

# Reconstructing complex cancer evolutionary histories from multiple bulk DNA samples using Pairtree

Jeff A. Wintersinger<sup>1,2,3,4</sup>, Stephanie M. Dobson<sup>5,6</sup>, Lincoln D. Stein<sup>3,5</sup>, John E. Dick<sup>5,6</sup>,  
and Quaid D. Morris<sup>1,3,4,5,7,\*</sup>

<sup>1</sup>*Department of Computer Science, University of Toronto, Toronto, ON, Canada*

<sup>2</sup>*Donnelly Centre for Cellular and Biomolecular Research, University of Toronto, Toronto, ON, Canada*

<sup>3</sup>*Ontario Institute for Cancer Research, Toronto, ON, Canada*

<sup>4</sup>*Vector Institute for Artificial Intelligence, Toronto, ON, Canada*

<sup>5</sup>*Department of Molecular Genetics, University of Toronto, Toronto, ON, Canada*

<sup>6</sup>*Princess Margaret Cancer Centre, University Health Network, Toronto, ON, Canada*

<sup>7</sup>*Memorial Sloan Kettering Cancer Center, New York City, NY, USA*

\* *Corresponding author*

## 1 Abstract

Cancers are composed of genetically distinct subpopulations of malignant cells. By sequencing DNA from cancer tissue samples, we can characterize the somatic mutations specific to each population and build *clone trees* describing the evolutionary ancestry of populations relative to one another. These trees reveal critical points in disease development and inform treatment.

Pairtree is a new method for constructing clone trees using DNA sequencing data from one or more bulk samples of an individual cancer. It uses Bayesian inference to compute posterior distributions over the evolutionary relationships between every pair of identified subpopulations, then uses these distributions in a Markov Chain Monte Carlo algorithm to perform efficient inference of the posterior distribution over clone trees. Unlike existing methods, Pairtree can perform clone tree reconstructions using as many as 100 samples per cancer that reveal 30 or more cell subpopulations. On simulated data, Pairtree is the only method whose performance reliably improves when provided with additional bulk samples from a cancer. This suggests a shortcoming of existing methods, as more samples provide more information, and

26 should always make clone tree reconstruction easier. On 14 B-progenitor acute lymphoblastic leukemias  
27 with up to 90 samples from each cancer, Pairtree was the only method that could reproduce or improve  
28 upon expert-derived clone tree reconstructions. By scaling to more challenging problems, Pairtree sup-  
29 ports new biomedical research applications that can improve our understanding of the natural history  
30 of cancer, as well as better illustrate the interplay between cancer, host, and therapeutic interventions.  
31 The Pairtree method, along with an interactive visual interface for exploring the clone tree posterior, is  
32 available at <https://github.com/morrislab/pairtree>.

## 33 2 Introduction

34 Individual cancers exhibit substantial genetic heterogeneity, reflecting an ongoing evolutionary process  
35 of random somatic mutation and selection [1]. Cancers typically arise from a small number of founder  
36 mutations that confer a growth advantage [2]. Over time, additional somatic mutations accrue, and their  
37 frequency and distribution are shaped by evolutionary forces such as selection and genetic drift, resulting  
38 in the emergence of multiple genetically distinct cell subpopulations [3] (Fig. 1a). A *clone tree* is the  
39 evolutionary tree delineating the cell subpopulations in a cancer, the genetic mutations specific to each,  
40 and the proportions of cells in each sample that arose from each subpopulation (Fig. 1). Within the tree,  
41 *subclones* correspond to a cell subpopulation and all its descendants.

42 Clone trees built from bulk cancer samples have biologically and clinically important applications.  
43 Those built from single samples already reveal important genomic events in evolution [3, 4] and provide  
44 insights into heterogeneity [1]. But as sequencing costs continue to drop, sequencing different regions  
45 of the same tumour [5], multiple tumours of the same cancer [6], or longitudinal samples from different  
46 timepoints [7] will become more common. When bulk samples have different mixtures of subpopulations,  
47 each sample can provide unique information about the single clone tree that characterizes the cancer's  
48 evolutionary history. This can include revealing new subpopulations or disentangling single large sub-  
49 populations into smaller constituents. Clone trees built from multiple samples of the same cancer have  
50 helped identify factors associated with metastasis [8] and probed how treatment [9–11] or tumour mi-  
51 croenvironment [12, 13] shape evolution. This, in turn, can inform strategies to counteract treatment  
52 resistance [14].

53 Current subclonal reconstruction methods [15–21] are severely limited in their ability to build clone  
54 trees based on large multi-sample studies. Most of these methods were designed for single cancer samples  
55 from which no more than three subclones can be discerned at typical whole-genome sequencing depths

56 [1]. Recent studies with greater sequencing depth and multiple cancer samples have revealed that a single  
57 cancer can have dozens of resolvable subclones [10]. Here we show that existing clone tree reconstruction  
58 methods become highly inaccurate on datasets with many subclones or many cancer samples, necessitating  
59 a new approach.

60 Here we introduce Pairtree, a new method that can accurately construct clone trees from up to 100  
61 samples per cancer, revealing as many as 30 subclones. Pairtree outperforms a representative set of state-  
62 of-the-art clone tree reconstruction packages on simulated benchmark datasets of variable complexity.  
63 Pairtree is also the only method tested that can recover or improve on expert reconstructions of clone  
64 trees for 14 B-progenitor acute lymphoblastic leukemias (B-ALLs) containing up to 90 samples and 26  
65 subclones per cancer.

## 66 3 Methods and results

### 67 3.1 Pairtree inputs and outputs

68 A clone tree represents the evolutionary history of a cancer. Fig. 1 outlines the process of clone tree  
69 reconstruction. Pairtree takes as input allele frequency data for point mutations detected in one or more  
70 samples from a single cancer. These data can be derived from whole-genome sequencing (WGS), whole-  
71 exome sequencing (WES), or targeted sequencing. Each bulk cancer sample is a mixture of genetically  
72 heterogeneous cells (Fig. 1a). For each mutation, Pairtree uses counts of variant and reference reads in  
73 each sample to estimate the variant allele frequency (VAF), i.e., the proportion of reads at a mutation's  
74 locus that contain the mutation. By correcting a mutation's VAF for copy-number aberrations (CNAs)  
75 affecting the locus, Pairtree computes an estimate of the proportion of cells in each sample carrying the  
76 mutation, termed the mutation's *subclonal frequency* [22] (Fig. 1b).

77 Pairtree outputs a set of possible clone trees explaining evolutionary relationships between the input  
78 mutations. Clone tree nodes correspond to cancerous subpopulations, while arrows (i.e., directed edges)  
79 extend from a subpopulation's node to the nodes representing its direct descendants (Fig. 1c). We define  
80 a subpopulation as those cells containing exactly the same subset of the somatic mutations input into  
81 Pairtree. In each cancer sample, each subpopulation is assigned a population frequency, representing  
82 what proportion of cells in that sample share the same mutation subset. Many, if not most, of a cancer's  
83 mutations will not be provided in the input because of incomplete genome coverage or because the  
84 mutations are too low in frequency to be detected.

85 Each subpopulation and its descendant subpopulations (both direct and indirect) form a subclone

86 (Fig. 1a). Pairtree assigns a tree-constrained subclonal frequency to each subclone in each cancer sample,  
87 which is equal to the sum of the population frequencies of all the subpopulations contained within the  
88 subclone (Fig. 1a-b). This relationship follows from the infinite sites assumption (ISA), which states  
89 that no site is mutated more than once during cancer evolution. The ISA implies that subpopulations  
90 inherit all the mutations of their parent populations, and that each mutation appears only once in the  
91 evolutionary history of the cancer. Though violations of the ISA occur [23], it remains broadly valid [24],  
92 and Pairtree can detect and discard ISA-violating mutations (Section 6.1.3). Pairtree and most other  
93 clone tree reconstruction methods use the ISA, though some methods allow limited ISA relaxations [25–  
94 27]. Using the ISA, Pairtree identifies what mutations belong to each subclone based on the estimated  
95 subclonal frequencies provided by the VAF data (Fig. 1b), then searches for clone trees whose structures  
96 allow subclonal frequencies that best match these estimates (Fig. 1c). Pairtree’s output consists of a set  
97 of clone trees, each scored by a likelihood indicating how well the tree-constrained subclonal frequencies  
98 match the frequency estimates given by the VAF data. Although there is a single true clone tree explaining  
99 how subpopulations are related, this tree is not observed directly, and the input data often permit multiple  
100 solutions.

101 Grouping mutations into subclones is not necessary—algorithms can instead build clone trees in which  
102 each mutation is assigned to a unique subclone, yielding a mutation tree. However, because of limited  
103 resolution in the data’s estimated subclonal frequencies, sets of mutations often have subclonal frequency  
104 estimates that are too similar to separate the mutations into distinct subclones. As such, the first step in  
105 clone tree reconstruction is often clustering mutations with similar estimated subclonal frequencies across  
106 all input samples, and associating subclones with these clusters. Mutation clustering can be performed  
107 with Pairtree (Section 10.1.1) or by another method [28–30] and input into Pairtree. This step simplifies  
108 clone tree reconstruction by reducing the number of subclones. Additionally, this approach permits  
109 more precise estimates of each subclone’s subclonal frequency by combining data from the subclone’s  
110 mutations (Section 6.2.8), at the risk of grouping together mutations from different subclones. As more  
111 cancer samples are used, each of which provides separate subclonal frequency estimates for the mutations,  
112 this caveat becomes less problematic.

## 113 **3.2 Delineating ancestral relationships between pairs of subclones using the** 114 **Pairs Tensor**

115 Pairtree uses the estimated subclonal frequencies to predict the ancestral relationship between every  
116 subclone pair. These pairwise relationships then serve as a guide when Pairtree searches for clone trees  
117 that best fit the VAF data. Under the ISA [31], one of three mutually exclusive ancestral relationships  
118 exist between a pair of subclones  $A$  and  $B$ .

- 119 1.  $A$  is ancestral to  $B$ . Here, the subpopulation associated with  $A$  contains  $A$ 's mutations but not  
120  $B$ 's. No cell subpopulation has  $B$ 's mutations without also inheriting  $A$ 's.
- 121 2.  $B$  is ancestral to  $A$ . This is as above, with the roles of  $A$  and  $B$  switched.
- 122 3. Neither  $A$  nor  $B$  is the ancestor of the other. In this case, they occur on different branches of the  
123 clone tree. Consequently, no subpopulations have both  $A$ 's and  $B$ 's mutations.

124 Each relationship constrains the frequencies that can be assigned to the two subclones (Section 6.1.3).  
125 For a given subclone pair, Pairtree combines the CNA-corrected VAF data for each subclone's mutations  
126 with a prior probability distribution incorporating these constraints, then uses Bayesian inference to  
127 compute the probability of each relationship type for the pair (Section 6.1). This yields a data structure  
128 termed the *Pairs Tensor*, the elements of which are the marginal posterior probability distributions over  
129 the three possible ancestral relationships for every subclone pair.

## 130 **3.3 Using pairwise ancestry to guide the search for clone trees**

131 Pairtree uses the Pairs Tensor to define a proposal distribution for a Markov Chain Monte Carlo (MCMC)  
132 algorithm [32] that samples from the posterior distribution over clone trees (Fig. 2). The algorithm's  
133 Metropolis-Hastings scheme generates proposal trees using two discrete distributions derived from the  
134 Pairs Tensor (Section 6.2.5). The first distribution helps choose an erroneous subclone to move within the  
135 tree, with each subclone's selection probability determined by how inconsistent its ancestral relationships  
136 to other subclones in the current tree are relative to the Pairs Tensor. The second distribution guides  
137 the choice of new parent for the selected subclone, evaluating potential destinations based on how much  
138 this inconsistency is reduced. Though other MCMC-based subclonal reconstruction methods also modify  
139 trees by moving subclones [15, 17, 33], they blindly select both the subclone to move and its destination.  
140 Pairtree, by contrast, considers the data when making these decisions, with the Pairs Tensor helping the  
141 method rapidly navigate to high-probability regions of clone-tree space.

142     Pairtree uses a MAP approximation of the clone tree’s marginal likelihood, both for the Metropolis-  
143     Hastings accept-reject decision and to compute the tree’s posterior probability. Computing a clone  
144     tree’s likelihood requires a maximum a posteriori (MAP) estimate of the subclonal frequencies, using  
145     a Bayesian prior to enforce tree constraints. By this prior, the root subclone must have a subclonal  
146     frequency of 1 in every sample, as it corresponds to the germline and all subclones are descended from  
147     it. Additionally, the prior requires that every subclone has a frequency greater than or equal to the  
148     sum of its direct descendants’ subclonal frequencies. Pairtree computes the MAP estimate using a fast  
149     approximate scheme [34] or a slower exact one (Section 6.3). A clone tree’s likelihood is then defined by  
150     how well the variant and reference read counts for each mutation match the MAP subclonal frequencies  
151     under a binomial sequencing noise model.

### 152     **3.4 Benchmarking Pairtree performance using novel scoring metrics**

153     Evaluating Pairtree against other common subclonal reconstruction methods required developing new  
154     metrics, as previously developed metrics are limited to datasets with single cancer samples [21]. Here,  
155     we introduce two novel metrics better suited for the multi-sample domain that also permit uncertainty  
156     about the best-fitting clone tree.

157     The first, termed *VAF reconstruction loss*, uses likelihood to compare the data fit of a tree’s subclonal  
158     frequencies to a baseline (Section 6.5.2). For simulated data, the baseline frequencies are the ground-truth  
159     frequencies used to generate the VAF data. For real data with an unknown ground truth, the baseline is  
160     MAP subclonal frequencies computed for an expert-constructed clone tree. Negative VAF losses indicate  
161     the evaluated frequencies have better data fit than the baseline.

162     The second evaluation metric, termed *relationship reconstruction error*, compares the structure of  
163     candidate clone trees to the ground truth (Section 6.5.3) using the evolutionary relationships between  
164     subclone pairs. To compute it, we construct an empirical Pairs Tensor from the clone tree solutions  
165     found by a method, then compare it via the Jensen-Shannon divergence (JSD) to a tensor based on the  
166     ground truth. As multiple clone trees may be consistent with the ground-truth subclonal frequencies,  
167     we construct the ground-truth Pairs Tensor by enumerating all trees consistent with these frequencies  
168     [35] and denoting the pairwise relationships between subclones that each expresses. Building this ground  
169     truth requires knowing the ground-truth subclonal frequencies with no measurement error, so this metric  
170     is best suited to simulated data.

171     For both metrics, we evaluate the quality of a solution set by computing the average over all trees  
172     reported by a method, weighted by the likelihood the method associates with each solution.

### 173 **3.5 Selecting comparison methods and generating simulated data**

174 Clone tree reconstruction methods use one of two approaches: exhaustive enumeration or stochastic  
175 search. To evaluate Pairtree, a stochastic search method, we compared it against three exhaustive  
176 enumeration methods (PASTRI [20], CITUP [16], and LICHeE [19]) and one stochastic search method  
177 (PhyloWGS [36]). All methods output multiple candidate clone trees.

178 We assessed method performance on 576 simulated datasets with variable read depths and numbers of  
179 subclones, cancer samples, and mutations. These included trees with 3, 10, 30, or 100 subclones. Three  
180 subclones are the most that can typically be resolved at WGS read depths of 50x [1]. Ten subclones  
181 are often discernible from multi-sample datasets [5], while 30 was the approximate maximum we could  
182 resolve in the high-depth, many-sample B-ALL data evaluated here [10]. We included datasets with 100  
183 subclones to probe the methods' limits, anticipating challenges presented by future datasets. The number  
184 of simulated cancer samples ranged from 1 to 100. We designed the simulation process (Section 6.4.2)  
185 to generate realistic, diverse, and resolvable clone trees (Section 10.7). We did not include one- or three-  
186 sample datasets in the 30- and 100-subclone simulations, as resolving so many subclones from so few  
187 samples would be unrealistic. Methods were allowed up to 24 hours of wall-clock time to produce results.

188 Some caveats must be noted. LICHeE does not report subclonal frequencies for its solutions, so we  
189 used Pairtree to fit MAP frequencies to LICHeE's trees. Though LICHeE does not produce a likelihood,  
190 unlike the other methods here, it reports an error score for each tree that we interpreted as a likelihood  
191 when weighting its solutions. PhyloWGS, unlike other methods, could not use a fixed mutation clustering.  
192 This led to the method incorrectly merging clusters, causing artificially high VAF loss and relationship  
193 error. More generally, all methods except Pairtree failed to produce output on some simulated datasets.  
194 These failures stemmed from methods terminating without producing output, crashing outright, or failing  
195 to finish within 24 hours (see Section 10.3 for details).

### 196 **3.6 Pairtree outperforms existing methods on simulated data**

197 Fig. 3 summarizes how the methods performed on simulated data, with a method's scores reflecting its  
198 performance on only the datasets for which it produced output. Pairtree was the only method that  
199 produced results for all 576 simulations (Fig. 3a). Nevertheless, Pairtree fared better than comparison  
200 methods on trees with 30 or fewer subclones, succeeding on all datasets while achieving negative median  
201 VAF losses (Fig. 3b-c). In fact, Pairtree always produced lower error than other methods for every such  
202 dataset (Fig. S4). Pairtree also performed better than comparison methods with respect to relationship

203 error. In general, for 30 subclones or fewer, relationship error was almost zero when the number of cancer  
204 samples exceeded the number of subclones (Fig. S5b). For these cases, only one possible tree occurred  
205 (S10a), with Pairtree achieving low error by finding that tree or a close approximation thereof (S10b-c).  
206 When applied to datasets with 100 subclones, Pairtree had higher VAF losses (Fig. 3b) and relationship  
207 errors (Fig. 3c) than with fewer subclones. Pairtree outperformed other methods for 100-subclone trees  
208 with respect to VAF loss, except for 16 datasets (15%) where PhyloWGS performed better (Fig. S4).

209 CITUP failed on all datasets with ten or more subclones, and on 32% of three-subclone cases (Fig. 3a).  
210 All failures on three-subclone datasets occurred because CITUP crashed (Section 10.3). On ten-subclone  
211 datasets, 29% of CITUP runs ran out of time, with the other 71% failing because CITUP crashed. On  
212 the three-subclone cases where it ran successfully, its VAF loss was poor (Fig. 3b), perhaps because of a  
213 mismatch between its sequencing error model and the model used for computing VAF loss. Conversely,  
214 the method exhibited better relationship error than other non-Pairtree methods (Fig. 3c), suggesting its  
215 tree structures were more accurate.

216 PASTRI, which cannot run on datasets with more than 15 subclones [37], failed for 83% of three-  
217 subclone cases and 96% of ten-subclone cases (Fig. 3). For datasets with three or ten subclones, PASTRI  
218 produced output on 10%, terminated without producing a result on 84%, and ran out of time on the  
219 remaining 6% (Section 10.3). When it produced solutions, PASTRI generally performed well, reaching  
220 negative median VAF losses for three- and ten-subclone datasets, and relatively low relationship errors.  
221 Occasionally, PASTRI produced high-error solutions, with VAF losses up to 492 bits on the three-subclone  
222 datasets.

223 LICHeE fared better, producing results on all cases with 3, 10, or 30 subclones (Fig. 3). However,  
224 the method ran out of time for 92% of 100-subclone datasets. After Pairtree, LICHeE was the next-best  
225 performing method, with low VAF losses and moderate relationship errors on datasets with three or  
226 ten subclones, beating PhyloWGS on both measures. LICHeE performed less well on 30-subclone cases,  
227 where it exhibited lower VAF losses than PhyloWGS but higher relationship errors.

228 PhyloWGS produced clone trees for all datasets with 30 or fewer subclones (Fig. 3). In these cases,  
229 PhyloWGS generally had worse VAF losses and relationship errors than Pairtree or LICHeE, except  
230 for the 30-subclone datasets, where it had better relationship error than LICHeE but worse VAF loss.  
231 PhyloWGS performed better than other non-Pairtree methods on 100-subclone cases, where it finished  
232 within 24 hours for 62% of such datasets, but usually had higher VAF losses than Pairtree (Fig. S4).

233 Relationship error can also be measured for the Pairs Tensor alone, without requiring trees. The Pairs  
234 Tensor estimates pairwise relationships accurately (Fig. 3c), requiring only a fraction of the computational



235 resources of the full Pairtree method (Fig. S8). Although the Pairs Tensor does slightly worse than  
236 Pairtree on trees with 30 or fewer subclones, it has less relationship error than any other method. On  
237 datasets with 100 subclones, the Pairs Tensor was better able to delineate pairwise relationships between  
238 subclones than the full Pairtree method (Fig. 3c).

### 239 **3.7 Pairtree improves with more cancer samples, but other methods worsen**

240 After controlling for other variables, all methods except Pairtree performed worse when provided more  
241 cancer samples. CITUP and PASTRI's failure rates increased with the number of cancer samples (Fig. 4a).  
242 Though LICHeE and PhyloWGS produced output for all cases with 30 subclones or fewer, they had higher  
243 VAF losses with more cancer samples (Fig. 4b). By contrast, Pairtree never failed and had nearly zero  
244 median VAF loss regardless of the number of simulated cancer samples on datasets with 30 subclones  
245 or fewer (Fig. 4a-b). Relationship errors decreased for both full Pairtree and the Pairs Tensor with  
246 more samples (Fig. 4c). LICHeE, conversely, exhibited rapidly increasing error with more samples, while  
247 PhyloWGS' performance fluctuated.

### 248 **3.8 Pairtree performs better than human experts on complex real clone tree** 249 **reconstructions**

250 We applied Pairtree, CITUP, LICHeE, PASTRI, and PhyloWGS to genomic data from 14 B-ALL patients  
251 [10]. Samples were obtained at diagnosis and relapse for each patient. In addition, each sample was  
252 transplanted into immunodeficient mice, generating multiple patient-derived xenografts (PDXs). The  
253 patient samples were subjected to WES, while the PDXs were subjected to targeted sequencing based  
254 on leukemic variants found in the patient WES data. There were 16 to 509 mutations called per patient  
255 (median 40), clustered into 5 to 26 subclones per patient (median 8). By combining patient and PDX  
256 samples, we obtained between 13 and 90 tissue samples per cancer (median 42). Across cancers, the  
257 median read depth was 212 reads.

258 To define ground truth for these datasets, we built high-quality clone trees for each dataset manually,  
259 subjecting them to extensive review and refinement before evaluating them for biological plausibility [10].  
260 We then fit MAP subclonal frequencies to these trees using Pairtree, yielding the *expert-derived baseline*.  
261 As with simulated data, methods that improve on the baseline achieve negative VAF losses.

262 CITUP and PASTRI failed on 13 of the 14 cancers, and so we excluded these methods from the  
263 comparison. Pairtree found trees as good as, or slighter better than, the expert baseline for 12 of 14

264 cancers (Fig. 5), resulting in VAF losses between 0 and -0.05 bits. On two cancers, Pairtree inferred clone  
265 trees that fit the VAF data substantially better than the expert baseline, resulting in negative losses of  
266 -0.32 bits and -1.42 bits. LICHeE beat the baseline for one cancer, reaching a negative loss of -0.86 bits;  
267 (nearly) matched the baseline for four other patients, incurring between 0 and 0.11 bits of loss; and had  
268 substantially worse VAF losses for the remaining nine patients. PhyloWGS suffered at least 0.35 bits  
269 of loss on all patients, reaching a median VAF loss of 4.42 bits. As PhyloWGS could not adhere to the  
270 expert-derived clustering, unlike other methods, it often merged clusters incorrectly, causing high VAF  
271 loss.

