minussi et al. [7] superclones. (d) The CMB distribution for the inferred clades for TN2.

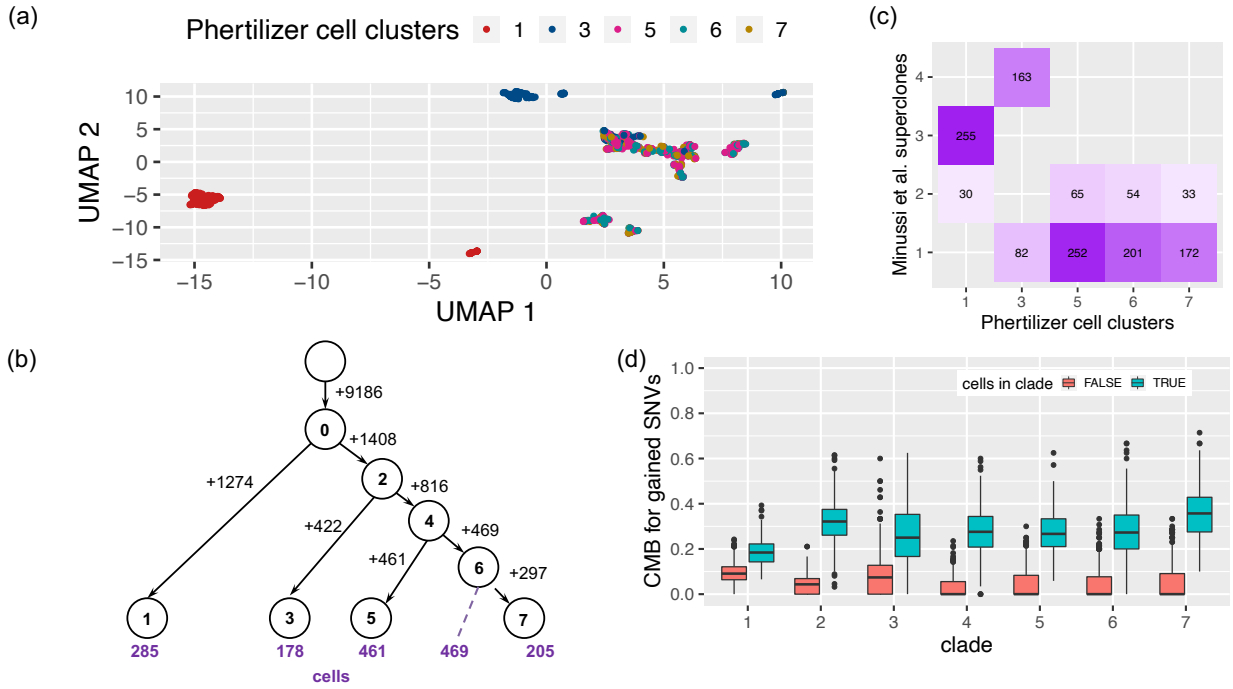


Figure S13: **PHERTILIZER infers clonal tree for breast cancer tumor TN4.** (a) UMAP for triple negative breast tumor TN4. (b) The tree inferred by PHERTILIZER with numbers of SNVs labeled beside the edges, and numbers of cells labeled beneath the leaves. (c) A mapping between PHERTILIZER's cell clusters and the Minussi et al. [7] superclones. (d) The CMB distribution for the inferred clades for TN4.

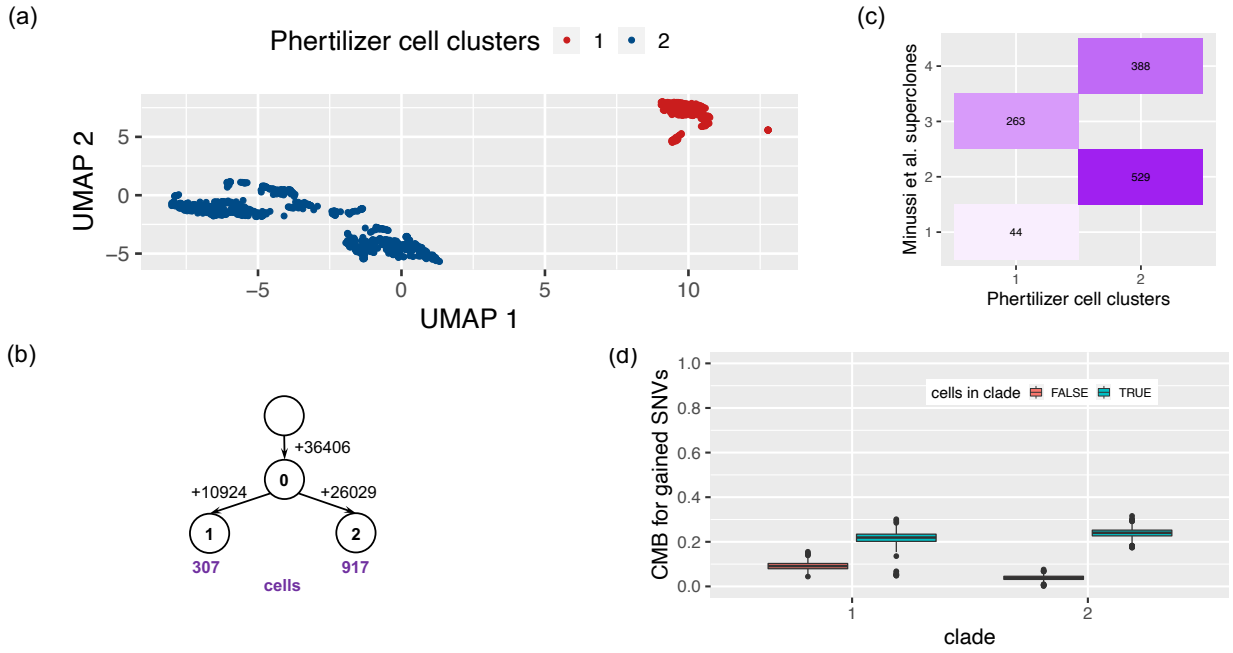


Figure S14: **PHERTILIZER infers clonal tree for breast cancer tumor TN8.** (a) UMAP for triple negative breast tumor TN8. (b) The tree inferred by PHERTILIZER with numbers of SNVs labeled beside the edges, and numbers of cells labeled beneath the leaves. (c) A mapping between PHERTILIZER's cell clusters and the Minussi et al. [7] superclones. (d) The CMB distribution for the inferred clades for TN8.