### 272 3.9 Consensus graphs intuitively illustrate uncertainty in clone trees

273 Pairtree provides interactive visualizations to help navigate the multiple clone tree solutions that it  
274 produces for each dataset (Fig. 6). By using the data likelihoods associated with each solution as weights,  
275 Pairtree produces a *weighted consensus graph*, in which the nodes represent subclones, and each directed  
276 edge is assigned a weight equal to the marginal probability that it appears in a clone tree drawn from  
277 the empirical clone tree distribution produced by Pairtree. Thus, the consensus graph summarizes the  
278 estimated posterior probability of each parental relationship between subclones. These summaries are  
279 useful for interpreting Pairtree’s results, as they provide a concise representation of the evolutionary  
280 relationships supported by the data, alongside the confidence underlying each. By taking the maximum-  
281 weight spanning tree of this graph, the user can generate a single consensus tree. To demonstrate  
282 the consensus graph’s utility, we ran Pairtree multiple times on one of the B-ALL cases from Fig. 5,  
283 using variable numbers of cancer samples (Fig. 6). As we provided more cancer samples, confidence in  
284 evolutionary relationships increased, until all parents were resolved with near certainty. Providing more  
285 samples can also correct erroneous inferences—with 30 samples, population 8 appeared to be the likely  
286 parent of population 15, but with 90 samples, it became clear that population 15’s parent is population  
287 6.

## 288 4 Discussion

289 Pairtree is the first automated method that reliably recovers large, complex clone trees from bulk DNA  
290 sequencing data. On simulated data, Pairtree recovers nearly perfect clone trees for cancer datasets with  
291 up to 30 subclones. On 14 B-ALL cancers, with up to 26 subclones and 90 samples per cancer, Pairtree’s  
292 clone trees are objectively as good as, or better than, those manually constructed by experts. No other

293 tested method was consistently accurate on real or simulated benchmarks containing ten subclones or  
294 more. Pairtree was also the only method whose clone trees reliably became more accurate when more  
295 samples were used in the reconstructions. This is surprising—as each cancer sample provides additional  
296 information about evolutionary relationships between subpopulations, subclonal reconstruction problems  
297 should become easier with more cancer samples, not more difficult.

298 A key factor in Pairtree’s success is its efficient search through the space of clone trees. Beyond ten  
299 subclones, this tree space quickly becomes too large for exhaustive enumeration (CITUP) or unguided  
300 stochastic search (PhyloWGS). Even methods that reduce the search space by applying hard constraints  
301 excluding some parent-child relationships (LICHHeE, PASTRI) still fail to recover more complex clone  
302 trees. Recovering complex trees requires more cancer samples than for simple trees, but when faced with  
303 many samples, the hard constraints become inaccurate and exclude the correct solution (Section 10.4).  
304 By contrast, Pairtree’s stochastic tree search is guided by the Pairs Tensor, which provides soft constraints  
305 defined by a well-motivated probability model. Consequently, Pairtree’s constraints become more precise  
306 as more cancer samples are provided, without excluding the true clone tree.

307 As Pairtree’s performance degrades on the 100-subclone benchmarks, alternative search strategies  
308 may be necessary for very large clone trees. While Pairtree almost always fails to correctly resolve a  
309 subclone’s parent (Fig. S10c), it achieves relatively low relationship error (Fig. S10d), suggesting it may  
310 be capturing the coarse tree structure. If so, Pairtree may fare better using a tiered approach, in which  
311 it would group together subclones with similar pairwise relations to others, build subtrees for each group  
312 separately, and then connect the subtrees using the groups’ pairwise relations to compose the full clone  
313 tree. Given 100 subclones with 10 or more cancer samples, the Pairs Tensor is already better than Pairtree  
314 itself at capturing the correct evolutionary relationships between subclones (Fig. S5b-c). Future work  
315 should focus on understanding what conditions (e.g., high read depth or many cancer samples) under  
316 which the Pairs Tensor converges to a partial clone tree [35] that succinctly summarizes all clone trees  
317 with non-negligible posterior probability.

318 Throughout this work, we have stressed performance metrics that recognize there are often many  
319 solutions consistent with observed data (Section 10.6). These metrics extend previous ones we developed  
320 [21] to score multiple candidate solutions from a method against a single ground-truth tree. Our new  
321 metrics permit the ground truth to be uncertain, with multiple potential truths equally consistent with  
322 noise-free observations. In general, characterizing uncertainty in clone tree reconstructions is critical.  
323 Even when methods produce multiple solutions, users typically want a single answer, and so select  
324 the highest-scoring tree while neglecting other credible candidates that fit their data nearly as well.

325 Consequently, they lose information about which evolutionary relationships between subclones are well-  
326 defined by the data, and which are uncertain because they have multiple equally likely possibilities. If  
327 users are to benefit from a method's ability to produce multiple solutions, the method must provide  
328 tools for interpreting this uncertainty. Pairtree's weighted consensus graph characterizes the uncertainty  
329 present in each evolutionary relationship, depicting all credible possibilities and the confidence underlying  
330 each (Fig. 6). This allows users to make informed conclusions about their cancer datasets.

331 In summary, Pairtree can reconstruct highly accurate trees representing the evolutionary relationships  
332 among up to 30 subclones based on sequencing data from up to 100 samples from a cancer. By scaling  
333 to many more subclones and cancer samples than past approaches, and by illustrating the uncertainty  
334 present in solutions, Pairtree can address questions in many cancer research domains. These include  
335 understanding the origin and progression of tumours, measuring tumour age and heterogeneity, mapping  
336 out mechanisms of tumour adaptation to therapy, and understanding the relationship between primaries  
337 and metastases. In the future, the Pairtree framework can be extended to scale to even more complex  
338 trees, integrate single-cell sequencing data (Section 10.9), and permit violations of the infinite sites  
339 assumption (Section 10.8).

340 5 Figures

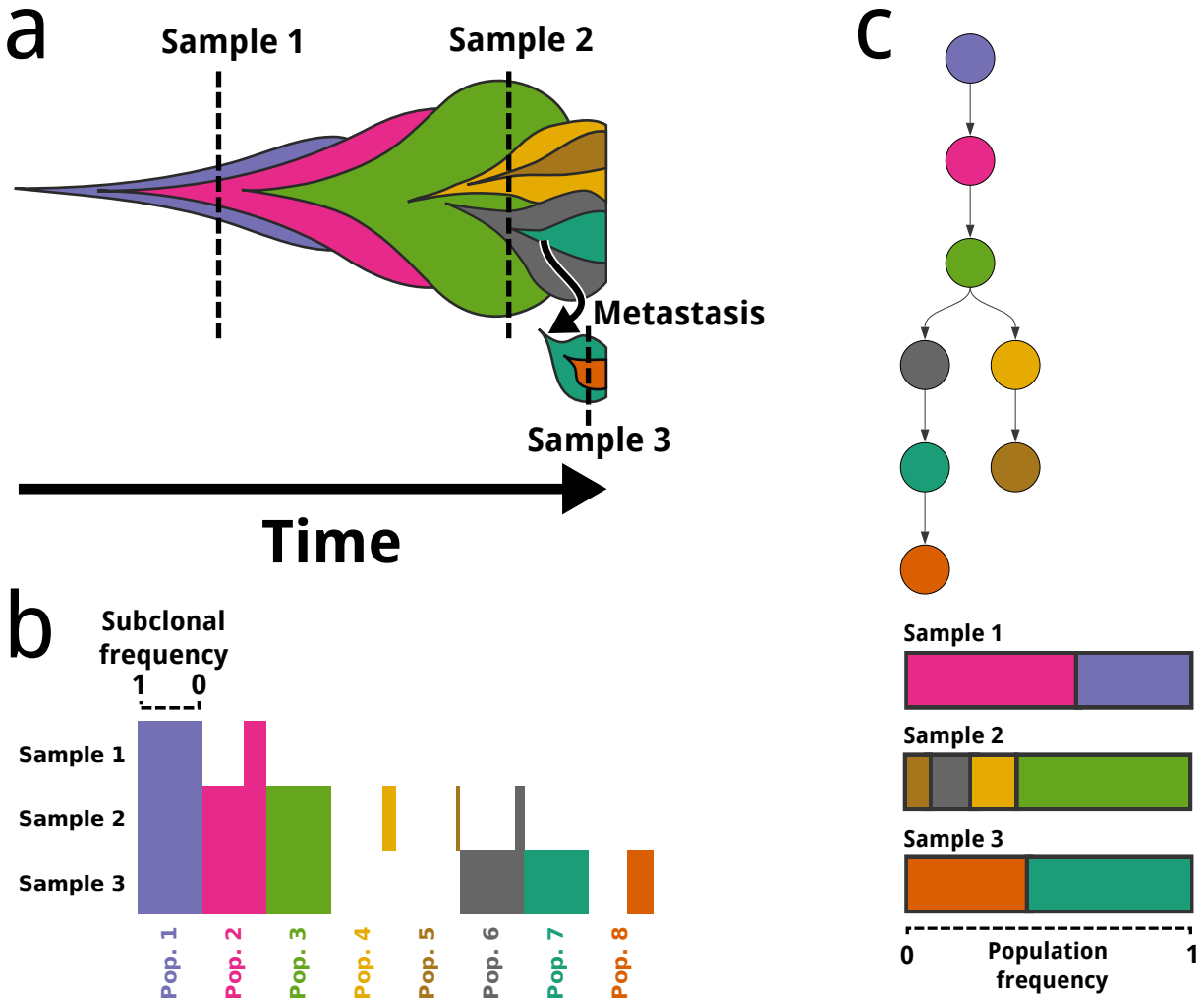
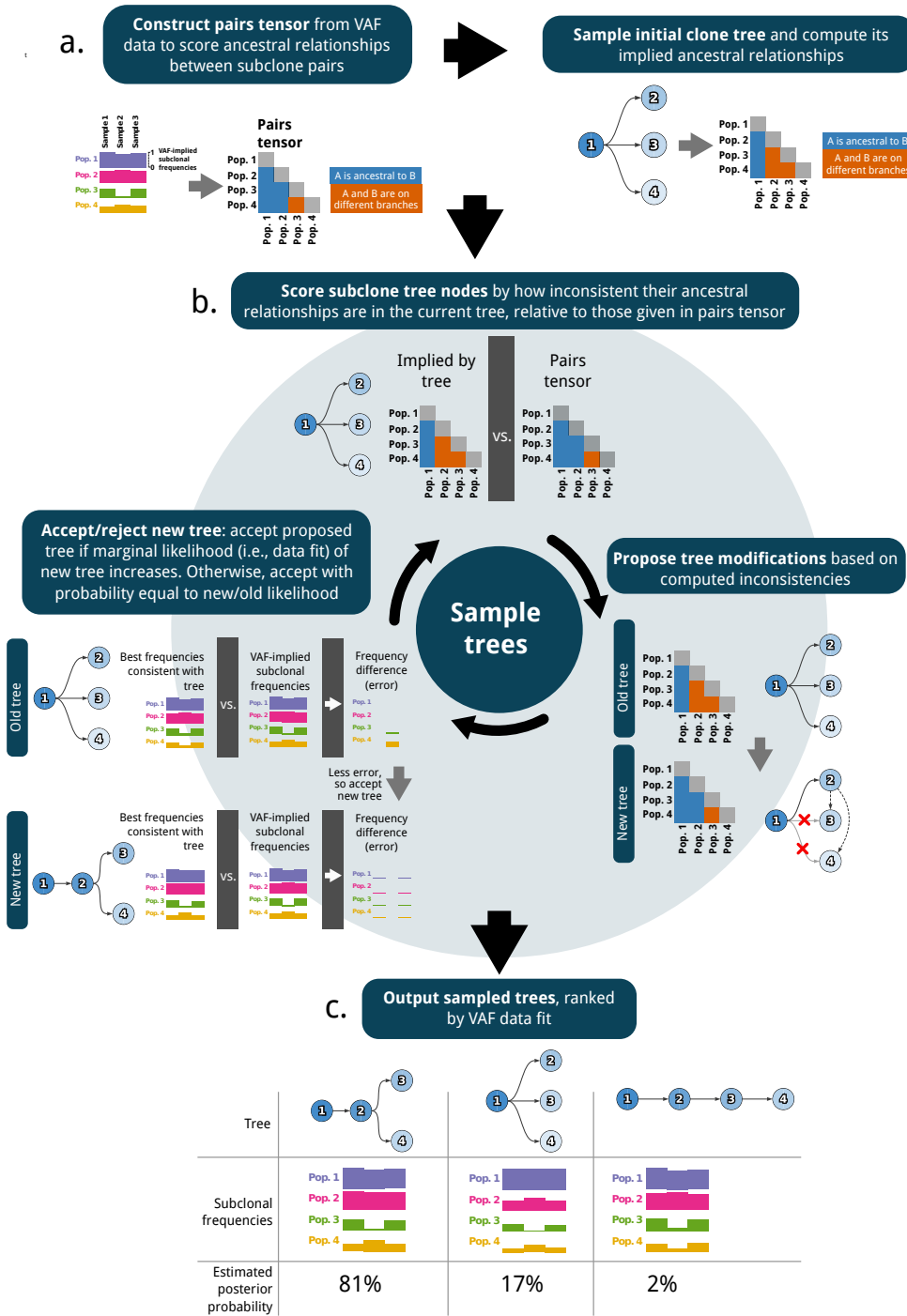
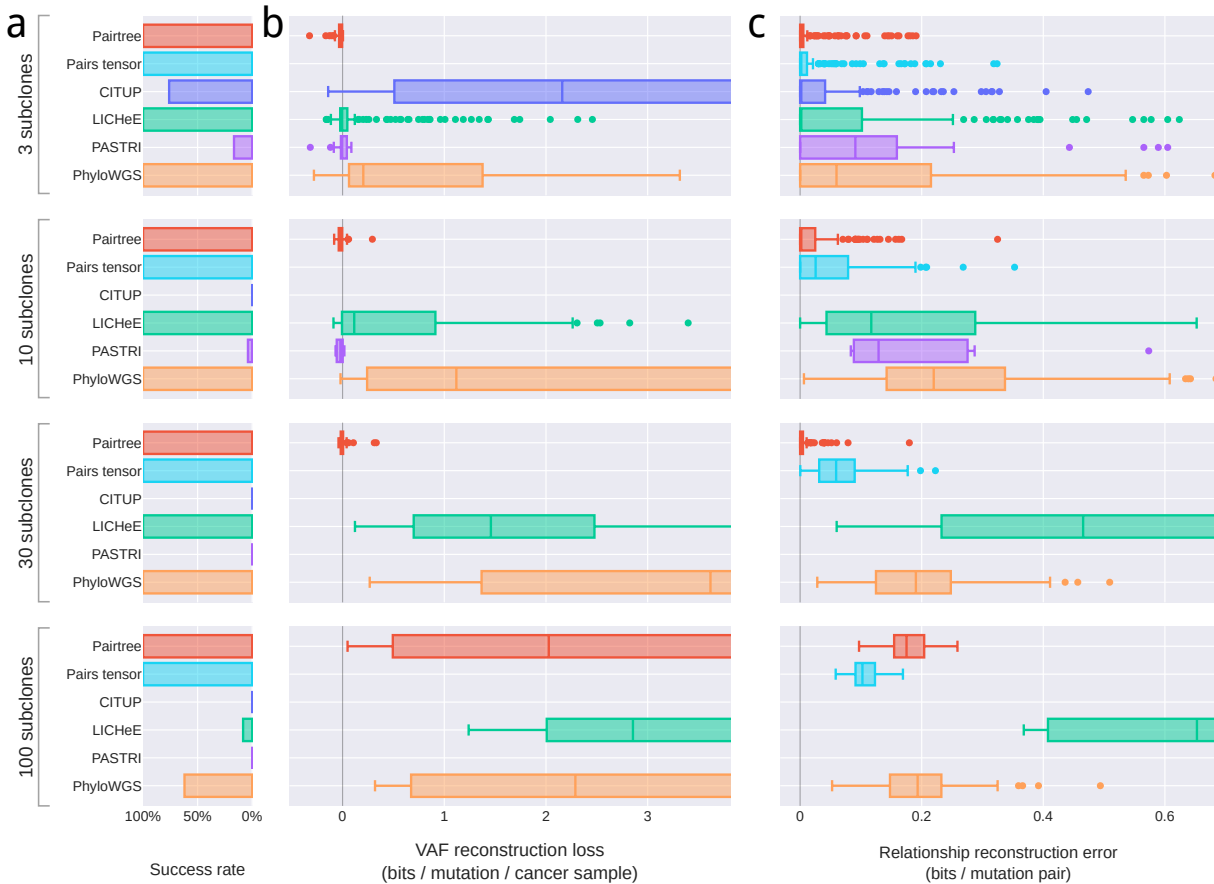


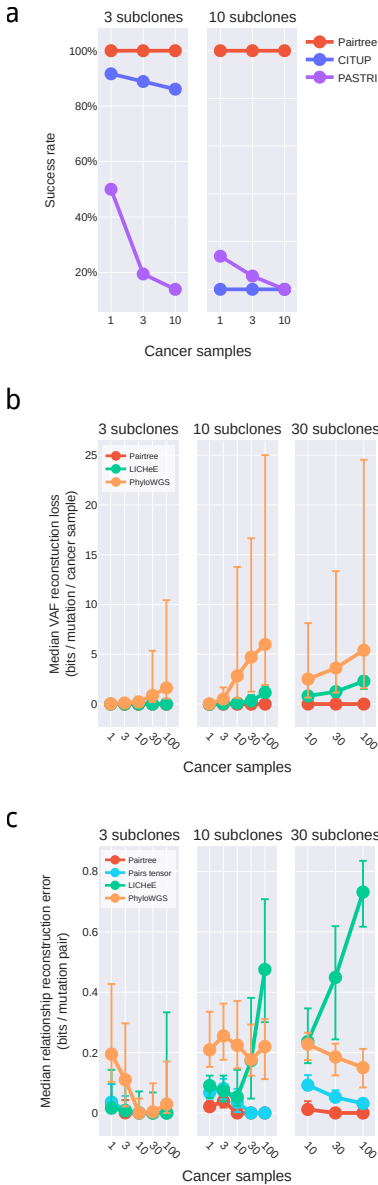
Figure 1: **Construction of clone trees from multiple cancer samples.** **a.** Schematic illustrates cancer development under the clonal evolution model. Each colour represents a genetically distinct subpopulation. Each subpopulation emerges within the mass of its parent. The leftmost point for a subpopulation denotes the cell that was its most recent common ancestor. Dashed vertical lines indicate when and where cancer samples were taken. The relative abundance of each subpopulation in a cancer sample, including any nested descendent subpopulations composing a subclone, is represented by the height of that subpopulation or subclone along the sample's dashed line. **b.** Horizontal bar plot showing idealized input to clone tree reconstruction algorithms. Bar length indicates the subclonal frequency of each subpopulation and its descendants (column) in each sequenced sample (row). The clonal evolution model asserts that a subpopulation's point mutations are inherited by its descendants. Consequently, mutation VAFs in DNA sequencing data provide estimates of subclonal frequencies, corresponding to the proportion of cells that originated from a subclonal population and its descendants. **c.** Clone tree representing the ancestry of subpopulations (top). Nodes indicate subpopulations. Arrows extend from each subpopulation to its direct descendants. Inferred frequencies of each subpopulation in each sample are based on the clone tree and mutation frequency data (bottom).



**Figure 2: The Pairtree algorithm.** **a.** Pairtree uses VAF data to compute the Pairs Tensor, which provides the probability of every possible ancestral relationship between subclone pairs (left). An initial clone tree is built using relationships scored by the Pairs Tensor. **b.** Pairtree samples trees using Markov Chain Monte Carlo. The method proposes tree modifications by identifying a subclone whose ancestral relationships in the current tree are assigned low probability by the Pairs Tensor (top), then ascertaining where that subclone can be moved within the tree to increase its ancestral relationship probabilities (bottom right). Proposed trees are then accepted or rejected based on their marginal likelihoods that reflect how well they fit the VAF data (bottom left). **c.** Sampled clone trees are returned along with posterior probability estimates proportional to the marginal likelihood of each tree. *VAF*, variant allele frequency.



**Figure 3: Benchmark performance on 576 simulated datasets.** Simulations are grouped by number of subclones (rows). **a.** Bar plots show each method’s success rate in the group. Successes are reconstruction problems for which the method produced at least one tree in 24 hours (wall-clock time) and did not crash. **b.** Boxplots show distributions of VAF reconstruction losses for a method on a problem group. Scores reflect only datasets where a method ran successfully. VAF reconstruction loss is the decrease in average, per-mutation log likelihood of VAF data using subclonal frequencies assigned by the method, when compared to the true frequencies used to generate the data. Negative loss indicates better VAF reconstructions than true trees, while high loss indicates inaccurate tree structures. Mid-lines in box plots indicate medians. Plots are truncated at four bits. Fig. S1 shows untruncated distributions. **c.** Boxplots show distributions of relationship reconstruction error in each group for each method’s successful runs. Relationship reconstruction error is measured as the average Jensen-Shannon divergence per subclone pair between the true distributions over pairwise relations, and empirical distributions computed from the trees output by a method. Errors can range between zero bits (perfect match) and one bit (complete mismatch), and are truncated at 0.7 bits. Fig. S2 shows untruncated distributions.



**Figure 4: Performance on simulated datasets as a function of number of subclones and cancer samples.** **a.** Method success rate. For CITUP and PASTRI, success rate depended on the number of subclones and/or cancer samples in datasets. Pairtree, LICHeE, and PhyloWGS succeeded on all datasets depicted. **b.** Median VAF reconstruction loss as a function of number of samples. For LICHeE and PhyloWGS, VAF loss increases with more cancer samples. **c.** Median relationship reconstruction error as a function of number of samples. LICHeE's error generally increased with more cancer samples, while other methods showed the opposite effect. Error bars represent the first and third quartiles in (b-c).



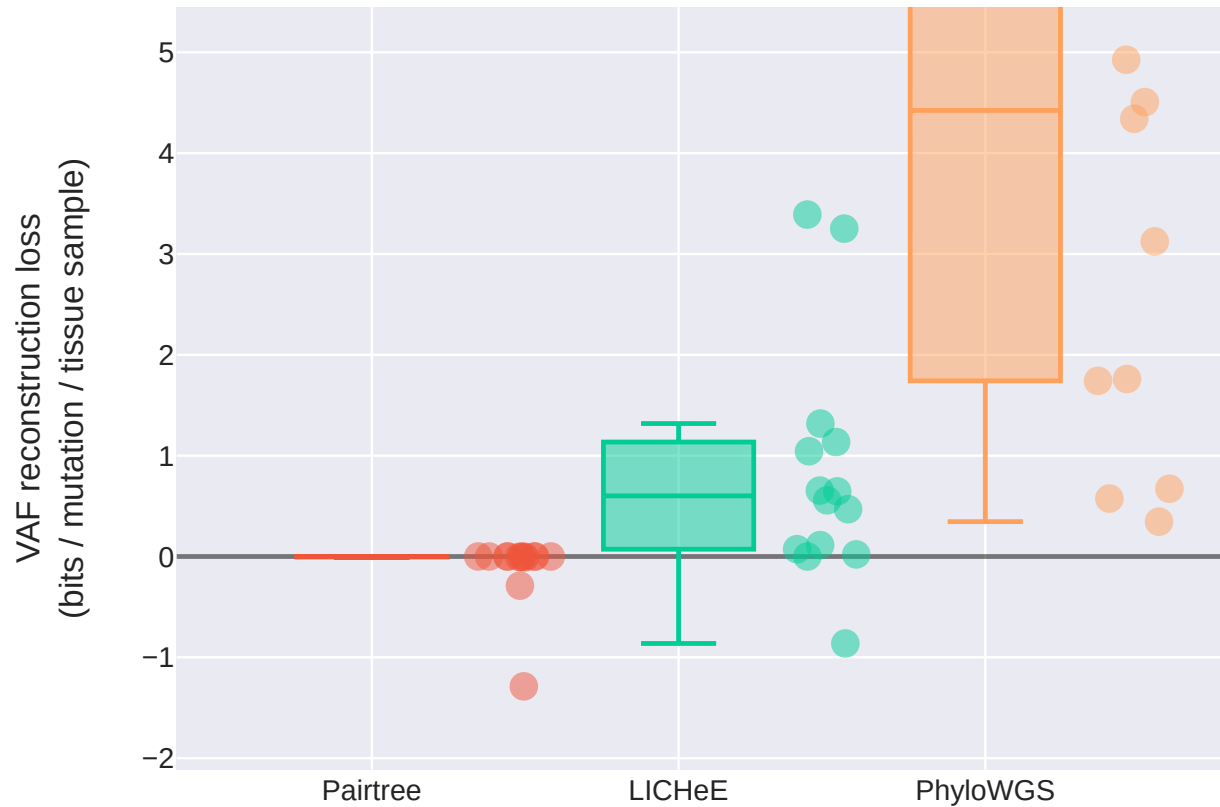


Figure 5: **VAF reconstruction loss for 14 B-ALL patient datasets.** The number of cancer samples for each dataset ranged from 13 to 90. Mid-lines in box plots indicate medians. CITUP and PASTRI each failed on 13 of 14 datasets and so are not shown. VAF reconstruction losses are reported as a negative log likelihood normalized to the number of mutations and cancer samples, relative to the MAP subclonal frequencies for expert-derived trees. Lower loss indicates better performance, while negative loss corresponds to performance better than human experts. Fig. S3 shows untruncated distributions.

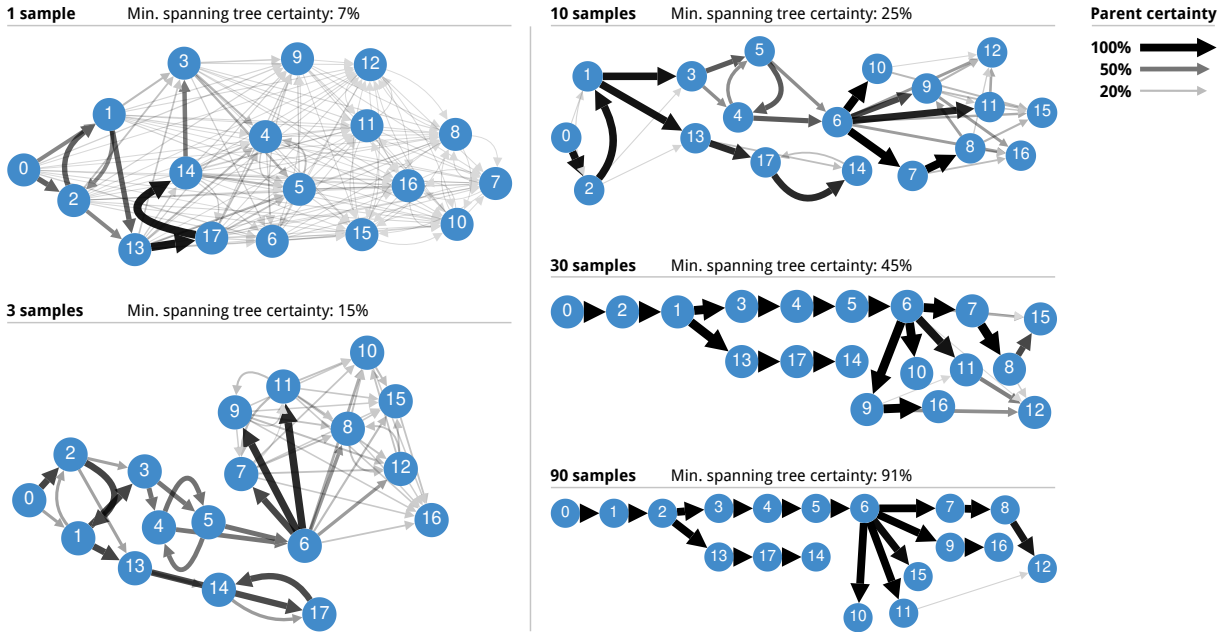


Figure 6: **Consensus graph visualization of posterior tree distributions.** These consensus graph visualizations are based on one of the 14 B-ALL cancers analyzed with Pairtree, for which 90 cancer samples were available. Consensus graphs are shown for variable numbers of samples, ranging from a single sample to all 90. All edges with less than 5% posterior certainty are hidden. The minimum spanning tree certainty indicates, if the graph is fully connected such that every subclone has at least one possible parent, how confident the least-certain single parent is.

## 341 6 Methods

### 342 6.1 Computing pairwise relations

#### 343 6.1.1 Establishing a probabilistic likelihood for pairwise relations

344 Let  $A$  and  $B$  represent two distinct mutations. We denote their observed read counts, encompassing both  
345 variant and reference reads, as  $x_A$  and  $x_B$ . Assuming both mutations obey the ISA, the pair  $(A, B)$  must  
346 fall in one of four pairwise relationships, denoted by  $M_{AB}$ .

- 347 1.  $M_{AB} = \textit{coincident}$ , meaning  $A$  and  $B$  are co-occurring. That is,  $A$  and  $B$  occur in precisely the  
348 same cell subpopulations, such that  $A$  is never present without  $B$  and vice versa. This reflects that  
349  $A$  and  $B$  occurred proximal to each other in evolutionary time, such that we cannot distinguish an  
350 intermediate subpopulation that occurred between them.
- 351 2.  $M_{AB} = \textit{ancestor}$ , meaning  $A$  is ancestral to  $B$ . That is,  $A$  occurred in a population ancestral to  
352  $B$ , such that some cells possess  $A$  without  $B$ , but no cell has  $B$  without  $A$ . This reflects that  $A$   
353 preceded  $B$ .
- 354 3.  $M_{AB} = \textit{descendent}$ , meaning  $B$  is ancestral to  $A$ . This mirrors relationship 2, reflecting  $B$  preceded  
355  $A$ .
- 356 4.  $M_{AB} = \textit{branched}$ , meaning  $A$  and  $B$  occurred on different branches of the clone tree, such that they  
357 never occur in the same set of cells. This relationship confers no information about the respective  
358 timing of  $A$  and  $B$ .

359 To the four possible relationships above, we add a fifth, termed the *garbage relation* and denoted  
360 by  $M_{AB} = \textit{garbage}$ . This represents mutation pairs with conflicting evidence for different relationships  
361 amongst the four already defined, providing a baseline against which the other four relationships can be  
362 compared. This catch-all category assumes that there is no consistent evolutionary relationship denoted  
363 by the subclonal frequencies of the two mutations across cancer samples, so it may represent ISA violations  
364 arising from the four-gamete test [38]. The garbage relation can also represent unreported CNAs that  
365 skew the relationship between VAF and subclonal frequency.

366 The likelihood of the pair's relationship is written as  $p(x_A, x_B | M_{AB})$ . First, we note that every cancer  
367 sample  $s$  can be considered independently of others, allowing us to factor the likelihood.

$$p(x_A, x_B | M_{AB}) = \prod_s p(x_{A_s}, x_{B_s} | M_{AB})$$

368 To compute the pairwise relationship likelihood for one cancer sample  $s$ , we integrate over the possible  
369 subclonal frequencies associated with the subclones that gave rise to mutations  $A$  and  $B$ , representing the  
370 proportions of cells in the cancer sample carrying the mutations. We indicate these subclonal frequencies  
371 as  $\phi_{A_s}$  and  $\phi_{B_s}$ .

$$p(x_{A_s}, x_{B_s} | M_{AB}) = \iint d\phi_{A_s} d\phi_{B_s} p(x_{A_s} | x_{B_s}, M_{AB}, \phi_{A_s}, \phi_{B_s}) p(x_{B_s} | M_{AB}, \phi_{A_s}, \phi_{B_s}) p(\phi_{A_s}, \phi_{B_s} | M_{AB})$$

372 As each mutation's likelihood is a function solely of its subclonal frequency, and is independent of  
373 both the other mutation and the pairwise relationship, we can simplify the integral.

$$p(x_{A_s}, x_{B_s} | M_{AB}) = \iint d\phi_{A_s} d\phi_{B_s} p(x_{A_s} | \phi_{A_s}) p(x_{B_s} | \phi_{B_s}) p(\phi_{A_s}, \phi_{B_s} | M_{AB}) \quad (1)$$

### 374 6.1.2 Defining a binomial observation model for read count data

375 We can now begin providing concrete definitions for each factor in the integral given in Eq. (1). For  
376 mutation  $j \in \{A, B\}$  from cancer sample  $s$ , whose observed read count data are represented by  $x_{j_s}$ , we  
377 define  $p(x_{j_s} | \phi_{j_s})$  using the following notation:

- 378 •  $\phi_{j_s}$ : subclonal frequency of subclone where  $j$  originated
- 379 •  $V_{j_s}$ : number of genomic reads of the  $j$  locus where the variant allele was observed
- 380 •  $R_{j_s}$ : number of genomic reads of the  $j$  locus where the reference allele was observed
- 381 •  $\omega_{j_s}$ : probability of observing the variant allele in a subclone containing  $j$ . Equivalently, this can be  
382 thought of as the probability of observing the variant allele in a cell bearing the  $j$  mutation. Thus,  
383 in a diploid cell,  $\omega_{j_s} = \frac{1}{2}$ .

384 Observe that  $\omega_{j_s}$  can be used to indicate changes in ploidy. For instance, a variant lying on either  
385 of the sex chromosomes in human males would have  $\omega_{j_s} = 1$ , since males possess only one copy of the  
386 X and Y chromosomes, so no wildtype allele would be present. Alternatively,  $\omega_{j_s}$  can indicate clonal

387 copy number changes, such that all cancer samples in a sample bore the same CNA. If, for instance, the  
388 founding cancerous subclone bearing a mutation underwent a duplication of the wildtype allele, then,  
389 once the mutation arose in a descendent subclone, every cell within that subclone would contribute two  
390 wildtype alleles and one variant allele. Thus, in this instance, we would have  $\omega_{js} = \frac{1}{3}$ . While this  
391 representation requires that the CNA be clonal, any SNVs affected by the CNA can be subclonal, and  
392 can in fact belong to different subclones.

393 Though this scheme can represent clonal CNAs, it cannot do so for subclonal CNAs. Fundamentally,  
394 the tree-building algorithm requires converting the observed  $\text{VAF}_{js} = \frac{V_{js}}{V_{js} + R_{js}}$  values into estimates of  
395 subclonal frequencies  $\hat{\phi}_{js} = \frac{\text{VAF}_{js}}{\omega_{js}} \approx \phi_{js}$ . If a subclonal CNA overlapping the mutation  $j$  occurs, different  
396 subclones will contribute different numbers of alleles to the cancer sample, implying this relationship is  
397 no longer valid. While the model could be extended to place subclonal CNA events on the clone tree  
398 and estimate how they change  $\hat{\phi}_{js}$ , the Pan-Cancer Analysis of Whole Genomes projects [39] reported  
399 frequent disagreement on allele-specific copy numbers among subclonal CNA-calling algorithms [1], and  
400 thus they discarded variants in regions affected by subclonal CNAs before constructing clone trees.

401 Using this notation, let the likelihood of observing  $V_{js}$  variant reads for mutation  $j$  in sample  $s$ , given  
402 a subclonal frequency  $\phi_{js}$ , be defined by the binomial. We have  $V_{js} + R_{js}$  observations of  $j$ 's genomic  
403 locus, and probability  $\omega_{js}\phi_{js}$  of observing a variant read, representing the proportion of alleles in the  
404 sample carrying the variant.

$$p(x_{js}|\phi_{js}) = \text{Binom}(V_{js}|V_{js} + R_{js}, \omega_{js}\phi_{js})$$

### 405 6.1.3 Defining constraints on subclonal frequencies imposed by pairwise relationships

406 To be fully realized, the likelihood Eq. (1) now requires only  $p(\phi_{As}, \phi_{Bs}|M_{AB})$  to be defined. We use  
407 this factor to represent whether  $\phi_{As}$  and  $\phi_{Bs}$  are consistent with the relationship  $M_{AB}$ . For the ances-  
408 tor, descendent, and branched relationships, the subclonal frequencies  $\phi_{As}$  and  $\phi_{Bs}$  dictate whether a  
409 relationship is possible.

$$\begin{aligned}
 p(\phi_{As}, \phi_{Bs} | M_{AB} = \textit{ancestor}) &= \begin{cases} C & \text{iff } 1 \geq \phi_{As} \geq \phi_{Bs} \geq 0 \\ 0 & \text{otherwise} \end{cases} \\
 p(\phi_{As}, \phi_{Bs} | M_{AB} = \textit{descendent}) &= \begin{cases} C & \text{iff } 0 \leq \phi_{As} \leq \phi_{Bs} \leq 1 \\ 0 & \text{otherwise} \end{cases} \\
 p(\phi_{As}, \phi_{Bs} | M_{AB} = \textit{branched}) &= \begin{cases} C & \text{iff } \phi_{As} \geq 0 \wedge \phi_{Bs} \geq 0 \wedge \phi_{As} + \phi_{Bs} \leq 1 \\ 0 & \text{otherwise} \end{cases}
 \end{aligned}$$

410 The subclonal frequencies  $\phi_{As}$  and  $\phi_{Bs}$  may each take values on the  $[0, 1]$  interval. Thus,  $p(\phi_{As}, \phi_{Bs} | M_{AB})$   
 411 for  $M_{AB} \in \{\textit{ancestor}, \textit{descendent}, \textit{branched}\}$  are each non-zero only inside a right triangle lying within  
 412 the unit square on the Cartesian plane with corners at  $(\phi_{As}, \phi_{Bs}) \in \{(0, 0), (0, 1), (1, 0), (1, 1)\}$ . The  
 413 location of the triangle within the unit square differs for each of the three  $M_{AB}$  relationships, but all  
 414 have an area of  $\frac{1}{2}$ . Consequently, to ensure  $\iint d\phi_{As} d\phi_{Bs} p(\phi_{As}, \phi_{Bs} | M_{AB}) = 1$ , we set  $C = \frac{1}{\frac{1}{2}} = 2$ . Thus,  
 415  $p(\phi_{As}, \phi_{Bs} | M_{AB}) = C = 2$  when  $\phi_{As}$  and  $\phi_{Bs}$  are consistent with  $M_{AB}$ , and zero otherwise.

416 We must still define the remaining two relationships  $M_{AB} \in \{\textit{coincident}, \textit{garbage}\}$ . The garbage rela-  
 417 tionship permits all combinations of  $\phi_{As}$  and  $\phi_{Bs}$  lying within the unit square, such that  $p(\phi_{As}, \phi_{Bs} | M_{AB} =$   
 418  $\textit{garbage}) = 1$ . Consequently, unlike the previous three relationships, the garbage relationship imposes no  
 419 constraints on  $\phi_{As}$  and  $\phi_{Bs}$  relative to each other.

$$p(\phi_{As}, \phi_{Bs} | M_{AB} = \textit{garbage}) = \begin{cases} 1 & \text{iff } \{\phi_{As}, \phi_{Bs}\} \subset \{x | 0 \leq x \leq 1\} \\ 0 & \text{otherwise} \end{cases}$$

420 The garbage relationship serves to establish a baseline against which evidence for the non-garbage  
 421 relationships can be evaluated. Observe that, in Eq. (1),  $p(x_{As} | \phi_{As})p(x_{Bs} | \phi_{Bs})$  is integrated over the  
 422 unit square when  $M_{AB} = \textit{garbage}$ . Conversely, when  $M_{AB} \in \{\textit{ancestor}, \textit{descendent}, \textit{branched}\}$ , we inte-  
 423 grate  $p(x_{As} | \phi_{As})p(x_{Bs} | \phi_{Bs})$  over a triangle covering half the square. Consequently,  $p(x_{AS}, x_{BS} | M_{AB} =$   
 424  $\textit{garbage}) \leq \frac{1}{2} p(x_{AS}, x_{BS} | M_{AB} \in \{\textit{ancestor}, \textit{descendent}, \textit{branched}\})$ . This arises because  $p(\phi_{As}, \phi_{Bs} | M_{AB}) =$   
 425 2 for subclonal frequencies consistent with  $M_{AB} \in \{\textit{ancestor}, \textit{descendent}, \textit{branched}\}$ , while  $p(\phi_{As}, \phi_{Bs} | M_{AB}) =$   
 426 1 for subclonal frequencies consistent with  $M_{AB} = \textit{garbage}$ . When the read counts for the mutations  $A$   
 427 and  $B$  clearly permit one of the three non-garbage relationships, most of the probability mass of the two

428 associated binomials will reside within the simplex permitted by the relationship, and so the evidence for  
429 the non-garbage relationship will be nearly double that of the evidence for garbage. Conversely, when  
430 the read counts push most of the binomial mass outside the permitted simplex, the non-garbage evidence  
431 will be substantially lower than the baseline provided by garbage.

432 By considering accumulated evidence across many cancer samples, the garbage model's utility becomes  
433 clear. If, across many cancer samples for a mutation pair, the evidence for one non-garbage relationship  
434 is consistently favoured over others, then that relationship will emerge as the most likely when the  
435 evidence is considered collectively across samples. However, if different cancer samples favour different  
436 relationship types, the steady accumulation of the baseline garbage evidence could, in concert, be more  
437 than the evidence for any of the other three relations, meaning garbage would be declared as the most  
438 likely relationship for the mutation pair. Mutations that make up many pairs with high garbage evidence  
439 are best excluded from clone tree construction, as such mutations likely suffered from uncalled CNAs,  
440 violations of the ISA, or highly erroneous read count data.

441 The only undefined relationship remaining is  $M_{AB} = \textit{coincident}$ . As the coincident relationship  
442 dictates that two mutations arose from the same subclone, and so share the same subclonal frequency,  
443 the corresponding constraint is defined thusly:

$$p(\phi_{As}, \phi_{Bs} | M_{AB} = \textit{coincident}) = \begin{cases} 1 & \text{iff } 0 \leq \phi_{As} = \phi_{Bs} \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

#### 444 6.1.4 Efficiently computing evidence for ancestral, descendent, and branched pairwise re- 445 lationships

446 We now consider how to compute the pairwise likelihood given in Eq. (1) for  $M_{AB} \in \{\textit{ancestor}, \textit{descendent}, \textit{branched}\}$ .

$$p(x_{As}, x_{Bs} | M_{AB}) = \iint d\phi_{As} d\phi_{Bs} p(x_{As} | \phi_{As}) p(x_{Bs} | \phi_{Bs}) p(\phi_{As}, \phi_{Bs} | M_{AB})$$

447 Observe that we can rearrange the integral to move the factor corresponding to the mutation  $A$   
448 observations outside the inner integral.

$$p(x_{A_s}, x_{B_s} | M_{AB}) = \int d\phi_{A_s} p(x_{A_s} | \phi_{A_s}) \int d\phi_{B_s} p(x_{B_s} | \phi_{B_s}) p(\phi_{A_s}, \phi_{B_s} | M_{AB})$$

449 Now, because  $p(\phi_{A_s}, \phi_{B_s} | M_{AB})$  is piecewise-constant when  $M_{AB} \in \{\textit{ancestor}, \textit{descendent}, \textit{branched}\}$ ,  
 450 we can, for these relationships, impose this factor's effect by changing the integration limits. Let  
 451  $L(\phi_{A_s}, M_{AB})$  and  $U(\phi_{A_s}, M_{AB})$  represent functions whose outputs are the lower and upper integra-  
 452 tion limits, respectively, for the inner integral whose differential is  $d\phi_{B_s}$ , as a function of  $\phi_{A_s}$  and the  
 453 relationship  $M_{AB}$ . These functions are defined thusly:

$$(L(\phi_{A_s}, M_{AB}), U(\phi_{A_s}, M_{AB})) = \begin{cases} (0, \phi_{A_s}) & \text{if } M_{AB} = \textit{ancestral} \\ (\phi_{A_s}, 1) & \text{if } M_{AB} = \textit{descendent} \\ (0, 1 - \phi_{A_s}) & \text{if } M_{AB} = \textit{branched} \end{cases}$$

454 By writing the inner integral using these integration limits, and limiting the outer integral to the  $[0, 1]$   
 455 interval permitted for  $\phi_{A_s}$ , the factor  $p(\phi_{A_s}, \phi_{B_s} | M_{AB})$  can be replaced by 2, as it is constant over the  
 456 interval of integration.

$$p(x_{A_s}, x_{B_s} | M_{AB}) = \int_0^1 d\phi_{A_s} 2p(x_{A_s} | \phi_{A_s}) \int_{L(\phi_{A_s}, M_{AB})}^{U(\phi_{A_s}, M_{AB})} d\phi_{B_s} p(x_{B_s} | \phi_{B_s})$$

457 To render the inner integral more computationally convenient, rather than integrate over  $\phi_{B_s}$ , we  
 458 would prefer to integrate over  $q_{B_s} \equiv \omega_{B_s} \phi_{B_s}$ . Thus, we will integrate by substitution, using  $\frac{dq_{B_s}}{d\phi_{B_s}} = \omega_{B_s}$ .

$$\begin{aligned} \int d\phi_{B_s} p(x_{B_s} | \phi_{B_s}) &= \int d\phi_{B_s} \text{Binom}(V_{B_s} | V_{B_s} + R_{B_s}, \omega_{B_s} \phi_{B_s}) \\ &= \frac{1}{\omega_{B_s}} \int dq_{B_s} \text{Binom}(V_{B_s} | V_{B_s} + R_{B_s}, q_{B_s}) \end{aligned} \quad (2)$$

459 Observe that the inner integral is now simply integrating the binomial PMF over its parameter  $q_{B_s}$ .  
 460 To compute this integral, we rely on the following equivalence between this integral and the incomplete  
 461 beta function  $\beta$ :



$$\begin{aligned} \int_0^b dp \text{Binom}(k|N, p) &= \int_0^b dp \binom{N}{k} p^k (1-p)^{N-k} \\ &= \binom{N}{k} \int_0^b dp p^k (1-p)^{N-k} \\ &= \binom{N}{k} \beta(b|k+1, N-k+1) \end{aligned}$$

462 Now we can compute the integral over an arbitrary limit by the fundamental theorem of calculus.

$$\int_a^b dp \text{Binom}(k|N, p) = \binom{N}{k} \left[ \beta(b|k+1, N-k+1) - \beta(a|k+1, N-k+1) \right] \quad (3)$$

463 Finally, we combine the above results, allowing us to compute the pairwise relationship likelihood  
464 when  $M_{AB} \in \{\textit{ancestor}, \textit{descendent}, \textit{branched}\}$  as a one-dimensional integral.

$$\begin{aligned} p(x_{As}, x_{Bs} | M_{AB}) &= \frac{2}{\omega_{Bs}} \binom{V_{Bs} + R_{Bs}}{V_{Bs}} \int_0^1 d\phi_{As} \text{Binom}(V_{As} | V_{As} + R_{As}, \omega_{As} \phi_{As}) \left[ \right. \\ &\quad \beta(\omega_{Bs} U(\phi_{As}, M_{AB}) | V_{Bs} + 1, R_{Bs} + 1) \\ &\quad \left. - \beta(\omega_{Bs} L(\phi_{As}, M_{AB}) | V_{Bs} + 1, R_{Bs} + 1) \right] \quad (4) \end{aligned}$$

465 To compute this numerically, we use the one-dimensional quadrature algorithm from `scipy.integrate.quad`.

### 466 6.1.5 Efficiently computing evidence for garbage and coincident pairwise relationships

467 We now examine how to compute the pairwise relationship likelihood for  $M_{AB} = \textit{garbage}$  using the  
468 general likelihood given in Eq. (1). First, observe that we are integrating over  $\phi_{As} \in [0, 1]$  and  $\phi_{Bs} \in [0, 1]$ ,  
469 meaning there is no constraint placed on  $\phi_{Bs}$  by  $\phi_{As}$ . By removing the dependence of  $\phi_{Bs}$  on  $\phi_{As}$ , the  
470 likelihood can be broken into the product of two one-dimensional integrals, each taken over the interval  
471  $[0, 1]$ . Then, by drawing on results Eq. (2) and Eq. (3), we can compute an analytic solution to each  
472 integral.

$$\begin{aligned}
 p(x_{A_s}, x_{B_s} | M_{AB} = \textit{garbage}) &= \iint d\phi_{A_s} d\phi_{B_s} p(x_{A_s} | \phi_{A_s}) p(x_{B_s} | \phi_{B_s}) p(\phi_{A_s}, \phi_{B_s} | M_{AB}) \\
 &= \left[ \int_0^1 d\phi_{A_s} p(x_{A_s} | \phi_{A_s}) \right] \left[ \int_0^1 d\phi_{B_s} p(x_{B_s} | \phi_{B_s}) \right] \\
 &= \frac{1}{\omega_{A_s} \omega_{B_s}} \beta(\omega_{A_s} | V_{A_s} + 1, R_{A_s} + 1) \beta(\omega_{B_s} | V_{B_s} + 1, R_{B_s} + 1) \quad (5)
 \end{aligned}$$

473 Finally, we compute the likelihood for  $M_{AB} = \textit{coincident}$ . As our coincident constraint requires  
 474  $\phi_{A_s} = \phi_{B_s}$ , we are integrating along the diagonal line  $\phi_{A_s} = \phi_{B_s}$  that cuts through the unit square  
 475 formed by  $\phi_{A_s} \in [0, 1]$  and  $\phi_{B_s} \in [0, 1]$ . This can be evaluated as a line integral along the curve  
 476  $r(\phi) \equiv \langle \phi, \phi \rangle$  for  $\phi \in [0, 1]$ , with the Euclidean norm  $\|r'(\phi)\| = \sqrt{2}$ .

$$\begin{aligned}
 p(x_{A_s}, x_{B_s} | M_{AB} = \textit{coincident}) &= \iint d\phi_{A_s} d\phi_{B_s} p(x_{A_s} | \phi_{A_s}) p(x_{B_s} | \phi_{B_s}) p(\phi_{A_s}, \phi_{B_s} | M_{AB}) \\
 &= \int_0^1 d\phi p(x_{A_s} | \phi) p(x_{B_s} | \phi) \|r'(\phi)\| \\
 &= \sqrt{2} \int_0^1 d\phi \text{Binom}(V_{A_s} | V_{A_s} + R_{A_s}, \omega_{A_s} \phi) \text{Binom}(V_{B_s} | V_{B_s} + R_{B_s}, \omega_{B_s} \phi) \quad (6)
 \end{aligned}$$

477 As with the ancestral, descendent, and branched relationships, we use the one-dimensional quadrature  
 478 algorithm from `scipy.integrate.quad` to compute this.

### 479 6.1.6 Computing the posterior probability for pairwise relationships

480 In Eq. (4), Eq. (5), and Eq. (6), we established how to compute the evidence for each of the five possible  
 481 relations between mutation pairs, which takes the general form  $p(x_A, x_B | M_{AB})$ . By combining these  
 482 evidences with a prior probability  $p(M_{AB})$  over relationships for mutation pair  $(A, B)$ , we can compute  
 483 the posterior probability  $p(M_{AB} | x_A, x_B)$  of each relationship.

$$p(M_{AB} | x_A, x_B) = \frac{p(x_A, x_B | M_{AB}) p(M_{AB})}{\sum_{M'} p(x_A, x_B | M') p(M')} \quad (7)$$

484 As we discuss in Section 6.2.8, we assume that, when `Pairtree` is run, mutations have already been  
 485 clustered into subpopulations and “garbage” mutations have already been discarded. Consequently, we

486 are computing pairwise relations between groups of mutations comprising subclones, and so we assign  
487 zero prior mass to the *coincident* and *garbage* relationships, ensuring these relationships also have zero  
488 posterior mass. The other three relationships are assigned the same prior probability, as we have no  
489 reason to believe one is more likely than the others.

$$p(M_{AB}) = \begin{cases} \frac{1}{3} & \text{if } M_{AB} \in \{\textit{ancestor}, \textit{descendent}, \textit{branched}\} \\ 0 & \text{if } M_{AB} \in \{\textit{coincident}, \textit{garbage}\} \end{cases}$$

## 490 6.2 Performing tree search

### 491 6.2.1 Representing cancer evolutionary histories with trees

492 Most clone tree reconstruction algorithms group mutations into subclones, with mutations that share  
493 the same subclonal frequency across cancer samples placed together. While thousands of mutations  
494 are typically observed using whole-genome sequencing, the mutations can typically be grouped into a  
495 much smaller number of subclones, simplifying the cancer’s evolutionary history. This grouping is valid  
496 because, as a cell population expands within a cancer, the frequencies of all mutations shared by cells in  
497 that population will increase in lockstep. Although Pairtree does not explicitly require that mutations  
498 be grouped into subclones, it can take these groupings as input. In this case, it replaces each mutation  
499 group with a single mutation, termed a *super-variant*, that represents the subclone.

500 When provided with  $K$  mutation clusters as input, each consisting of one or more mutations, Pairtree  
501 will produce a distribution over trees with  $K + 1$  nodes. Node 0 corresponds to the non-cancerous cell  
502 lineage that gave rise to the cancer, while node  $k \in \{1, 2, \dots, K\}$  corresponds to the subclone associated  
503 with mutation cluster  $k$ . Node 0 always serves as the tree root, representing that the patient’s cancer  
504 developed from non-cancerous cells, and thus has no assigned mutations and a subclonal frequency of  
505  $\phi_{0s} = 1$  in every cancer sample  $s$ .

506 An edge from node  $A$  to node  $B$  indicates that subclone  $B$  evolved from subclone  $A$ , acquiring the  
507 mutations associated with cluster  $B$  while also inheriting all mutations present in  $A$  and  $A$ ’s ancestral  
508 nodes. The children of node 0 are termed the *clonal cancer populations*. Typically, there is only one  
509 clonal cancer population, but the algorithm allows multiple such populations when the data imply them.  
510 Multiple clonal cancer populations indicate that multiple cancers developed independently in the patient,  
511 such that they shared no common cancerous ancestor.

512 An edge from node  $A$  to node  $B$  means that, at the resolution permitted by the data, we cannot discern

513 any intermediate cell subpopulations that occurred between these two evolutionary points. Nevertheless,  
514 such subpopulations may well have existed in the cancer.

## 515 6.2.2 Tree likelihood

516 To describe the tree likelihood, we develop the following notation:

- 517 •  $K$ : number of cancerous subpopulations (and mutation clusters), with individual populations in-  
518 dexed as  $k \in \{1, 2, \dots, K\}$
- 519 •  $S$ : number of cancer samples, with individual samples indexed as  $s \in \{1, 2, \dots, S\}$
- 520 •  $M_k$ : set of mutations associated with subclone  $k$ . Note this is distinct from the  $M_{AB}$  notation used  
521 in Section 6.1 to denote the pairwise relationship between mutations.
- 522 •  $V_{ms}$ : observed variant read count for mutation  $m$  in cancer sample  $s$
- 523 •  $R_{ms}$ : observed reference read count for mutation  $m$  in cancer sample  $s$
- 524 •  $\omega_{ms}$ : probability of observing a variant read at mutation  $m$ 's locus within a subclone possessing  $m$ ,  
525 in cancer sample  $s$
- 526 •  $\phi_{ks}$ : subclonal frequency of subclone  $k$  in cancer sample  $s$
- 527 •  $\Phi$ : set of  $\phi_{ks}$  values for all  $K$  and  $S$

528 The data  $x$  consists of the set of all  $V_{ms}$ ,  $R_{ms}$ , and  $\omega_{ms}$  mutation values, as well as the  $M_k$  clustering of  
529 those mutations into subclones. Given the tree  $t$ , consisting of a tree structure and associated subclonal  
530 frequencies  $\Phi = \{\phi_{ks}\}$ , Pairtree uses the likelihood  $p(x|t, \Phi)$  to score the tree. We describe how to  
531 compute the subclonal frequencies in Section 6.3. Below we use  $x_{ks}$  to represent all data in sample  $s$  for  
532 the mutations associated with subclone  $k$ , while  $x_{ms}$  refers to the data for an individual mutation  $m$ .

$$\begin{aligned}
 p(x|t, \Phi) &= \prod_{k \in \{1, 2, \dots, K\}} \prod_{s \in \{1, 2, \dots, S\}} p(x_{ks}|t, \Phi) \\
 &= \prod_{k \in \{1, 2, \dots, K\}} \prod_{m \in M_k} \prod_{s \in \{1, 2, \dots, S\}} p(x_{ms}|\phi_{ks}) \\
 &= \prod_{k \in \{1, 2, \dots, K\}} \prod_{m \in M_k} \prod_{s \in \{1, 2, \dots, S\}} \text{Binom}(V_{ms}|V_{ms} + R_{ms}, \omega_{ms}\phi_{ks})
 \end{aligned} \tag{8}$$

533 The likelihood Eq. (8) demonstrates that tree structure is not explicitly considered in the tree like-  
 534 lihood. Instead, we assess tree likelihood by how well the observed mutation data are fit by the tree-  
 535 constrained subclonal frequencies accompanying the tree. Typically, we obtain a tree’s subclonal frequen-  
 536 cies by making a maximum a posteriori (MAP) estimate, as described in Section 6.3.

537 Though Eq. (8) is ultimately the likelihood used by Pairtree for tree search, examining another  
 538 perspective can help us understand what this likelihood represents. If we wished to directly assess the  
 539 quality of a tree structure independent of its subclonal frequencies, thereby obtaining the likelihood  
 540  $p(x|t)$  rather than  $p(x|t, \Phi)$ , we would integrate over the range of tree-constrained subclonal frequencies  
 541 permitted by the tree structure.

$$\begin{aligned}
 p(x|t) &= \prod_{k \in \{1, 2, \dots, K\}} \prod_{s \in \{1, 2, \dots, S\}} p(x_{ks}|t) \\
 &= \prod_{k \in \{1, 2, \dots, K\}} \prod_{m \in M_k} \prod_{s \in \{1, 2, \dots, S\}} p(x_{ms}|t) \\
 &= \prod_{k \in \{1, 2, \dots, K\}} \prod_{m \in M_k} \prod_{s \in \{1, 2, \dots, S\}} \int d\Phi p(x_{ms}, \Phi|t) \\
 &= \prod_{k \in \{1, 2, \dots, K\}} \prod_{m \in M_k} \prod_{s \in \{1, 2, \dots, S\}} \int d\Phi p(x_{ms}|\phi_{ks}) p(\Phi|t) \tag{9} \\
 &\approx \prod_{k \in \{1, 2, \dots, K\}} \prod_{m \in M_k} \prod_{s \in \{1, 2, \dots, S\}} p(x_{ms}|\phi_{ks}) \tag{10}
 \end{aligned}$$

542 In Eq. (9), the factor  $p(\Phi|t)$  is an indicator function representing whether the set of subclonal fre-  
 543 quencies  $\Phi$  obeys the constraints imposed by the tree structure  $t$ :

- 544 1. All subclonal frequencies exist within the unit interval, such that  $\phi_{ks} \in [0, 1]$  for all  $k$  and  $s$ .
- 545 2. The non-cancerous node 0 is an ancestor of all subpopulations, such that  $\phi_{ks} = 1$  for all  $k$  and  $s$ .
- 546 3. Let  $C(k)$  represent the children of population  $k$  in the tree. The subclonal frequency for  $k$  must be  
 547 at least as great as the sum of its childrens’ frequencies, such that  $\phi_{ks} \geq \sum_{c \in C(k)} \phi_{cs}$ .

548 To obtain Eq. (9), we assume that only a narrow range of subclonal frequencies are permitted by  
 549 the tree structure, and so we can use the MAP subclonal frequencies to approximate the integral and  
 550 obtain Eq. (10), which is the likelihood function that Pairtree uses, as per Eq. (8). Consequently, we  
 551 use Pairtree’s likelihood  $p(x|t, \Phi)$  of the tree  $t$  and subclonal frequencies  $\Phi$  as an approximation of the  
 552 marginal likelihood of the tree  $p(x|t)$ .

553 As an aside, note that a set of subclonal frequencies  $\Phi$  obeying the three constraints enumerated  
554 above may be consistent with multiple tree structures—i.e., we may have  $p(\Phi|t) \neq 0$  for a fixed  $\Phi$  and  
555 different tree structures  $t$ . This shows how ambiguity may exist: a tree’s subclonal frequencies may  
556 permit multiple possible tree structures, all of which would be assigned the same likelihood. Each cancer  
557 sample’s subclonal frequencies typically impose additional constraints on possible tree structures, reducing  
558 this ambiguity.

### 559 6.2.3 Using Metropolis-Hastings to search for trees

560 Pairtree uses the Metropolis-Hastings algorithm [32], a Markov Chain Monte Carlo method, to search for  
561 trees that best fit the observed read count data  $x$ . For notational convenience, our references to a tree  
562  $t$  should be understood to implicitly include a set of subclonal frequencies  $\Phi$  that have been computed  
563 for  $t$ , such that the likelihood denoted  $p(x|t)$  actually represents the likelihood  $p(x|t, \Phi)$  described in  
564 Section 6.2.2.

565 According to the Metropolis-Hastings algorithm, to obtain samples from the posterior distribution  
566 over trees  $p(t|x)$ , we must modify an existing tree  $t$  to create a new proposal tree  $t'$ . The  $t'$  tree is  
567 accepted or rejected as a valid sample from the posterior according to how its likelihood  $p(x|t')$  compares  
568 to the existing tree’s  $p(x|t)$ , as well as the probabilities  $p(t \rightarrow t')$  of transitioning from the  $t$  tree to the  
569  $t'$  tree, and  $p(t' \rightarrow t)$  of returning from  $t'$  to  $t$ . By Metropolis-Hastings, we assume that, given enough  
570 samples generated in this manner, we are eventually obtaining samples from the posterior distribution  
571 over trees  $p(t|x) = \frac{p(x|t)p(t)}{p(x)} = \frac{p(x|t)p(t)}{\sum_{t'} p(x|t')p(t')}$ . To establish our tree prior  $p(t)$ , we denote the number of  
572 possible tree topologies for  $K$  subclones as  $T(K)$ , which is a large but finite number that is exponential  
573 as a function of  $K$  [20]. Thus, we define our tree prior as a uniform distribution  $p(t) \equiv \frac{1}{T(K)}$ , as we have  
574 no reason to prefer one tree structure to another a priori. Consequently, in computing the posterior ratio  
575  $\frac{p(t'|x)}{p(t|x)}$  required for Metropolis-Hastings, all factors except the likelihoods  $p(x|t)$  and  $p(x|t')$  cancel.

576 Pairtree can run multiple MCMC chains in parallel, with each starting from a different initialization  
577 (Section 6.2.7). By default, Pairtree runs a total of  $C$  chains, with  $C$  set to the number of CPU cores  
578 present on the system by default, and  $P = C$  executing in parallel. Both  $P$  and  $C$  can be customized  
579 by the user. From each chain,  $S = 3000$  samples are drawn by default. The first  $B \in [0, 1]$  proportion  
580 of trees are assumed to be early attempts by the sampling procedure to migrate toward high-probability  
581 regions of tree space, and so are discarded as burn-in samples because they are assumed to poorly reflect  
582 the true posterior. By default, we set  $B = \frac{1}{3}$ . To reduce correlation between successive samples, Pairtree  
583 supports thinning, by which only a fraction  $T \in [0, 1]$  of non-burn-in samples are retained. By default,

584 Pairtree does not thin samples, so  $T = 1$ . Pairtree uses  $T$  to calculate a parameter  $N = \text{round}(\frac{1}{T})$ , such  
585 that the algorithm records every  $N$ th sample. Thus, the actual number of trees recorded from a chain is  
586  $L = 1 + \lfloor \frac{S-1}{N} \rfloor$ . Only after thinning the chain are the burn-in samples discarded, resulting in  $\text{round}(BL)$   
587 trees being returned as posterior samples from the chain. The  $C$ ,  $P$ ,  $S$ ,  $B$ , and  $T$  parameters can all be  
588 changed by the user.

589 Once all chains finish sampling, Pairtree combines their results to provide an estimate of the posterior  
590 tree distribution. Given the uniform tree prior  $p(t)$ , the posterior tree probability simplifies to  $p(t|x) =$   
591  $\frac{p(x|t)}{\sum_{t'} p(x|t')}$ . If the same tree  $t$  appears multiple times in this multiset—as it will, for instance, if proposal  
592 trees are rejected in Metropolis-Hastings and the last accepted tree is sampled again—each instance will  
593 appear as a separate term in the sum over  $t'$ , reflecting that each is a distinct sample from the posterior  
594 estimate.

#### 595 6.2.4 Modifying trees via tree proposals

596 To generate a new proposal tree  $t'$  from an existing tree  $t$ , Pairtree relies on tree updates similar to those  
597 established in [15, 33]. The algorithm modifies  $t$  by moving an entire sub-tree under a new parent, or  
598 by swapping the position of two nodes. Specifically, Pairtree generates a pair  $(A, B)$ , where  $B$  denotes  
599 a tree node to be moved, and  $A$  represents its destination. This pair is subject to the constraints  
600  $\{A, B\} \subset \{0, 1, \dots, K\}$ , such that  $A \neq B$ ,  $A$  is not the current parent of  $B$ , and  $B$  is not the root node 0.  
601 Two possible cases result. If  $A$  is a descendant of  $B$ , then the positions of  $A$  and  $B$  are swapped, without  
602 modifying any other tree nodes. This implies that the previous descendants of  $B$  (excluding  $A$  itself)  
603 become the descendants of  $A$ , while the previous descendants of  $A$  become the (only) descendants of  $B$ .  
604 Otherwise,  $A$  is not a descendant of  $B$  (i.e.,  $A$  is an ancestor of  $B$ , or  $A$  is on a different tree branch),  
605 and so the sub-tree with  $B$  at its head is moved so that  $A$  becomes its parent. Observe that both moves  
606 can be reversed, which is a necessary condition for the Markov chain to satisfy detailed balance. In the  
607 first case, if  $A$  was descendent of  $B$  in  $t$ , then the pair  $(B, A)$  applied to the tree  $t'$  will restore  $t$ . In the  
608 second case, if  $A$  was not descendent of  $B$  in  $t$ , and  $B$ 's parent in  $t$  was node  $P$ , then the pair  $(P, B)$   
609 applied to tree  $t'$  will restore  $t$ .

610 Pairtree provides two means of choosing the pair  $(A, B)$ . The first mode uses the pairs tensor to  
611 inform tree proposals (Section 6.2.5). The second mode proposes tree updates blindly without reference  
612 to the data (Section 6.2.6), and is helpful for escaping pathologies associated with the first mode. Pairtree  
613 randomly selects between these modes for each update (Section 6.2.6).

## 6.2.5 Using the pairs tensor to generate tree proposals

One of Pairtree’s key contributions is to recognize that the pairs tensor provides an effective guide for tree search, conferring insight into what portions of an existing tree suffer from the most error, and how those portions should be modified to reduce error. To create the proposal  $(A, B)$  for modifying the tree  $t$ , as described in Section 6.2.3, Pairtree generates discrete probability distributions  $W^{(A,B)}$  and  $W^{(B)}$ , corresponding to distributions over  $0, 1, \dots, K$  that are used to sample  $A$  and  $B$ , respectively. The choice of  $B$  depends only on the current tree state  $t$ , and so we denote the corresponding probability distribution as  $W^{(B)}$ . The choice of  $A$ , conversely, depends both on the current tree state  $t$  and whatever choice we made for  $B$ , and so we denote the corresponding probability distribution as  $W^{(A,B)}$ . Every  $W^{(A,B)}$  and  $W^{(B)}$  depends solely on the tree state, such that the Markov chain used for Metropolis-Hastings is time-invariant.

The algorithm generates the probability distribution  $W^{(B)}$  such that the most probability mass is placed on elements corresponding to tree nodes with the highest pairwise error. First, observe that a tree induces a pairwise relationship between every pair of mutations—i.e., a tree places every mutation pair in a coincident, ancestral, descendent, or branched relationship. In Section 6.1, we described how to use mutation read counts to compute a probability distribution over these four relationships for every pair. For a given mutation  $B$ , we can thus compute the joint probability of the pairwise relationships between  $B$  and every other mutation induced by the tree  $t$  to determine how well-placed  $B$  is within the tree. Consider the mutation pair  $(k, B)$ . If  $p(M_{kB}|x_k, x_B)$  represents the probability of the pair taking pairwise relation  $M_{kB}$ , then the probability of the pair taking one of the three other possible relationships is  $p(\neg M_{kB}|x_k, x_B) = 1 - p(M_{kB}|x_k, x_B)$ , which we can think of as the pairwise relationship error. Then, the joint pairwise relationship error for all  $K - 1$  pairs that include  $B$  is  $E(B) \equiv p(\neg M_{1B}, \neg M_{2B}, \dots, \neg M_{KB}|x_1, x_2, \dots, x_B, \dots, x_K) = \prod_{k \neq B} 1 - p(M_{kB}|x_k, x_B)$ .

We compute the probability distribution  $W^{(B)}$ , whose elements represent the probability of selecting the node  $B$  to be moved within the tree, in accordance with the pairwise relationship error  $E(B)$ . To accomplish this, we treat  $\log E(B)$  as the logarithms of elements in an unnormalized probability distribution. To normalize the tuple  $(E(1), E(2), \dots, E(K))$  to create a probability distribution, we use the scaled softmax function  $\text{ssmax}(x) \equiv \text{softmax}(Sx)$ , where the  $S$  scalar is chosen such that  $\frac{\max(\text{ssmax}(x))}{\min(\text{ssmax}(x))} \leq R \equiv 100$ . Specifically, the  $S$  scalar is set to 1 if  $\frac{\max(\text{softmax}(x))}{\min(\text{softmax}(x))} \leq R$ , or otherwise to whatever value greater than 1 is necessary to make  $\frac{\max(\text{softmax}(Sx))}{\min(\text{softmax}(Sx))} = R$ . The scaled softmax can be understood as a “softer softmax,” ensuring no element in  $W^{(B)} \equiv \text{ssmax}((\log E(1), \log E(2), \dots, \log E(K)))$  has more than 100



645 times the probability mass of any other. In practice, this results in every tree node having a non-trivial  
646 probability of being selected for modification.

647 With the probability distribution  $W^{(B)}$  established, we sample  $B \sim W^{(B)}$ . We now need to establish  
648 how to compute the probability distribution  $W^{(A,B)}$ , whose elements represent the probability of selecting  
649 the destination  $A$  for the node  $B$ . Critically, pairwise relations provide a computationally efficient means  
650 of evaluating hypothetical trees that modify  $B$ 's position—we can, in fact, test every possible proposal  
651 for  $A \in \{0, 1, \dots, K\} - \{B, P_B\}$ , where  $P_B$  denotes the current parent of  $B$ . With the choice of  $B$  already  
652 made, let  $D_B(A) \equiv \prod_{(j,k)} p(M_{jk}|x_j, x_k)$  represent the joint probability of choosing  $A$  as the destination  
653 for  $B$ . By this formulation,  $(j, k)$  ranges over all  $\binom{K}{2}$  pairs within the set  $\{1, 2, \dots, K\}$ , and  $D_B(A)$   
654 represents the joint probability of all pairwise relations induced by the tree  $t^{(A,B)}$ , which results from  
655 making the modification to tree  $t$  denoted by  $(A, B)$ . Similar to  $W^{(B)}$ , we apply the scaled softmax to  
656 the  $\log D_B(A)$  elements to create  $W^{(A,B)}$ , with  $W^{(A,B)} \equiv \text{ssmax}((\log D_B(1), \log D_B(2), \dots, \log D_B(K)))$ .  
657 We then sample  $A \sim W^{(A,B)}$ .

658 We now have a concrete realization of the  $(A, B)$  pair that we can apply to tree  $t$ , yielding a modified  
659 tree  $t'$ . By using the pairwise relations as a guide, we selected a node (or subtree)  $B$  to modify, whose  
660 selection probability was dictated by the pairwise errors induced by its position in the tree. Then, we  
661 selected a destination  $A$ , which we swapped with the node  $B$  if  $A$  was already a descendant of  $B$ , or  
662 otherwise made the parent of the  $B$  subtree. In choosing  $B$ , we considered only the joint pairwise error  
663 of the  $K - 1$  pairs including  $B$ ; however, in choosing  $A$ , we considered the pairwise probabilities of all  
664  $\binom{K}{2}$  pairs that would result from the modified tree. Considering all pairs is necessary because moving the  
665 whole subtree rooted by  $B$  changed the position of all  $B$ 's descendants, potentially affecting many pairs  
666 that did not include  $A$  or  $B$ .

667 Thus, we selected a modification  $(A, B)$  to  $t$  that should, on average, yield a  $t'$  tree with less error in  
668 pairwise relations. Ultimately, however, the question of whether to accept  $t'$  as a posterior tree sample  
669 is decided by the Metropolis-Hastings decision rule that requires computing new subclonal frequencies  
670  $\Phi'$  for  $t'$ , then considering the likelihood of the previous tree  $p(x|t, \Phi)$  relative to the new likelihood  
671  $p(x|t', \Phi')$ . Intuitively, once  $B$  is chosen, considering the change in pairwise relations induced by every  
672 possible choice of  $A$  captures substantial information about the quality of the tree that would be created  
673 by the  $(A, B)$  modification, while incurring only a modest computational cost. To fully evaluate the  
674 new tree  $t'$ , we must, however, use the full likelihood, which captures more subtle information about  
675 higher-order relations beyond pairwise. Though this is a more reliable indicator of the new tree's quality,  
676 it requires the computationally expensive step of computing  $\Phi'$ , which is why Pairtree does not do this

677 when evaluating potential tree modification proposals.

### 678 **6.2.6 Escaping local maxima in tree space by allowing uniformly sampled tree proposals**

679 Sampling the  $(A, B)$  tree modifications solely using the pairs tensor sometimes results in Pairtree becoming  
680 stuck in local maxima that exist in the tree space whose likelihood is defined with respect to the pairs  
681 tensor, but that have low likelihood in the tree space defined by the tree likelihood. Consequently, the  
682 tree-proposal algorithm may repeatedly propose tree modifications that improve consistency with pairwise  
683 relationships while worsening the overall tree, leading to many successive proposals being rejected. That  
684 is, some tree nodes may have high pairwise error, such that they are often sampled as the  $B$  subtree to  
685 modify. These nodes may in addition have destinations  $A$  within the tree that substantially reduce this  
686 pairwise error, resulting in the  $(A, B)$  modification being sampled with high probability. When the tree  
687  $t'$  induced by this modification is evaluated using the tree likelihood  $p(x|t', \Phi')$ , however, it may have  
688 poor likelihood, resulting in the modified tree being rejected by Metropolis-Hastings. This pathology  
689 occurs because  $t'$  may appear to be a good candidate when only pairwise relations are considered, but  
690 when higher-degree relationships, such as those between mutation triplets, are captured in the subclonal  
691 frequency-based likelihood  $p(x|t', \Phi')$ , the tree is revealed to be poor.

692 Were the tree proposals  $(A, B)$  generated solely using the pairwise relations, Pairtree would repeatedly  
693 propose the same modification only to have it rejected, resulting in the algorithm becoming stuck at a  
694 sub-optimal point in tree space. To overcome this, we added two decision points in the tree generation  
695 process that permit uniformly sampled modifications. Firstly, when sampling the node  $B$  to move within  
696 the tree, Pairtree will use the pairwise relation-informed choice only  $\gamma = 70\%$  of the time. In the other  
697  $1 - \gamma = 30\%$  of cases, Pairtree will sample  $B$  from the discrete uniform distribution over  $\{1, 2, \dots, K\}$ .  
698 Secondly, in  $\zeta = 70\%$  of cases, Pairtree will sample the destination node  $A$  from the discrete uniform  
699 distribution over  $\{0, 1, \dots, K\} - \{B, P_B\}$ , where  $P_B$  denotes the current parent of  $B$ . Both decisions  
700 are made independently and at random when generating the tree proposal, such that a proposal using  
701 pairwise relations for both  $A$  and  $B$  is generated for only  $\gamma\zeta = 49\%$  of tree modifications. Conversely,  
702  $(1 - \gamma)(1 - \zeta) = 9\%$  of tree modifications are generated without considering the pairwise relations in  
703 any capacity. Both  $\gamma$  and  $\zeta$  can be modified at runtime by the user. Their default values were chosen  
704 to ensure that approximately half of tree modification proposals are fully informed by pairwise relations,  
705 while the remaining half ignore the pairwise relations for at least part of the proposal generation, allowing  
706 the algorithm to explore regions of tree space that might otherwise be rendered difficult to reach.

## 707 6.2.7 Tree initialization

708 To sample trees via Metropolis-Hastings, the MCMC chain must be initialized with a tree structure.  
709 Similar to the tree-sampling process, which can generate proposals using the pairs tensor (described in  
710 Section 6.2.5) or without it (Section 6.2.6), the initialization algorithm can use the pairs tensor to infer  
711 reasonable relationships between subclones, or can ignore the pairs tensor and thereby avoid potential  
712 biases that would inhibit tree search.

713 We first describe tree initialization using the pairs tensor. In this mode, Pairtree constructs the tree  
714 in a top-down fashion, selecting subclones to add to the tree with a sampling probability based on which  
715 appear to have the most ancestral relationships relative to subclones not yet added. Once the algorithm  
716 determines which subclone to add, it considers all possible parents from amongst the nodes already  
717 added, sampling a choice based on which induces the least pairwise relation error for all subclones. This  
718 algorithm uses the scaled softmax described in Section 6.2.5.

```
719 1 function init_tree_from_pairs_tensor {  
720 2   # Only the root node exists in the tree initially.  
721 3   let added = {0}  
722 4   # Track which nodes we've added.  
723 5   let unadded = {1, 2, ..., K}  
724 6   # List of edges in tree. This starts as the empty set.  
725 7   # Each element consists of pair '(a, b)',  
726 8   # representing edge from 'a' to 'b'.  
727 9   let tree = {}  
728 0  
729 1   while length(unadded) > 0:  
730 2     # Set 'c_weights' elements according to joint probability  
731 3     # that 'c' is ancestral to other unadded nodes.  
732 4     let c_weights =  $\sum_{k \in \text{unadded} \wedge k \neq c} \log p(M_{ck} = \text{ancestral} | x_c, x_k)$   
733 5     let c_normed = ssmax(c_weights)  
734 6     # 'c_normed' is now a categorical probability distribution, so sample from it.  
735 7     let c_choice = sample(c_normed)  
736 8  
737 9     # Now we have the node to place in the tree.  
738 0     # We must sample a parent for it from the nodes already  
739 1     # placed in the tree. 'p_weights' is a dictionary.  
740 2     let p_weights = {}  
741 3     for p in added:  
742 4       let candidate_tree = tree  $\cup$  {(p, c)}
```

```
743:5     # Let (a, b) range over ever pair of elements
744:6     # generated from the set 'added ∪ c'.
745:7     # We only consider every pair once -- i.e., only (a, b) and not (b, a).
746:8     # We also do not consider the pairs (a, a).
747:9     #  $M_{ab}$  is the relation this pair takes in 'candidate_tree'.
748:0     p_weights[p] =  $\sum_{a,b} \log p(M_{ab}|x_a, x_b)$ 
749:1
750:2     let p_normed = ssmax(p_weights)
751:3     let p_choice = sample(p_normed)
752:4     tree.add((p_choice, c_choice))
753:5     unadded.remove(c_choice)
754:6     added.add(c_choice)
755:7     return tree
756:8 }
```

Listing 1: Tree initialization algorithm using pairs tensor.

757 In the second mode, Pairtree initializes a tree without reference to the pairwise relations, by placing  
758 every subclone as an immediate child of the root. This initialization is unbiased insofar as it imposes no  
759 ancestral or descendent relationships amongst subclones, assuming instead that the Metropolis-Hastings  
760 update scheme can rapidly alter this initial tree to produce a reasonable solution.

761 When initializing an MCMC chain, Pairtree selects between the two initialization modes at random,  
762 with probability  $\iota = 70\%$  of selecting the pairwise-relation-based mode, and  $1 - \iota = 30\%$  probability of  
763 the unbiased mode. The  $\iota$  parameter can be specified by the user, with the default value chosen under  
764 the assumption that Pairtree will typically be used in multi-chain mode, such that different chains will  
765 benefit from different initializations that allow the algorithm to more fully explore tree space.

### 766 6.2.8 Reducing Pairtree’s computational burden using supervariants

767 Pairtree assumes that mutations have been clustered into subpopulations, with “garbage” variants dis-  
768 carded, before the tree-construction algorithm begins. As a result, all mutations within a subpopulation  
769 are rendered *coincident* relative to one another. Mutations within a subclone also share the same evolu-  
770 tionary relationships to all mutations outside the subclone. Thus, to reduce the computational burden  
771 imposed by the method, rather than working with individual mutations, we can instead represent each  
772 subpopulation with a single *supervariant*, then compute pairwise relations between these rather than  
773 their constituent mutations.

774 Conceptually, relative to the individual mutations that compose it, a supervariant should provide a

775 more precise estimate of the subclonal frequency of its corresponding subclone. Specifically, a mutation  $m$   
 776 in a cancer sample  $s$  has  $V_{ms}$  variant reads and  $R_{ms}$  reference reads, yielding total reads  $T_{ms} \equiv V_{ms} + R_{ms}$   
 777 and a VAF  $\equiv \frac{V_{ms}}{T_{ms}}$ . Given a probability of observing the variant allele  $\omega_{ms}$ , we conclude that  $\omega_{ms}T_{ms}$   
 778 reads originated from the variant allele, and so we can estimate the corresponding subclone's subclonal  
 779 frequency by  $\hat{\phi}_{ms} \equiv \frac{V_{ms}}{\omega_{ms}T_{ms}}$ . Each mutation's  $\hat{\phi}_{ms}$  should thus serve as a noisy estimate of its subclone's  
 780 true  $\phi_{ms}$ .

781 Let  $x_{ms}$  represent the data associated with mutation  $m$  in sample  $s$ , such that  $x_{ms} \equiv \{V_{ms}, R_{ms}, \omega_{ms}\}$ .  
 782 Under a binomial observation model (Section 6.2.2), given subclonal frequency  $\phi_{ks}$  for the subclone  $k$   
 783 harboring mutation  $m$  in sample  $s$ , we have the mutation likelihood  $p(x_{ms}|\phi_{ks}) \equiv \text{Binom}(V_{ms}|V_{ms} +$   
 784  $R_{ms}, \omega_{ms}\phi_{ks})$ . Let  $M_k$  be the set of mutations associated with subclone  $k$ . Then, from all  $j \in M_k$ , we  
 785 get the following joint likelihood for cancer sample  $s$ :

$$\begin{aligned} p(M_k|\phi_{ks}) &= \prod_{j \in M_k} p(x_{js}|\phi_{ks}) \\ &= \prod_{j \in M_k} \text{Binom}(V_{js}|V_{js} + R_{js}, \omega_{js}\phi_{ks}) \end{aligned}$$

786 Assuming  $\omega_{js}$  takes the same value  $\omega_{ks}$  for all  $j \in M_k$ , the joint likelihood takes the following form:

$$\begin{aligned} p(M_k|\phi_{ks}) &= \prod_{j \in M_k} \binom{V_{js} + R_{js}}{V_{js}} (\omega_{js}\phi_{ks})^{V_{js}} (1 - \omega_{js}\phi_{ks})^{R_{js}} \\ &= \left[ \prod_{j \in M_k} \binom{V_{js} + R_{js}}{V_{js}} \right] (\omega_{ks}\phi_{ks})^{\sum_j V_{js}} (1 - \omega_{ks}\phi_{ks})^{\sum_j R_{js}} \end{aligned} \quad (11)$$

787 We want the likelihood for the supervariant  $k$  representing the variants in  $M_k$  to take the same func-  
 788 tional form. Thus, we set  $V_{ks} \equiv \sum_{j \in M_k} V_{js}$  and  $R_{ks} \equiv \sum_{j \in M_k} R_{js}$ , yielding the following supervariant  
 789 likelihood.

$$\begin{aligned} p(x_{ks}|\phi_{ks}) &= \binom{V_{ks} + R_{ks}}{V_{ks}} (\omega_{ks}\phi_{ks})^{V_{ks}} (1 - \omega_{ks}\phi_{ks})^{R_{ks}} \\ &= \binom{\sum_j V_{js} + R_{ks}}{\sum_j V_{js}} (\omega_{ks}\phi_{ks})^{\sum_j V_{js}} (1 - \omega_{ks}\phi_{ks})^{\sum_j R_{js}} \end{aligned} \quad (12)$$

790 Observe that Eq. (12) takes the same functional form as Eq. (11), such that they differ only by a

791 constant of proportionality  $C$  that does not depend on  $\phi_{ks}$ .

$$\begin{aligned}
 C &= \frac{p(M_k|\phi_{ks})}{p(x_{ks}|\phi_{ks})} \\
 &= \frac{\prod_{j \in M_k} \binom{V_{js}+R_{js}}{V_{js}}}{\binom{\sum_j V_{js}+R_{js}}{\sum_j V_{js}}} \\
 \therefore p(x_{ks}|\phi_{ks}) &\propto p(M_k|\phi_{ks}) = \prod_{j \in M_k} p(x_{js}|\phi_{ks}) \tag{13}
 \end{aligned}$$

792 Consequently, the supervariant's likelihood accurately reflects the joint likelihood of the subclone's  
 793 constituent variants, while reducing the algorithm's computational burden. In practice, the constant  
 794 factor  $C$  by which the two differ does not matter, as the Metropolis-Hastings scheme (Section 6.2.3) that  
 795 uses the likelihood (Section 6.2.2) requires only the ratio of two likelihoods to navigate tree space, such  
 796 that  $C$  cancels.

797 Of course, Eq. (13) holds only when  $\omega_{ks} = \omega_{js}$  for all  $j \in M_k$ . Most often, we are given diploid  
 798 variants with  $\omega_{js} = \frac{1}{2}$ , and so we fix  $\omega_{ks} = \frac{1}{2}$  for all supervariants. Thus, supervariants are assured to  
 799 accurately represent their constituent variants when those variants are from diploid genomic regions. For  
 800 non-diploid variants with  $\omega_{js} \neq \frac{1}{2}$ , we must rescale the provided data  $x_{js}$  to use a fixed  $\omega_{ks} = \frac{1}{2}$ , allowing  
 801 us to use an approximation of the correct likelihood. To achieve this, we establish the following:

$$\begin{aligned}
 \hat{x}_{js} &= \{\hat{V}_{js}, \hat{R}_{js}, \hat{\omega}_{js}\} \\
 T_{js} &= V_{js} + R_{js} \\
 \hat{T}_{js} &= 2\omega_{js}T_{js} \\
 \hat{V}_{js} &= \min(V_{js}, \hat{T}_{js}) \\
 \hat{R}_{js} &= \hat{T}_{js} - \hat{V}_{js} \\
 \hat{\omega}_{js} &= \frac{1}{2}
 \end{aligned}$$

802 This representation ensures the corrected variant read count  $\hat{V}_{js}$  cannot exceed the corrected total  
 803 read count  $\hat{T}_{js}$ , which could otherwise occur because of binomial sampling noise inherent to the genomic  
 804 sequencing process, or an erroneous  $\omega_{js}$  value that does not correctly reflect a copy number change. Note  
 805 that both  $\hat{T}_{js}$  and  $\hat{V}_{js}$  can take non-integer values. If the original  $\omega_{js} = \frac{1}{2}$ , then the corrected read counts

806 are unchanged from their original values. From this point, for all mutations  $j \in M_k$  associated with  
807 subclone  $k$ , we compute corrected supervariant read counts  $\hat{V}_{ks}$  and  $\hat{R}_{ks}$ :

$$\begin{aligned}\hat{x}_{ks} &= \{\hat{V}_{ks}, \hat{R}_{ks}, \hat{\omega}_{ks}\} \\ \hat{V}_{ks} &= \text{round}\left(\sum_j \hat{V}_{js}\right) \\ \hat{T}_{ks} &= \text{round}\left(\sum_j \hat{T}_{js}\right) \\ \hat{R}_{ks} &= \hat{T}_{ks} - \hat{V}_{ks} \\ \hat{\omega}_{ks} &= \frac{1}{2}\end{aligned}$$

808 Based on Eq. (13), if the mutations  $j \in M_k$  all had  $\omega_{js} = \frac{1}{2}$ , the  $\phi_{ks}$  value we obtain in maximiz-  
809 ing the supervariant likelihood  $p(\hat{x}_{ks}|\phi_{ks})$  is also optimal for the full joint likelihood over the individual  
810 mutations  $p(M_k|\phi_{ks}) = \prod_{j \in M_k} p(x_{js}|\phi_{ks})$ , since the two likelihoods differ only by a constant of propor-  
811 tionality. If some mutations  $j$  had  $\omega_{js} \neq \frac{1}{2}$ , the supervariant likelihood  $p(\hat{x}_{ks}|\phi_{ks})$  approximates the full  
812 joint likelihood, and so the obtained  $\phi_{ks}$  value is only approximately optimal for the latter. To overcome  
813 this, Pairtree's implementation of the rprop optimization algorithm could be easily modified to optimize  
814  $\phi_{ks}$  with respect to the individual variants  $j$ , each with its own  $\omega_{js}$ , rather than the combined supervari-  
815 ant representation that requires a single  $\omega_{ks}$ . Equivalently, rprop could use multiple supervariants per  
816 subclone, with a single supervariant representing all constituent mutations possessing the same  $\omega_{js}$ . The  
817 projection algorithm, however, necessitates using a single supervariant, which in turn requires a single  
818  $\omega_{ks}$ . Though the adjusted supervariant read counts yield only an approximation of the likelihood for non-  
819 diploid mutations, this is not a critical flaw, as projection is already computing a Gaussian approximation  
820 of the likelihood, rather than the exact binomial likelihood used by rprop.

### 821 6.3 Fitting subclonal frequencies to trees

822 Pairtree provides two algorithms for computing subclonal frequencies for a tree structure. Both attempt  
823 to maximize the data likelihood (Section 6.2.2), fitting the observed read count data as well as possible  
824 while fulfilling all constraints imposed by the tree structure. The first algorithm, named *rprop*, is based  
825 on gradient descent (Section 6.3.2), and directly maximizes the tree's binomial likelihood. The second  
826 algorithm, named *projection*, uses techniques from convex optimization to compute subclonal frequencies

827 maximizing the likelihood of a Gaussian approximation to the binomial [34]. While rprop typically  
828 produces higher-likelihood subclonal frequencies than projection, particularly for subclones where the  
829 Gaussian approximation to the binomial is poor, the projection algorithm runs substantially faster with  
830 many subclones (e.g., for 30 subclones or more). By default, Pairtree uses the projection algorithm, but  
831 the user can select rprop at runtime.

### 832 6.3.1 Converting between subclonal frequencies and subpopulation frequencies

833 To permit a more convenient representation, both rprop and projection work with subpopulation fre-  
834 quencies  $H = \{\eta_{ks}\}$ , rather than the subclonal frequencies  $\Phi = \{\phi_{ks}\}$ , where  $k$  and  $s$  are indices over  
835 subclones and cancer samples, respectively. Given a tree structure  $t$ , we can readily convert from one  
836 representation to the other. Let  $D(k)$  represent the set of descendants of subclone  $k$  in tree structure  $t$ ,  
837 and  $C(k)$  represent the set of direct children of subclone  $k$ . Then, in cancer sample  $s$ , we have

$$\phi_{ks} = \eta_{ks} + \sum_{j \in D(k)} \eta_{js} .$$

838 Equivalently, we obtain

$$\eta_{ks} = \phi_{ks} - \sum_{j \in C(k)} \phi_{js} .$$

839 .

840 From the subclonal frequency constraints described in Section 6.2.2, we see that, because the root  
841 node takes  $\phi_{0s} = 1$ , we must have the constraint

$$\sum_{j=0}^K \eta_{js} = 1$$

842 across all  $K$  subclones, and that each individual  $\eta_{js} \in [0, 1]$ . As each cancer sample  $s$  is independent  
843 from every other, both rprop and projection optimize the set  $\{\eta_{ks}\}$  separately for each fixed  $s$ .



### 844 6.3.2 Fitting subclonal frequencies using rprop

845 The rprop algorithm is a simpler version of RMSprop [40, 41], intended for use with full data batches  
846 rather than mini-batches. To perform unconstrained optimization on the parameters  $H_s = \{\eta_{ks}\}$  for a  
847 fixed cancer sample  $s$ , the algorithm first reparameterizes to  $H_s = \text{softmax}(\{\psi_{ks}\})$ , so that we need not  
848 enforce constraints on  $\{\psi_{ks}\}$  but can ensure  $H_s \subset [0, 1]$  and  $\sum_k \eta_{ks} = 1$ .

849 On each iteration, given a tree structure  $t$  and existing subclonal frequencies  $\Phi$ , rprop converts  $\Phi$  to  
850 population frequencies  $H$ , then computes the derivatives

$$\frac{\partial p(x|t, \Phi)}{\partial \psi_{ks}} = \frac{\partial p(x|t, \Phi)}{\partial \eta_{js}} \frac{\partial \eta_{js}}{\partial \psi_{ks}}$$

851 for all subclone combinations  $j$  and  $k$ , using the tree likelihood (Section 6.2.2). The algorithm then  
852 uses the sign of the gradient to update the  $\psi_{ks}$  values, ignoring the gradient's magnitude. For each  
853 value of  $k$ , rprop maintains a step-size parameter  $\lambda_k$ , which is limited to lie within the interval  $[10^{-6}, 50]$ ,  
854 preventing excessively small or large step sizes. The algorithm also maintains a step-size multiplier  $M_{ki}$   
855 for subclone  $k$  on iteration  $i$ , with  $M_{ki} = 1.2$  if  $\text{sign}(\frac{\partial p(x|t, \Phi)}{\partial \psi_{ks}})$  agrees with the sign from the previous  
856 iteration  $i - 1$ , and  $M_{ki} = \frac{1}{2}$  otherwise. Using these values, rprop performs the gradient update

$$\lambda_k := M_{ki} \lambda_k$$

$$\lambda_k := \min(\lambda_k, 50)$$

$$\lambda_k := \max(\lambda_k, 10^{-6})$$

$$\psi_{ks} := \psi_{ks} + \lambda_k \text{sign}\left(\frac{\partial p(x|t, \Phi)}{\partial \psi_{ks}}\right)$$

857 The rprop algorithm continues this process until none of the  $\frac{\partial p(x|t, \Phi)}{\partial \psi_{ks}}$  values exceed  $10^{-5}$  in a particular  
858 iteration, or until  $I = 10000$  iterations elapse, with  $I$  being customizable by the user.

859 To initialize the  $H_s = \{\eta_{ks}\}$  values, we generate initial values  $\hat{\eta}_{ks}$  with the following algorithm.  $C(k)$   
860 represents the set of direct children of  $k$  in the tree.

$$\hat{\phi}_{ks} := \frac{V_{ks}}{\omega_{ks}(V_{ks} + R_{ks})}$$

$$\hat{\phi}_{ks} := \min(1, \hat{\phi}_{ks})$$

$$\hat{\phi}_{ks} := \max(0, \hat{\phi}_{ks})$$

$$\hat{\eta}_{ks} := \hat{\phi}_{ks} - \sum_{j \in C(k)} \hat{\phi}_{js}$$

$$\hat{\eta}_{ks} := \min(1, \hat{\eta}_{ks})$$

$$\hat{\eta}_{ks} := \max(0, \hat{\eta}_{ks})$$

861 Observe that the constraint  $\hat{\eta}_{ks} \in [0, 1]$  is satisfied. To ensure  $\sum_j \hat{\eta}_{js} = 1$ , we finally set  $\hat{\eta}_{ks} := \frac{\hat{\eta}_{ks}}{\sum_j \hat{\eta}_{js}}$ .  
 862 This initialization reflects that, if the provided tree structure  $t$  is consistent with the data and there is  
 863 minimal noise in the data, the  $\hat{\phi}_{ks} = \frac{V_{ks}}{\omega_{ks}(V_{ks} + R_{ks})}$  subclonal frequencies should be close to the maximum  
 864 likelihood estimate for  $\Phi$  in  $p(x|t, \Phi)$ .

### 865 6.3.3 Fitting subclonal frequencies using projection

866 The projection algorithm draws on the approach provided in [34]. The authors describe a method to effi-  
 867 ciently enumerate mutation trees, in which individual nodes correspond to genomic mutations. To make  
 868 this enumeration feasible, they developed an algorithm to rapidly compute tree-constrained subclonal  
 869 frequencies. Using our supervariant representation, we can apply their approach to computing subclonal  
 870 frequencies for clone trees by representing our binomial likelihood with a Gaussian approximation. First,  
 871 we review the authors' notation and map it to the equivalent notation in Pairtree.

- 872 • :  $q$ : number of mutations, equivalent to our number of subclones  $K$
- 873 • :  $p$ : number of cancer samples, equivalent to our  $S$
- 874 • :  $F \in \mathbb{R}^{q \times p}$ : equivalent to our subclonal frequencies  $\Phi$ , with  $F_{vs}$  equivalent to our  $\phi_{ks}$
- 875 • :  $U \in \{0, 1\}^{q \times q}$ : ancestry matrix created from tree structure  $t$ , such that  $U_{j,k} = 1$  iff subclone  $j$  is  
 876 an ancestor of subclone  $k$  or  $j = k$
- 877 • :  $M \in \mathbb{R}^{q \times p}$ : equivalent to our population frequencies  $H = \{\eta_{ks}\}$ , with  $M_{vs}$  equivalent to our  $\eta_{ks}$

878 With  $\mathcal{U}$  representing the set of all ancestral matrices consistent with the perfect phylogeny problem  
879 (Section 10.8), the authors solve the optimization problem  $\min_{U \in \mathcal{U}} \mathcal{C}(U)$ , such that

$$\mathcal{C}(U) = \min_{M, F \in \mathbb{R}^{q \times p}} \|\hat{F} - F\| \text{ subject to } F = UM, M \geq 0, M^T \mathbf{1} = \mathbf{1} .$$

880 Here,  $\|\cdot\|$  is the Frobenius norm, and  $\hat{F} \in \mathbb{R}^{q \times p}$  is the noisy estimate of the subclonal frequencies  
881 obtained from the data. Observe there is a one-to-one correspondence between  $U$  and  $t$ , as changing  
882 the structure of  $t$  will necessarily change ancestral relations described in  $U$ , and vice versa. Thus, the  
883 authors attempt to find the optimal ancestry matrix  $U$ , corresponding to an optimal tree  $t$ , that allows  
884 tree-constrained subclonal frequencies  $F$  best matching the noisy subclonal frequencies  $\hat{F}$  observed in  
885 the data. Ultimately, the authors solve this problem through enumeration. While this scales better than  
886 previous enumerative approaches because of the authors' efficient computation of the optimal  $M$  for a  
887 given ancestry matrix  $U$ , the approach is still rendered infeasible for the large trees that Pairedtree works  
888 with using a search-based method.

889 Useful for Pairedtree is the authors' extremely efficient means of projecting the observed frequencies  $\hat{F}$   
890 on to the space of valid perfect-phylogeny models using Moreau's decomposition for proximal operators  
891 and a tree reduction scheme [34]. We utilize this to quickly compute subclonal frequencies  $\Phi$  for a given  
892 tree  $t$  that corresponds to an ancestry matrix  $U$ . To allow us to use a Gaussian estimate of our binomial  
893 likelihood, the authors developed an extended version of their algorithm [42], in which they additionally  
894 take as input a scaling matrix  $D \in \mathbb{R}^{q \times p}$  with all  $D_{k_s} > 0$ . Using the element-wise multiplication operator  
895  $\odot$ , the modified algorithm computes

$$\mathcal{C}'(U) = \min_{M, F \in \mathbb{R}^{q \times p}} \|D \odot \hat{F} - D \odot F\| \text{ subject to } F = UM, M \geq 0, M^T \mathbf{1} = \mathbf{1} . \quad (14)$$

896 We will refer to the algorithm as the "projection optimization algorithm," and to Eq. (14) as the  
897 "projection objective." We now show how to use the projection objective to compute the MAP for a  
898 Gaussian approximation of our original binomial likelihood. First, observe that our goal is to maximize  
899 the binomial likelihood defined in Section 6.2.2 by finding optimal subclonal frequencies  $\Phi$  for a given  
900 tree  $t$ . Thus, we wish to find

$$\max_{\Phi_s=\{\phi_{ks}\}} p(x_s|t, \Phi_s) = \max_{\Phi_s} p(V_{1s}, V_{2s}, \dots | t, T_{1s}, T_{2s}, \omega_{1s}, \omega_{2s}, \Phi) \text{ subject to } p(\Phi_s|t) \neq 0. \quad (15)$$

901 Here,  $t$  represents the provided tree structure, while  $\Phi_s$  refers to a set of scalar  $\phi_{ks}$  values that  
 902 obey the tree constraints described in Section 6.2.2, with  $p(\Phi_s|t) \neq 0$  indicating that the set obeys the  
 903 constraints. The  $s$  index represents the cancer sample, with each sample optimized independently. Our  
 904 data  $x_s$  consists of, for subclone  $k$ , a count of variant reads  $V_{ks}$  and reference reads  $R_{ks}$ , yielding total  
 905 reads  $T_{ks} = V_{ks} + R_{ks}$ . We define this as a binomial likelihood, in which we are optimizing the  $\phi_{ks}$  values.

$$p(V_{1s}, V_{2s}, \dots | t, T_{1s}, T_{2s}, \omega_{1s}, \omega_{2s}, \Phi) = \prod_k p(V_{ks} | T_{ks}, \omega_{ks}, \phi_{ks}) \quad (16)$$

906 To approximate this using the Gaussian, we perform the following operations.

$$\prod_k p(V_{ks} | T_{ks}, \omega_{ks}, \phi_{ks}) = \prod_k \text{Binom}(V_{ks} | T_{ks}, \omega_{ks} \phi_{ks}) \quad (17)$$

$$\approx \prod_k \mathbb{N}(V_{ks} | T_{ks} \omega_{ks} \phi_{ks}, T_{ks} \omega_{ks} \phi_{ks} (1 - \omega_{ks} \phi_{ks})) \quad (18)$$

$$\propto \prod_k \mathbb{N}\left(\frac{V_{ks}}{\omega_{ks} T_{ks}} \approx \hat{\phi}_{ks} | \phi_{ks}, \frac{\phi_{ks}}{\omega_{ks} T_{ks}} (1 - \omega_{ks} \phi_{ks})\right) \quad (19)$$

$$\approx \prod_k \mathbb{N}(\phi_{ks} | \hat{\phi}_{ks}, \frac{\hat{\phi}_{ks}}{\omega_{ks} T_{ks}} (1 - \omega_{ks} \hat{\phi}_{ks})) \quad (20)$$

907 We relied on the following operations to achieve the above:

- 908 • Eq. (17) defined Eq. (16) with respect to the binomial distribution.
- 909 • Eq. (18) approximated Eq. (17) with the Gaussian distribution. We represent the Gaussian PDF  
 910 for a random variable  $x$  drawn from a Gaussian with mean  $\mu$  and variance  $\sigma^2$  as  $\mathbb{N}(x|\mu, \sigma^2)$ .
- 911 • Eq. (19) divided the Gaussian random variable by the scalar  $\omega_{ks} T_{ks}$ , yielding another Gaussian  
 912 proportional to the preceding. The new Gaussian random variable is  $\frac{V_{ks}}{\omega_{ks} T_{ks}} \approx \hat{\phi}_{ks}$ , our MAP of the  
 913 subclonal frequency  $\phi_{ks}$  for  $\text{Binom}(V_{ks} | T_{ks}, \omega_{ks} \phi_{ks})$ . As  $\phi_{ks} \in [0, 1]$ , we set  $\hat{\phi}_{ks} \equiv \min(1, \frac{V_{ks}}{\omega_{ks} T_{ks}})$ .
- 914 • To achieve a distribution over the unknown  $\phi_{ks}$ , Eq. (20) swaps the Gaussian's random variable  
 915  $\hat{\phi}_{ks}$  and mean  $\phi_{ks}$ , yielding the same Gaussian PDF. Additionally, it approximates the variance of

916 the Gaussian in Eq. (19) by replacing  $\phi_{ks}$  with its MAP in the variance definition.

917 Let the variance of each Gaussian be represented with  $\sigma_{ks}^2 = \max(10^{-4}, \frac{\hat{\phi}_{ks}}{\omega_{ks} T_{ks}}(1 - \omega_{ks} \hat{\phi}_{ks}))$ . We  
 918 set a minimum variance of  $10^{-4}$  to prevent our  $\hat{\phi}_{ks}$  estimates from being too precise to permit effective  
 919 optimization. To transform Eq. (20) into the form required by the projection objective Eq. (14), observe

$$\prod_k \mathbb{N}(\phi_{ks} | \hat{\phi}_{ks}, \sigma_{ks}^2) \propto \exp - \sum_k \frac{(\phi_{ks} - \hat{\phi}_{ks})^2}{\sigma_{ks}^2}. \quad (21)$$

920 Thus, maximizing Eq. (21) is equivalent to optimizing the objective

$$\min_{\Phi_s} \exp \sum_k \frac{(\phi_{ks} - \hat{\phi}_{ks})^2}{\sigma_{ks}^2}. \quad (22)$$

921 As both  $\exp x$  and  $\sqrt{x}$  are monotonic functions, this is equivalent in turn to

$$\min_{\Phi_s} \sqrt{\sum_k \frac{(\phi_{ks} - \hat{\phi}_{ks})^2}{\sigma_{ks}^2}}. \quad (23)$$

922 To complete the transformation of Eq. (23) to the projection objective Eq. (14), we establish the  
 923 following notation.

$$\begin{aligned} D_s &= [D_{1s}, D_{2s}, \dots, D_{Ks}] = \left[ \frac{1}{\sigma_{1s}}, \frac{1}{\sigma_{2s}}, \dots, \frac{1}{\sigma_{Ks}} \right] \\ F_s &= [\phi_{1s}, \phi_{2s}, \dots, \phi_{Ks}] \\ \hat{F}_s &= [\hat{\phi}_{1s}, \hat{\phi}_{2s}, \dots, \hat{\phi}_{Ks}] \\ U &= \text{ancestry matrix corresponding to tree } t \end{aligned}$$

924 Now, Eq. (23) can be rewritten using the Frobenius norm:

$$\begin{aligned} \min_{\Phi_s} \sqrt{\sum_k \frac{(\phi_{ks} - \hat{\phi}_{ks})^2}{\sigma_{ks}^2}} &= \min_{M_s, F_s} \|D_s \odot (F_s - \hat{F}_s)\| \\ &= \min_{M_s, F_s} \|D_s \odot (UM_s) - D_s \odot \hat{F}_s\|, \end{aligned}$$

925 Thus, we can now call the projection optimization algorithm to compute  $F_s$  and  $M_s$ , which are  
 926  $K$ -length vectors representing tree-constrained subclonal frequencies and subpopulation frequencies, re-  
 927 spectively. Both obey the constraints inherent to the tree  $t$  that are expressed through the ancestry  
 928 matrix  $U$ . The  $F_s$  values are the MAP under the Gaussian approximation Eq. (20) of binomial likelihood  
 929 Eq. (17), ultimately achieving a near-optimal solution to the original optimization objective Eq. (15).

## 930 6.4 Creating simulated data

### 931 6.4.1 Parameters for simulating data

932 We first define parameters characterizing the different simulated cancers.

- 933 •  $K$ : number of subpopulations
- 934 •  $S$ : number of cancer samples
- 935 •  $M$ : number of variants
- 936 •  $T$ : number of total reads per variant

937 We created simulated datasets with the following parameter combinations.

Parameter	Values
$K$	3, 10, 30, 100
$S$	1, 3, 10, 30, 100
Mutations per cluster	10, 20, 100
$T$	50, 200, 1000

Table 1: Simulated data parameters. All combinations of these parameter values were used to generate simulated data, excepting cases when  $K \in \{30, 100\}$  and  $S \in \{1, 3\}$ . This provided 144 parameter combinations, with four datasets generated from each, yielding 576 simulated datasets.

938 Observe there are  $4 \times 5 \times 3 \times 3 = 180$  parameter combinations. When  $K \in \{30, 100\}$ , we did not simulate  
 939 datasets with  $S \in \{1, 3\}$  samples, as trees with so many subpopulations and so few cancer samples are

940 unrealistic—to resolve a large number of distinct mutation clusters, a large number of cancer samples is  
941 typically needed. Simulated datasets with  $K \geq 30$  and  $S < 10$  would thus correspond to complex trees  
942 with few cancer samples, posing a highly underconstrained computational problem that would not reflect  
943 how methods perform on realistic datasets. Thus, as there are  $2 \times 2 \times 3 \times 3 = 36$  parameter sets yielding  
944 under-constrained simulations, we used the remaining  $180 - 36 = 144$  sets to generate simulations. For  
945 each valid parameter set, we generated four distinct datasets, yielding  $144 \times 4 = 576$  simulated datasets.

946 Above, rather than setting the number of mutations per dataset  $M$  directly, we instead specified the  
947 average number of mutations per cluster. This reflects that, because each subclone is distinguished by  
948 one or more unique mutations, trees with more subclones should have more mutations. Consequently,  
949 the number of mutations generated per dataset was  $M = K(\text{mutations per cluster})$ . Nevertheless, as  
950 described in Section 6.4.2, mutations are assigned to subclones in a non-uniform probabilistic fashion,  
951 such that the number of mutations in each subclone is only rarely equal to the parameter value for number  
952 of mutations per cluster used when generating the data.

#### 953 6.4.2 Algorithm to generate simulated data

954 We generated simulated data using the following algorithm. Python code implementing this algorithm is  
955 available at <https://github.com/morrislab/pearsim>.

956 1. Generate the tree structure. For each subclone  $k$ , with  $k \in \{1, 2, \dots, K - 1\}$ , sample a parent  
957  $\mathcal{P}(k)$ . We extended the previous subpopulation (i.e.,  $\mathcal{P}(k) = k - 1$ ) with probability  $\mu = 0.75$ , and  
958 otherwise sample  $\mathcal{P}(k)$  from the discrete Uniform( $0, k - 1$ ) distribution. This extension probability  
959 created “linear chains” of successive subpopulations, with each member of the chain taking only a  
960 single child, interrupted sporadically by the creation of new tree branches. As the normal tree root,  
961 denoted as node 0, exists at the outset, node 1 will always take it as a parent. Note that this scheme  
962 allows for the creation of “polyprimary” trees, in which the root 0 takes multiple clonal cancerous  
963 children. Such polyprimary cases are created for approximately  $1 - \mu = 0.25$  of datasets.

964 2. Generate the subpopulation frequencies  $\eta_{ks}$  for each subpopulation  $k$  in each cancer sample  $s$ ,  
965 with  $s \in 1, 2, \dots, S$ . These values were sampled separately for each  $s$ , with  $[\eta_{0s}, \eta_{1s}, \dots, \eta_{Ks}] \sim$   
966  $\text{Dirichlet}(\alpha, \dots, \alpha) = \text{Dirichlet}(0.1, \dots, 0.1)$ . We use the symmetric Dirichlet distribution with a  
967 single  $\alpha$  parameter because we have no reason to desire that any population frequency tend to be  
968 greater or less than others a priori. The choice of  $\alpha$  has important implications for the structure of  
969 the simulated data (Section 10.7). As the  $\eta$  vector is drawn from the Dirichlet, we have  $\sum_{k=0}^K \eta_{ks} = 1$

970 for each sample  $s$ .

971 3. Compute the subclonal frequencies  $\phi_{ks}$  for each subclone  $k$  in each cancer sample  $s$  using the tree  
972 structure and  $\eta_{ks}$  values. Let  $D(k)$  represent the set of  $k$ 's descendants in the tree. Then, we have

$$\phi_{ks} = \eta_{ks} + \sum_{d \in D(k)} \eta_{ds} .$$

973 4. Assign the  $M$  variants to subclones. To ensure every subclones has at least one variant, set the  
974 subclones of the first  $K$  SNVs to  $1, 2, \dots, K$ . To assign the remaining  $M - K$  SNVs, sample  
975 subclone weights from the  $K$ -dimensional Dirichlet( $1, 1, \dots, 1$ ), then sample assignments from the  
976  $K$ -dimensional categorical distribution using these weights.

977 5. Sample read counts for the variants. Let  $A(m) \in \{1, 2, \dots, K\}$  represent the subclone to which  
978 variant  $m$  was assigned. Let  $\omega_{ms} = \frac{1}{2}$  represent the probability of observing a variant read when  
979 sampling reads from the variant's locus, for all subpopulations contained within  $m$ 's subclone,  
980 reflecting a diploid variant not subject to any CNAs. Then, for each cancer sample  $s$ , given the  
981 fixed total read count  $T$  used for all variants in a dataset, we sample the number of variant reads  
982  $V_{ms} \sim \text{Binomial}(T, \omega_{ms} \phi_{A(m),s})$ .

## 983 6.5 Evaluation metrics for method comparisons

### 984 6.5.1 Intuitive explanation of metrics

985 We developed two metrics for evaluating clone-tree reconstruction algorithms that are suitable for use with  
986 multiple cancer samples. The first, termed *VAF reconstruction loss* (henceforth "VAF loss"), measures  
987 how well a tree's subclonal frequencies match the allele frequency for each mutation implied by its CNA-  
988 corrected VAF. Each tree structure permits a range of subclonal frequencies, with the best subclonal  
989 frequencies matching the data as well as possible while also satisfying the tree constraints. Thus, the  
990 VAF loss evaluates a tree by determining how closely its subclonal frequencies match the observed data.  
991 VAF loss is reported in bits per mutation per cancer sample, representing the number of bits required  
992 to describe the data using the tree, normalized to the number of mutations and cancer samples. Lower  
993 values reflect better trees. As LICHeE could not compute subclonal frequencies itself, producing only  
994 tree structures, we used Pairtree to compute the MAP subclonal frequencies for its trees.

995 All evaluated methods report multiple solutions for each dataset, scored by a method-specific likelihood  
996 or error measure. To determine a single VAF loss for each method on each dataset, we used the method-



specific solution scores to compute the expectation over VAF loss (equivalent to the weighted-mean VAF loss). VAF loss is always reported relative to a baseline. For simulated data, the baseline is the VAF loss achieved using the true subclonal frequencies that generated the data. For real data, the baseline is expert-constructed, manually-built trees that were subjected to extensive refinement, with Pairtree used to compute the MAP subclonal frequencies. Thus, VAF loss indicates the average extra number of bits necessary to describe the data using a method’s solutions rather than the baseline solution. Methods can find solutions that fit the data better than the baseline, yielding a negative VAF loss.

The second evaluation metric we developed, termed *relationship reconstruction error* (henceforth “relationship error”), recognizes that a clone tree defines pairwise relations between its constituent mutations, placing every pair in one of the four relationships discussed earlier. Using the set of trees reported by a method for a given dataset, we computed the empirical categorical distributions over pairwise mutation relations, with each tree’s relationships weighted by the likelihood or error measure reported by the method. We then compared these distributions to the distributions imposed by all tree structures permitted by the true subclonal frequencies, computing the Jensen-Shannon divergence (JSD) between distributions for each pair. This yields a relationship error ranging between 0 bits and 1 bit. Using these, we report the joint JSD across all mutation pairs to summarize the quality of the solution set, normalized to the number of pairs. Thus, the relationship error for a given dataset ranges between 0 bits and 1 bit, with smaller values indicating that a method better recovered the full set of trees consistent with the data. We did not apply this metric to real data, whose true subclonal frequencies, and thus true possible tree structures, are unavailable.

### 6.5.2 VAF reconstruction loss

The VAF reconstruction loss represents how closely the subclonal frequencies associated with a method’s clone tree solution set match the simulated data’s VAFs (Section 3.4). The constraints imposed by good solution trees should permit subclonal frequencies that closely match the data. In Section 6.2.2, we described the tree likelihood Eq. (8), which we also use to define the VAF reconstruction loss.

Assume the method provides a distribution over different clone trees  $t$ , with the posterior probability of  $t$  represented as  $p(t)$ , such that  $\sum_t p(t) = 1$ . The loss is defined for each tree  $t$  over the mutation read count data  $x$ , with mutations  $m$  and cancer samples  $s$ . We use  $\phi_{ms}$  to indicate the subclonal frequency in  $t$  for sample  $s$  associated with the subpopulation containing mutation  $m$ . For mutation  $m$  in sample  $s$ , we define the likelihood

$$\begin{aligned} p(x_{ms}) &= \sum_t p(x_{ms}|t)p(t) \\ &= \sum_t p(x_{ms}|\phi_{ms})p(t) \\ &= \sum_t p(t)\text{Binom}(V_{ms}|V_{ms} + R_{ms}, \omega_{ms}\phi_{ms}) \end{aligned}$$

1027 To compute the VAF reconstruction loss  $\epsilon_\Phi$ , we calculate the mean negative log-likelihood across all  
1028  $M$  mutations and  $S$  cancer samples, with

$$\epsilon_\Phi = -\frac{1}{MS} \sum_{m=1}^M \sum_{s=1}^S \log_2 \sum_t p(x_{ms}|\phi_{ms})p(t) . \quad (24)$$

1029 As  $p(x_{ms}|\phi_{ms}) \leq 1$  and  $p(t) \leq 1$ , given that both are discrete distributions, we have  $\epsilon_\Phi \geq 0$ . We  
1030 report VAF reconstruction loss relative to a baseline, though this is not necessary—the absolute metric  
1031 is still useful for quantifying the error in the tree-constrained subclonal frequencies that are part of a  
1032 solution set. Nevertheless, by reporting error relative to a baseline, we can more easily see how well a  
1033 method is faring, given that some datasets will necessarily yield higher absolute VAF losses than others.  
1034 For simulated data, we use as the baseline the true subclonal frequencies that generated the data. For  
1035 real data, we use as the baseline the subclonal frequencies computed by Pairtree (Section 6.3) for our  
1036 expert-derived trees. In both cases, we use Eq. (24) to compute the baseline VAF loss  $\tilde{\epsilon}_\Phi$ , with the  
1037 distribution over trees  $p(t)$  consisting of a single tree, for which  $p(t) = 1$ . This yields the relative VAF  
1038 loss

$$\hat{\epsilon}_\Phi = \epsilon_\Phi - \tilde{\epsilon}_\Phi .$$

1039 These are the values reported in this study for VAF loss. The relative VAF loss  $\hat{\epsilon}_\Phi$  can be negative,  
1040 indicating that a method has found a better solution than the baseline. On simulated data, for instance,  
1041 this can occur if there is only one tree consistent with the simulated subclonal frequencies, and the clone-  
1042 tree-reconstruction method finds only that tree, to which it then fits the MAP subclonal frequencies.  
1043 These will necessarily fit the observed data better than the true frequencies, yielding a negative relative  
1044 VAF loss.

### 1045 6.5.3 Relationship reconstruction error

1046 In determining relationship reconstruction error (Section 3.4), we wish to compare the distribution over  
1047 pairwise mutation relationships imposed by a method's set of candidate solutions relative to the simulated  
1048 truth. Though there was a single true tree structure used to generate the observed data, we cannot simply  
1049 compare the candidate solutions to the relations imposed by this true tree—the observed VAF data are  
1050 noisy reflections of the true subclonal frequencies accompanying the true tree structure, and while the  
1051 true tree will be consistent with the noise-free frequencies (i.e., it will not violate the constraints they  
1052 impose), there may also be other consistent tree structures. Thus, our baseline must be not the single  
1053 set of relationships imposed by the true tree, but the distribution over relationships implied by all tree  
1054 structures consistent with the true subclonal frequencies. Determining this baseline requires that we  
1055 enumerate all such trees (Section 6.5.4). We can then measure the quality of a set of proposed solution  
1056 trees by the extent to which the distribution over pairwise relations they imply recapitulates the baseline.  
1057 To excel according to this metric, methods must be able to recover the full set of trees permitted by the  
1058 observed VAF data, rather than only a single consistent tree. Moreover, methods must be able to deal  
1059 with noise inherent to the VAF observations, such that the methods find trees that make small violations  
1060 of tree constraints if we take the VAFs as exact observations of the subclonal frequencies.

1061 Suppose a dataset consists of  $M$  mutations. Every clone tree built for this dataset by a method  
1062 places each mutation pair  $(A, B)$  unambiguously into one of the four pairwise relationships. We use  
1063  $M_{AB}$  to delineate the pairwise model for the mutation pair induced by a given clone tree. (Provided the  
1064 method uses a fixed mutation clustering provided as input, the coincident relations are determined by the  
1065 clustering, and so are fixed before the method is run.) Assume the method provides a distribution over  
1066 different clone trees  $t$ , with the posterior probability of  $t$  represented as  $p(t)$ , such that  $\sum_t p(t) = 1$ . In  
1067 this case, we can compute the posterior probability of the  $M_{AB}$  relation as  $p(M_{AB}) = \sum_t p(M_{AB}|t)p(t)$ ,  
1068 where

$$p(M_{AB}|t) = \begin{cases} 1 & \text{iff } (A, B) \text{ are in the } M_{AB} \text{ relation in } t \\ 0 & \text{otherwise} \end{cases} .$$

1069 Using the set of true trees (Section 6.5.4), we will define  $p(\tilde{M}_{AB})$  as the distribution over different  
1070 relations for all  $N$  trees consistent with the true subclonal frequencies. For the true tree set, we will  
1071 establish a uniform prior  $p(t) = \frac{1}{N}$ , since no true tree should be privileged over another. For the mutation

1072 pair  $(A, B)$ , we can now compute the Jensen-Shannon divergence (JSD) between a clone-tree-construction  
1073 method's  $p(M_{AB})$  and the true  $p(\tilde{M}_{AB})$ , which we denote as  $\text{JSD}(M_{AB} \parallel \tilde{M}_{AB})$ . We use the base-two  
1074 logarithm in computing JSD, yielding a measurement in bits.

1075 Given  $M$  mutations in a dataset, there are  $\binom{M}{2} = \frac{M(M+1)}{2}$  mutation pairs  $(A, B)$ . We thus define the  
1076 relationship reconstruction error  $\epsilon_R$  for a solution set as the mean JSD between pairs, such that

$$\epsilon_R = \frac{2}{M(M+1)} \sum_{(A,B)} \text{JSD}(M_{AB} \parallel \tilde{M}_{AB}) .$$

1077 Using the mean allows us to compare  $\epsilon_R$  values for datasets with different numbers of mutations, so  
1078 that we can understand which result sets have more or less error. As an aside, though it may be tempting  
1079 to view  $\epsilon_R$  as the joint JSD for all mutation pairs, normalized to the number of mutation pairs, this  
1080 perspective is wrong. The JSD can be defined with respect to the Kullback-Leibler (KL) divergence.  
1081 Under our definition of  $p(M_{AB}|t)$ , every pair is independently distributed, such that the KL divergence  
1082 of the joint distribution over all pairs is equal to the sum of KL divergences of individual pairs. This  
1083 property is not, however, true for the JSD, and so our sum over pairs does not equal the JSD of the joint  
1084 distributions.

1085 Note that relationship error is similar to the probabilistic ancestor-descendant matrix (ADM) metric  
1086 developed in [21], where it is referred to as metric 3B. To represent the ground truth, given  $M$  mutations  
1087 and a single true tree  $\tilde{t}$ , metric 3B constructs four matrices of size  $M \times M$ , which can be represented by the  
1088  $M \times M \times 4$  tensor denoted by  $T$ . Let  $T_{ijk}$  be the binary indicator corresponding to whether mutations  $i$  and  
1089  $j$  fall into pairwise relationship  $k \in \{\text{ancestor, descendant, branched, coincident}\}$  (Section 6.1). Similarly,  
1090 a candidate solution set can be represented with an  $M \times M \times 4$  tensor denoted by  $R$ , with  $R_{ijk}$  indicating  
1091 the probability that mutations  $i$  and  $j$  fall into relationship  $k$ . Both  $T$  and  $R$  are thus akin to the pairs  
1092 tensor computed by Pairedtree. To compute the similarity between  $T$  and  $R$ , the 3B metric concatenates  
1093 the column vectors of each tensor's constituent  $M \times M$  matrices, forming vectors of length  $4M^2$  that we  
1094 denote with  $\vec{T}$  and  $\vec{R}$ . The metric 3B is then computed as the Pearson correlation between  $\vec{T}$  and  $\vec{R}$ ,  
1095 equivalent to the mean-centered cosine similarity between these vectors.

1096 Relationship error differs from metric 3B in two ways [21]. Though both operate on information  
1097 about similarity in pairwise relations between a ground truth and candidate solution set, they compute  
1098 distance differently. Relationship error uses the mean JSD between all pairs, and so ranges between 0  
1099 and 1, while metric 3B uses Pearson correlation, and so ranges between -1 and 1. More importantly,

1100 relationship error's truth is defined with respect to all trees, and thus pairwise relationships, consistent  
1101 with the true subclonal frequencies. Metric 3B, conversely, defines truth with respect to the single tree  
1102 structure used to generate the data. Relationship error thus better reflects a method's performance, as  
1103 it recognizes the fundamental ambiguity in tree structure.

#### 1104 6.5.4 Enumerating trees quickly

1105 To enumerate all trees consistent with the true subclonal frequencies for a simulated dataset, henceforth  
1106 termed “consistent trees,” we first construct a directed graph  $\tau$ . Given  $K$  subclones and  $S$  cancer  
1107 samples,  $\tau$  consists of a graph of  $K + 1$  nodes, with the  $i$ th node corresponding to the  $i$ th subclone,  
1108 and the implicit node 0 that has no incoming edges. We place an edge from node  $i$  to node  $j$  in  $\tau$ ,  
1109 such that  $\tau_{ij} = 1$ , if node  $i$  is a potential parent of subclone  $j$  in a tree consistent with the subclonal  
1110 frequencies  $\Phi = \{\phi_{ks}\}$ . The  $\tau$  graph represents edges that will be present in at least one consistent  
1111 tree. Thus, the spanning trees of  $\tau$  compose a superset of the consistent trees—i.e., all consistent trees  
1112 exist as a spanning tree of  $\tau$ , but not all spanning trees of  $\tau$  must be consistent trees.

1113 By definition,  $\phi_{0s} = 1$  for all cancer samples  $s$ . Without loss of generality, assume  $\phi_{is} \geq \phi_{(i+1)s}$  for  
1114  $i \in \{1, 2, \dots, K - 1\}$  for all cancer samples  $s$ , as the subclones can be sorted to fulfill this requirement  
1115 without affecting the problem structure. We then construct  $\tau$  as follows.

```
1116 1 function make_tau( $\Phi$ ) {  
1117 2     let  $\tau = \{0\}$  # Dimensions:  $(K + 1) \times S$   
1118 3     for  $i \in (0, 1, \dots, K - 1)$ :  
1119 4         for  $j \in (i + 1, \dots, K)$ :  
1120 5             let  $\tau_{ij} = 1$  iff  $\phi_{is} \geq \phi_{js} \forall s$   
1121 6     return  $\tau$   
1122 7 }
```

Listing 2: Algorithm to create graph adjacency matrix  $\tau$ .

```
1123 1 function enum_trees( $\Phi$ ,  $\tau$ , traversal  $\in$  {DFS, BFS}) {  
1124 2     # Each partial is a triplet.  
1125 3     # Element 1: index  $j \in \{1, 2, \dots, K\}$  of node for which we must  
1126 4     # next resolve parent, with  $1 \leq j' < j$  fully resolved and  $j'' > j$  not yet resolved  
1127 5     # Element 2: graph  $\tau'$  whose edges are a subset of those in  $\tau$ ,  
1128 6     # with in-degree of nodes  $1 \leq i < j$  equal to 1.  
1129 7     # Element 3: sum of subclonal frequencies for each node's children, for the portions  
1130 8     # of the graph that have been fully resolved. This data structure allows us to  
1131 9     # quickly determine whether a prospective parent choice violates tree constraints.
```

```
1132 0     let partials = [(1, copy( $\tau$ ), zeroes( $K+1$ ,  $S$ ))]
1133 1     let trees = []
1134 2
1135 3     while len(partials) > 0:
1136 4         if traversal == DFS:
1137 5             # For depth-first search, remove last element
1138 6              $j$ ,  $\tau'$ , childsum = trees.pop(-1)
1139 7         else:
1140 8             # For breadth-first search, remove first element
1141 9              $j$ ,  $\tau'$ , childsum = trees.pop(0)
1142 0
1143 1         if  $j == K+1$ :
1144 2             # We have resolved a single parent for nodes  $i \in \{1, 2, \dots, K\}$ ,
1145 3             # so the tree is fully resolved.
1146 4             trees.append( $\tau'$ )
1147 5             continue
1148 6         parents = { $i | i \in \{0, 1, \dots, K\} \wedge \tau_{ij} = 1$ }
1149 7         for  $i$  in parents:
1150 8             # It's possible to leave this loop with all possible parents having been
1151 9             # deemed invalid because of a previous parent choice made in this
1152 0             # partial tree. In that case, the partial tree is effectively discarded.
1153 1             if  $\exists s$  s.t.  $\text{childsum}[i, s] + \phi_{js} > \phi_{is}$ :
1154 2                 # We violate tree constraints in cancer sample  $s$ ,
1155 3                 # so reject this as a potential solution.
1156 4                 continue
1157 5             new_csum = copy(childsum)
1158 6             new_csum[ $i, s$ ] +=  $\phi_{js} \forall s$ 
1159 7              $\tilde{\tau} = \text{copy}(\tau')$ 
1160 8             # We don't violate tree constraints, so for this partial tree,
1161 9             # resolve  $j$ 's parent as node  $i$ .
1162 0              $\tilde{\tau}_{cj} = 0 \forall c \in \{0, 1, \dots, K\} - \{i\}$ 
1163 1             partials.append(( $i+1$ ,  $\tilde{\tau}$ , new_csum))
1164 2         return trees
1165 3 }
```

Listing 3: Algorithm to enumerate trees based on  $\tau$  graph.

1166 By implementing this algorithm in Python and exploiting Numba, we can enumerate trees for all  
1167 576 simulated datasets quickly. Improving runtime through parallelization would be trivial, given that the  
1168 algorithm need make only a single pass through each  $\tau'$  graph, without having to backtrack “up” the

1169 graph to alter edges corresponding to fully resolved parents. Though the algorithm offers the choice of  
1170 DFS or BFS when exploring the  $\tau$  graph, DFS is generally superior. As the tree enumeration algorithm  
1171 proceeds down the  $\tau$  graph, DFS allows it to quickly determine whether a parental choice made upstream  
1172 of the nodes being considered was invalid, making it impossible for a downstream node to find any parent.  
1173 DFS will quickly find this parent-less downstream node and so discard the partial tree. BFS, conversely,  
1174 will keep the invalid partial tree in memory as it futilely resolves parents of other nodes before locating  
1175 the parent-less node, while also storing in memory other variants of the invalid partial tree that retain  
1176 the erroneous parental choice. The memory demands of the BFS algorithm variant can thus be much  
1177 higher than DFS, while conferring no benefit.

1178 Additionally, we could alter the `make_tau` algorithm to remove edges that are clearly invalid before  
1179 beginning enumeration. Suppose in  $\tau$  we have a node  $j$  whose only possible parent is  $i$ , and that there  
1180 exists another node  $k$  who is also a possible child of  $i$ , implying  $\phi_{is} \geq \phi_{js}$  and  $\phi_{is} \geq \phi_{ks}$  for all cancer  
1181 samples  $s$ . Furthermore, suppose  $\phi_{is} - \phi_{js} < \phi_{ks}$  for at least one  $s$ . This implies that, by exploiting the  
1182 knowledge that  $i$  must be the parent of  $j$ , we can eliminate  $i$  as being a possible parent of  $k$ . Moreover,  
1183 by eliminating the  $i$ -to- $k$  edge from  $\tau$ , we may have determined with certainty the parent of  $k$ . Supposing  
1184 this is true, we label  $k$ 's parent as  $i'$ , and can eliminate any edges from  $i'$  to other possible children  $k'$   
1185 that would now violate the tree constraints. In this manner, we can propagate constraints through  $\tau$  at  
1186 the algorithm's outset to eliminate edges from consideration. We have not implemented this optimization  
1187 here, as tree enumeration was already sufficiently fast for our purposes.

## 1188 7 Acknowledgements

1189 J.A.W. was supported by a Canada Graduate Scholarship from the National Sciences and Engineering  
1190 Research Council of Canada, a Sir James Lougheed Award of Distinction from the Government of Alberta,  
1191 and additional awards and funding from the University of Toronto Department of Computer Science  
1192 and School of Graduate Studies, the Ontario Institute for Cancer Research, and the Vector Institute  
1193 for Artificial Intelligence. Experiments were run using computational resources provided by SciNet and  
1194 Compute Canada. The authors gratefully acknowledge Bei Jia and José Bento for extending their method  
1195 for computing subclonal frequencies.

## 1196 8 Author contributions

1197 Q.D.M. conceived of and supervised the project. Q.D.M. and J.A.W. designed the project with input  
1198 from S.M.D., and J.A.W. implemented Pairedtree and ran the experiments. J.A.W. and Q.D.M. drafted the  
1199 manuscript, and L.D.S. provided extensive edits and feedback. S.M.D. and J.E.D. designed the project  
1200 and collected the data that motivated Pairedtree’s development, and provided feedback throughout the  
1201 project that guided the design of how Pairedtree reports and visualizes its results. All authors reviewed  
1202 and approved the final manuscript.

## 1203 9 Competing interests statement

1204 J.A.W., S.M.D., J.E.D., L.D.S., and Q.D.M. declare no competing interests.

## 1205 10 Supplementary information

### 1206 10.1 Clustering mutations into subclones

#### 1207 10.1.1 Clustering overview

1208 Pairedtree takes as input a clustering of mutations into subclones. Pairedtree provides two mutation clustering  
1209 algorithms for grouping mutations into subclones. Mutation clusters may also be generated by other  
1210 methods. Alternatively, Pairedtree may be run on the mutations directly without first clustering them into  
1211 subclones, yielding a *mutation tree* instead of a clone tree. A mutation tree is equivalent to a clone tree  
1212 in which each clone bears only a single distinct mutation, such that every tree node corresponds to a  
1213 unique mutation.

1214 Both of Pairedtree’s mutation-clustering algorithms use a Dirichlet process mixture model (DPMM)  
1215 and perform inference via Gibbs sampling. The algorithms differ in how they define their probabilistic  
1216 clustering models. Let  $\Pi = \{\pi_1, \pi_2, \dots, \pi_M\}$  represent a clustering of  $M$  mutations into  $K$  clusters, with  $\pi_i$   
1217 indicating the assignment to a cluster of mutation  $i$ , such that  $\pi_i \in \{1, 2, \dots, K\}$ . Each cluster corresponds  
1218 to a genetically distinct subclone. By virtue of using a DPMM,  $K$  is not fixed, but instead inferred from  
1219 the data.

1220 Let  $x$  represent the mutation read count data. From these, we will define the posterior distribution  
1221 over clusterings



$$p(\Pi|x) = \frac{p(x|\Pi)p(\Pi)}{p(x)} . \quad (25)$$

Each clustering model defines its own likelihood  $p(x|\Pi)$ , but uses the same clustering prior  $p(\Pi)$ . The clustering prior draws on the DPMM concentration hyperparameter  $\alpha$ , representing the cost of placing a mutation in a new cluster relative to adding it to an existing cluster. For  $K$  clusters over  $M$  mutations, with  $n_k$  mutations in cluster  $k$ , we define

$$p(\Pi) = \frac{\alpha^K \prod_{k=1}^K (n_k - 1)!}{\alpha(\alpha + 1) \dots (\alpha + M - 1)} . \quad (26)$$

1222 Both clustering models use Gibbs sampling, such that each clustering iteration resamples the cluster  
 1223 assignment of each mutation individually, conditioned upon the assignments of all other mutations. Thus,  
 1224 we wish to compute  $p(\pi_i|\tilde{\Pi}_i, \tilde{x}_i)$ , where  $\pi_i$  indicates the cluster assignment of mutation  $i$ ,  $\Pi$  is the cluster  
 1225 assignments of all mutations including  $i$ , and  $\tilde{\Pi}_i$  represents the cluster assignments of all mutations  
 1226 excluding  $i$ , such that  $\tilde{\Pi}_i = \Pi - \{\pi_i\} = \{\pi_1, \pi_2, \dots, \pi_{i-1}, \pi_{i+1}, \dots, \pi_M\}$ .

1227 By representing the data associated with all mutations except  $i$  with  $\tilde{x}_i$ , we get

$$\begin{aligned} p(\pi_i|\tilde{\Pi}_i, x) &= \frac{p(\Pi|x)}{p(\tilde{\Pi}_i|x)} \\ &= \frac{p(\Pi|x)}{p(\tilde{\Pi}_i|\tilde{x}_i)} \\ &= \frac{p(x|\Pi)p(\Pi)}{p(x)} \\ &= \frac{p(\tilde{x}_i|\tilde{\Pi}_i)p(\tilde{\Pi}_i)}{p(\tilde{x}_i)} \\ &\propto \frac{p(x|\Pi)p(\Pi)}{p(\tilde{x}_i|\tilde{\Pi}_i)p(\tilde{\Pi}_i)} . \end{aligned} \quad (27)$$

1228 In Eq. (27), we use Eq. (26) to establish

$$\frac{p(\Pi)}{p(\tilde{\Pi}_i)} = \begin{cases} \frac{n_k}{\alpha + M - 1} & \text{if } \pi_i = k \text{ and cluster } k \text{ already exists with } n_k \text{ members} \\ \frac{\alpha}{\alpha + M - 1} & \text{if } \pi_i = k \text{ and cluster } k \text{ is a new cluster} \end{cases} . \quad (28)$$

1229 To complete Eq. (27), we need only define  $\frac{p(x|\Pi)}{p(\tilde{x}_i|\tilde{\Pi}_i)}$ . We leave this definition to the clustering models

1230 described in Section 10.1.2 and Section 10.1.3. Once this factor is defined, we can compute  $p(\pi_i|\tilde{\Pi}_i, x)$   
 1231 because we have in Eq. (27) a quantity proportional to it.

$$p(\pi_i = k|\tilde{\Pi}_i, x) = \frac{\frac{p(x|\tilde{\Pi}_i, \pi_i=k)p(\tilde{\Pi}_i, \pi_i=k)}{p(\tilde{x}_i|\tilde{\Pi}_i)p(\tilde{\Pi}_i)}}{\sum_{k'=1}^K \frac{p(x|\tilde{\Pi}_i, \pi_i=k')p(\tilde{\Pi}_i, \pi_i=k')}{p(\tilde{x}_i|\tilde{\Pi}_i)p(\tilde{\Pi}_i)}} . \quad (29)$$

1232 We use this definition to perform Gibbs sampling, as described in Section 10.1.4.

### 1233 10.1.2 Clustering mutations using subclonal frequencies

1234 For each mutation  $i$  in each cancer sample  $s$ , we have a variant read count  $V_{is}$ , reference read count  $R_{is}$ ,  
 1235 total read count  $T_{is} = V_{is} + R_{is}$ , and probability of observing the variant allele  $\omega_{is}$ . To cluster mutations  
 1236 using subclonal frequencies, we first define for each mutation  $m$  in each cancer sample  $s$  an adjusted total  
 1237 read count  $T'_{ms} = \max(\omega_{ms}T_{ms}, V_{ms})$ . Thus,  $T'_{ms}$  represents the (potentially fractional) number of reads  
 1238 originating from the variant allele across all cells, regardless of whether the reads include mutation  $m$  on  
 1239 that allele. The complete data likelihood is then defined using the following notation:

- 1240 •  $S$ : number of cancer samples
- 1241 •  $K$ : number of clusters
- 1242 •  $M$ : number of mutations
- 1243 •  $\phi_{ks}$ : subclonal frequency of cluster  $k$  in sample  $s$
- 1244 •  $C_k \subseteq \{1, 2, \dots, M\}$ : set of mutations assigned to cluster  $k$ , with  $C_i \cap C_j = \emptyset$  for all  $i$  and  $j$

This yields the complete data likelihood

$$p(x|\Pi) = \prod_{s=1}^S \prod_{k=1}^K \int_0^1 d\phi_{ks} \prod_{m \in C_k} p(x_{ms}|\phi_{ks}) ,$$

with  $p(x_{ms}|\phi_{ks}) = \text{Binom}(V_{ms}|T'_{ms}, \phi_{ks})$ . Strictly speaking, as  $T'_{ms}$  may take a fractional value, it may not be a valid parameter choice for the binomial. Nevertheless, for computational convenience, we compute the integral over the binomial using the beta function, which allows for continuous  $T'_{ms}$  values. Consequently, we have

$$p(x|\Pi) = \prod_{s=1}^S \prod_{k=1}^K \left[ \prod_{m \in C_k} \binom{T'_{ms}}{V_{ms}} \right] \beta(1 + \sum_{m \in C_k} V_{ms}, 1 + \sum_{m \in C_k} T'_{ms} - V_{ms}) . \quad (30)$$

1245 By Eq. (29), we need only define  $\frac{p(x|\Pi)}{p(\tilde{x}_i|\tilde{\Pi}_i)}$  to complete the definitions required for Gibbs sampling.  
 1246 This follows easily from Eq. (30), yielding

$$\frac{p(x|\Pi)}{p(\tilde{x}_i|\tilde{\Pi}_i)} = \prod_{s=1}^S \left( \frac{T'_{is}}{V_{is}} \right) \frac{\beta(1 + V_{is} + \sum_{m \in C_k} V_{ms}, 1 + (T'_{is} - V_{is}) + \sum_{m \in C_k} T'_{ms} - V_{ms})}{\beta(1 + \sum_{m \in C_k} V_{ms}, 1 + \sum_{m \in C_k} T'_{ms} - V_{ms})}. \quad (31)$$

1247 This allows us to proceed with Gibbs sampling, as described in Section 10.1.4.

### 1248 10.1.3 Clustering mutations using pairwise relations

1249 As an alternative to clustering with subclonal frequencies, we can cluster mutations using the pairwise  
 1250 relations described in Section 6.1. To do so, we compute the posterior distributions over pairwise relations  
 1251 for every pair of individual variants  $A$  and  $B$ , rather than the supervariants defined from an established  
 1252 clustering that are used for tree search. Computing the pairwise posterior distributions over relationships  
 1253  $M_{AB}$  necessitates that we first redefine the pairwise prior described in Section 6.1.6 to permit non-zero  
 1254 mass on the *coincident* relationship. For this, we allow the user to set a constant  $P$  representing the  
 1255 prior probability that mutations  $A$  and  $B$  are coincident, with  $P = \frac{1}{4^S}$  for  $S$  cancer samples by default,  
 1256 yielding

$$p(M_{AB}) = \begin{cases} P & \text{if } M_{AB} = \textit{coincident} \\ \frac{1}{3}(1 - P) & \text{if } M_{AB} \in \{\textit{ancestor}, \textit{descendent}, \textit{branched}\} \\ 0 & \text{if } M_{AB} = \textit{garbage} \end{cases} \cdot$$

1257 We define  $p(M_{ab} \neq \textit{coincident}|x) = 1 - p(M_{ab} = \textit{coincident}|x)$ . After computing these pairwise  
 1258 relation posteriors for every mutation pair  $(a, b) \in \{1, 2, \dots, M\} \times \{1, 2, \dots, M\}$  with  $a > b$ , we can define  
 1259 the clustering data likelihood as

$$p(x|\Pi) = \prod_{(a,b)} \mathbb{1}_{\pi_a=\pi_b} p(M_{ab} = \textit{coincident}|x) + \mathbb{1}_{\pi_a \neq \pi_b} p(M_{ab} \neq \textit{coincident}|x). \quad (32)$$

As we consider every pair  $(a, b)$  without also including the pair  $(b, a)$ , there are  $\binom{M}{2}$  factors in the

product for  $M$  mutations. This notation relies on the indicator function

$$\mathbb{1}_c = \begin{cases} 1 & \text{if } c \text{ is true} \\ 0 & \text{otherwise} \end{cases}.$$

1260 From this, we can define  $\frac{p(x|\Pi)}{p(\tilde{x}_i|\tilde{\Pi}_i)}$ , completing the definitions required for Gibbs sampling.

$$\frac{p(x|\Pi)}{p(\tilde{x}_i|\tilde{\Pi}_i)} = \prod_{s=1}^S \left[ \prod_{a \in \{1, 2, \dots, i-1, i+1, \dots, M\}} \mathbb{1}_{\pi_i = \pi_a} p(M_{ia} = \textit{coincident}|x) + \mathbb{1}_{\pi_i \neq \pi_a} p(M_{ia} \neq \textit{coincident}|x) \right]. \quad (33)$$

1261 Thus,  $\frac{p(x|\Pi)}{p(\tilde{x}_i|\tilde{\Pi}_i)}$  is a product over the  $S$  cancer samples and  $M - 1$  pairs that include mutation  $i$ . This  
1262 allows us to proceed with Gibbs sampling, as described in Section 10.1.4.

#### 1263 10.1.4 Performing Gibbs sampling

1264 Pairtree clusters mutations using Gibbs sampling, drawing on the probabilistic framework given in  
1265 Eq. (29), and the subclonal frequency likelihood Eq. (31) or pairwise relationship likelihood Eq. (33).  
1266 The primary advantage of the subclonal frequency model is that, unlike the pairwise model, it does not  
1267 require the time-intensive computation of the pairs tensor before clustering can begin. The pairwise  
1268 model, conversely, can be easily applied to data types other than bulk sequencing that can be represented  
1269 within the pairwise relation framework, such as single-cell sequencing.

1270 By default, the algorithm runs a total of  $C$  chains, with  $C$  set to the number of CPU cores present  
1271 on the system by default, and  $P = C$  executing in parallel. Both  $P$  and  $C$  can be customized by the  
1272 user. Each chain takes 1000 samples by default, which can also be changed by the user. Unlike the  
1273 tree search algorithm, the clustering algorithm makes no attempt to discard burn-in samples from each  
1274 chain. As tree search relies on a single clustering common to all trees, we select the clustering result  
1275 with the highest posterior probability as the algorithm's output. Nevertheless, the user could easily  
1276 adapt the implementation to represent different possible clusterings alongside their posterior probabilities,  
1277 conferring insight into multiple possible solutions.

1278 The subclonal frequency and pairwise relationship clustering models use different clustering initializa-  
1279 tions, purely as an implementation artifact. The subclonal frequency models simply assigns all variants  
1280 to a single cluster. Conversely, the pairwise relationship model places each variant in a separate cluster.

1281 Alternative, the pairwise model also permits the user to specify an initial clustering to use for initializa-  
1282 tion. In this case, user-specified clusters can be merged, but will never be split, such that the user can  
1283 force multiple variants to always remain in the same cluster.

1284 Two hyperparameters affect clustering results. The first,  $\alpha$ , is used in Eq. (26), with higher values  
1285 corresponding to an increased number of clusters. Let  $\hat{\alpha}$  be the value provided by the user as input to the  
1286 algorithm. Given a dataset with  $S$  cancer samples, The  $\alpha$  value used in Eq. (26) is computed from this  
1287 as  $\alpha = 10^{S\hat{\alpha}}$ , with  $\hat{\alpha} = -2$  by default. Representing  $\alpha$  on a logarithmic scale via  $\hat{\alpha}$  makes representing  
1288 especially large or small values of  $\alpha$  more convenient for the user, while scaling it with  $S$  ensures that the  
1289 algorithm's preference for placing data points in new clusters is unaffected by the magnitude of posterior  
1290 weight contributed by data likelihood factors—i.e., each cancer sample-specific likelihood is effectively  
1291 weighted by its own  $10^{\hat{\alpha}}$  prior factor in computing the posterior. Finally, to prevent numerical issues, we  
1292 force  $\alpha \in [\exp(-600), \exp(600)]$ .

1293 The second clustering hyperparameter is  $P$ , the prior probability of two mutations being coincident  
1294 (Section 10.1.3). Similar to how the  $\alpha$  parameter is specified, the algorithm ensures that the number of  
1295 cancer samples  $S$  does not affect the algorithm's preference for starting new clusters by taking as input  
1296  $\hat{P}$ , with  $P = \hat{P}^S$ . By default, we take  $\hat{P} = \frac{1}{4}$ , such that we enforce a uniform distribution over the four  
1297 possible pairwise relations for each cancer sample.

## 1298 10.2 Running comparison methods

1299 All methods were run on systems with dual Intel Xeon 6148 CPUs, with 40 CPU cores and 192 GB of  
1300 RAM. Methods were allowed up to 24 hours of compute time per dataset, and were terminated if they  
1301 exceeded this threshold.

1302 We used CITUP v0.1.2 from <https://anaconda.org/dranew/citup>, corresponding to the most re-  
1303 cent revision at <https://bitbucket.org/dranew/citup/>. CITUP offers both a quadratic integer pro-  
1304 gramming (QIP) mode and a faster iterative approximation to it. We used the QIP mode because it alone  
1305 was able to take a fixed clustering as input. The iterative approximation insists on clustering mutations  
1306 itself, which would have unfairly disadvantaged CITUP relative to other methods, as it would not have  
1307 known which mutations belonged to which clusters. Regardless, we tried running CITUP's iterative mode  
1308 with the same supervariant-based approach we used for PhyloWGS (described below), but this did not  
1309 improve CITUP's failure rate.

1310 We used LICHeE version 26c2a701 from <https://github.com/viq854/lichee>. LICHeE could not  
1311 compute subclonal frequencies, so we invoked Pairtree to perform this task using the tree structures

1312 LICHeE produced. LICHeE can optionally cluster mutations itself, but we gave it the correct mutation  
1313 clustering as input.

1314 We used PASTRI version 1d2fb83c from <https://github.com/raphael-group/PASTRI>, which is  
1315 limited to running on datasets with 15 or fewer subclones. PASTRI was given the correct mutation  
1316 clusters as input.

1317 We used PhyloWGS version 2205be16 from <https://github.com/morrislab/phylowgs>. PhyloWGS  
1318 did not offer a means of taking a fixed clustering as input, unlike the other four methods examined, and  
1319 so was disadvantaged in the method comparisons. We provided as much clustering information to Phy-  
1320 loWGS as possible by using *supervariants* (Section 6.2.8), preventing the method from splitting clusters  
1321 such that mutations from the same cluster would be assigned to different subpopulations. Nevertheless,  
1322 PhyloWGS could still merge clusters such that multiple clusters' variants would be assigned to the same  
1323 subpopulation.

### 1324 10.3 Examining method failures

1325 CITUP produced results for 137 of the three-subclone datasets (76%), failing on the remainder. CITUP  
1326 also failed on all datasets with 10, 30, or 100 subclones. For 3- and 10-subclone failures, 137 exited  
1327 with the error `failed to optimize LP: Infeasible`, while 34 failed with `failed to optimize LP:`  
1328 `Unknown`. Another 52 of the 10-subclone runs failed to finish in 24 hours. All 216 datasets with 30 or 100  
1329 subclones failed with the error `create_trees failed to complete`.

1330 LICHeE succeeded on 477 cases. its 99 failures all occurred on 100-subclone datasets, where the  
1331 method failed to finish in 24 hours.

1332 PASTRI only supports 15 or fewer subclones, and so failed on all 216 datasets with 30 or 100 subclones.  
1333 For 37 datasets with 3 or 10 subclones, PASTRI succeeded in sampling at least one tree with subclonal  
1334 frequencies. On 22 datasets, all of which had 10 subclones, PASTRI failed to finish within 24 hours.  
1335 PASTRI terminated without sampling any trees for 220 datasets, comprising a mixture of 3- and 10-  
1336 subclone cases. Additionally, on 81 datasets, PASTRI sampled one or more trees, but failed at later steps  
1337 of its pipeline, without writing usable output. These 81 cases included four types of failure.

1338 • PASTRI failed the assertion `assert(round(slack[j],10) >= 0)` in `gabow_myers.py` for one ten-  
1339 subclone case.

1340 • PASTRI failed with a `ValueError: too many values to unpack` exception for other cases.

1341 • In some cases, the trees had fewer nodes than expected, despite being given the correct number of  
1342 subclones as input.

1343 • Some cases included invalid blank lines for some of their subclonal frequencies, evidently stemming  
1344 from an error when frequencies of exactly 1 were output as blanks.

1345 PhyloWGS succeeded on 535 datasets. Amongst these, it finished all 1000 burn-in and 2500 posterior  
1346 samples within 24 hours for 463. For another 72 cases, comprising a mixture of 30- and 100-subclone  
1347 datasets, it finished the burn-in samples and at least one posterior sample, without finishing all 2500  
1348 posterior samples. These 72 cases were counted as successes, but assigned wall-clock times and CPU  
1349 times of 24 hours (Section 10.5.2). The remaining 41 runs failed to complete their burn-in portion within  
1350 24 hours, and so were counted as failures. All such cases had 100 subclones.

## 1351 10.4 Why existing algorithms failed

1352 Given that the algorithms we compared against often failed to produce results on our simulated datasets,  
1353 considering possible reasons for this poor performance is a worthwhile exercise. When building trees  
1354 with few subpopulations, exhaustive enumeration algorithms are attractive, as they promise to find the  
1355 single best tree by considering all possibilities. As our simulations demonstrated, however, enumeration  
1356 algorithms cannot cope with more than ten subpopulations, as the number of possible trees becomes  
1357 too great, even when constraints are employed to reduce possible tree configurations. Stochastic search  
1358 algorithms are a superior approach when faced with numerous subpopulations, provided they can locate  
1359 high-likelihood regions of tree space and limit their search to those areas. When this space is searched  
1360 blindly, however, it remains difficult to navigate, given the massive number of possible clone trees formed  
1361 from having many subpopulations.

1362 We hypothesize that CITUP attempted to enumerate all trees with a given number of subpopula-  
1363 tions, but faced too many trees to make this approach feasible when provided with more than three  
1364 subpopulations. Thus, CITUP is limited to datasets with only a small number of subclones.

1365 PASTRI attempted to overcome the difficulties of enumeration by first sampling subclonal frequencies,  
1366 then enumerating only trees consistent with those frequencies. Because mutation VAFs are independent  
1367 from the tree when conditioned upon the subclonal frequencies, PASTRI can treat its approximate pos-  
1368 terior over subclonal frequencies as a proposal distribution for importance sampling, where the target is  
1369 the posterior distribution over subclonal frequencies permitted by the true tree. The PASTRI implemen-  
1370 tation is nevertheless limited to 15 subpopulations [37]. Even with ten subpopulations or fewer, because

1371 PASTRI samples frequencies without considering tree structure, the frequencies are often inconsistent  
1372 with any tree when the algorithm is given many cancer samples, as the frequencies collectively impose  
1373 constraints that rule out all possible trees. A weakness of this approach becomes apparent in real cancer  
1374 datasets, where new subpopulations often emerge when they acquire driver mutations that confer a strong  
1375 selective advantage, leading to them displacing their parents such that the subclonal frequency of the  
1376 child is only slightly greater than that of the parent. Indeed, this situation often occurs in the leukemias  
1377 considered here. As PASTRI samples subclonal frequencies before enumerating consistent trees, the fre-  
1378 quencies sampled for children in this situation will often by chance be slightly higher than their parent,  
1379 rendering the correct tree structure impossible to recover.

1380 LICHeE fared better than CITUP and PASTRI, as it first constructed a directed acyclic graph (DAG)  
1381 containing possible trees permitted by the noisy subclonal frequency estimates provided by the VAFs, then  
1382 only considered spanning trees of this graph [19]. However, this approach could not scale to most 100-  
1383 subpopulation trees, presumably because the corresponding DAGs have too many spanning trees. Even  
1384 in settings with 30 or fewer subclones, LICHeE exhibited considerably higher error than Pairetree both  
1385 with respect to subclonal frequencies and pairwise relations, despite us computing subclonal frequencies  
1386 for LICHeE's tree structures using the same algorithm as Pairetree. This suggests that the DAGs did  
1387 not include as spanning trees good tree candidates, or that the error scoring function LICHeE used  
1388 to indicate tree quality did not properly reflect tree quality. Some of LICHeE's shortcomings may have  
1389 arisen because it takes as input only VAFs, rather than mutation read counts. Consequently, LICHeE has  
1390 no knowledge of how precisely the VAFs should reflect underlying subclonal frequencies, unlike methods  
1391 such as Pairetree that use a binomial observation model.

1392 When PhyloWGS fared poorly, its performance could often be attributed to its inability to use a fixed  
1393 clustering, unlike the other methods. Because we gave PhyloWGS supervariants rather than individual  
1394 mutations in an attempt to mitigate this discrepancy, even though PhyloWGS could not split clusters into  
1395 multiple subclones, the algorithm could effectively merge distinct subclones into single entities, causing  
1396 considerable pairwise relationship error.

1397 Given that non-Pairetree methods may have been particularly prone to failing on the most challenging  
1398 simulations, summary statistics reported for these methods may be unfairly biased in their favour, as  
1399 they would only reflect performance on less-challenging datasets. Nevertheless, when we compare Pairetree  
1400 to each method on only the subset of datasets for which the comparison method succeeded (Fig. S4),  
1401 we see that Pairetree almost always produces better VAF losses, with the only exception being several  
1402 100-subpopulation datasets where PhyloWGS beat Pairetree.



1403 In general, stochastic search algorithms are a superior approach relative to exhaustive enumeration  
1404 methods when faced with numerous subpopulations, since they avoid the exponential growth in number  
1405 of trees as a function of number of subclones [20]. For stochastic search algorithms to work well, they  
1406 must locate high-likelihood regions of tree space and limit their search to those areas. However, as data  
1407 become richer, tree space is rendered more complex, such that existing search algorithms struggle to  
1408 navigate through it. This was apparent with PhyloWGS, which consistently exhibited higher error for  
1409 many-cancer-sample simulations than few-cancer-samples ones. By constructing the pairs tensor and  
1410 using this as a guide to tree search, Pairtree is better able to cope with many cancer samples and the  
1411 constraints they impose.

## 1412 **10.5 Comparing the computational costs of methods**

### 1413 **10.5.1 Criteria for measuring computational costs**

1414 Pairtree and the four methods we compared to it differed substantially in the computational costs they  
1415 imposed, as well as their ability to conduct computations in parallel using multiple CPU cores, using  
1416 either multiple processes or multiple threads. Pairtree, CITUP, and PhyloWGS had the ability to conduct  
1417 computations in parallel, while LICHeE and PASTRI did not. We used this ability only for Pairtree,  
1418 however. For CITUP, using the method's multiple-process mode did not improve its failure rate. Though  
1419 PhyloWGS allows running multiple MCMC chains in parallel, doing so was not helpful for this study—  
1420 PhyloWGS' failures stemmed from an inability to sample enough trees to form a posterior estimate in  
1421 24 hours from a single chain, and so increasing the number of chains only amplified the computational  
1422 burden without improving the failure rate.

1423 We measured runtime on each simulated dataset for each method both with respect to CPU time  
1424 and wall-clock time. CPU time indicates the number of CPU seconds consumed by a method's primary  
1425 process and any subprocesses or threads it spawned, in either user or kernel mode. Wall-clock time  
1426 measures the elapsed time a method took. Runs that exited with an error without producing a result,  
1427 or that failed to finish in 24 hours of wall-clock time, are excluded from the results. Thus, the maximum  
1428 wall-clock time observed for any method is 86,400 seconds. Considering both CPU time and wall-clock  
1429 time is worthwhile—CPU time reflects the total computational burden imposed by a method, while wall-  
1430 clock time indicates how long a method will take to finish in a multi-CPU environment. We conducted  
1431 all experiments on compute nodes using dual Intel Xeon Gold 6148 CPUs, such that 40 CPU cores were  
1432 available to each method. On systems with only one CPU, we expect that wall-clock time will generally

1433 be slightly more than CPU time, as that single CPU must also be used for the operating system and  
1434 other concurrent tasks. In our experiments, however, non-Pairtree methods that used only a single CPU  
1435 core for a run typically achieved wall times that were less than CPU times, given that system or library  
1436 calls they made (e.g., to numerical routines in the Python library NumPy) could be parallelized.

## 1437 10.5.2 Examining method runtime

1438 In cases with 3, 10, or 30 subclones, we see similar patterns of CPU time consumed for Pairtree, LICHeE,  
1439 and PhyloWGS (Fig. S6). These three methods succeeded on all simulations with 30 or fewer subclones,  
1440 simplifying comparisons. Across datasets with 3, 10, or 30 subclones, LICHeE was fastest, realizing  
1441 median CPU times of 0.46 seconds, 1.6 seconds, and 2,722 seconds, respectively. This characterization  
1442 is unfair to other methods, however, as LICHeE did not compute subclonal frequencies for the tree  
1443 structures it produced. To overcome this deficiency, we invoked Pairtree to compute subclonal frequencies  
1444 for LICHeE's results, but did not include the time this step took in LICHeE's CPU time or wall-clock  
1445 time measurements. Pairtree was slower than LICHeE, taking median times of 993 seconds, 1506 seconds,  
1446 and 4391 seconds in settings with 3, 10, or 30 subclones, respectively. PhyloWGS was faster than Pairtree  
1447 for 3-subclone cases, needing only a median CPU time of 509 seconds, but slower in 10- and 30-subclone  
1448 cases, taking median times of 1,781 and 35,472 seconds. When we compare each method's CPU time  
1449 to Pairtree's on only the subset of datasets for which each method succeeded, these observations are  
1450 reinforced, with LICHeE usually being faster than Pairtree excepted for outliers corresponding to 100-  
1451 subclone cases, and PhyloWGS usually being slower than Pairtree (Fig. S8). As CITUP could not produce  
1452 results for datasets with more than three subclones, and PASTRI failed on most three- and ten-subclone  
1453 cases, we do not consider their performance in depth, except to note that CITUP and PASTRI are  
1454 generally fast when they can produce results for three-subclone cases, while PASTRI is slower than all  
1455 other methods on the 4% of 10-subclone datasets where it ran successfully (Fig. S6).

1456 When examining wall-clock times, however, we see that Pairtree fares better because of its use of  
1457 multiple CPU cores. In few-subclone cases, Pairtree is still slower than LICHeE, with Pairtree taking  
1458 median wall times of 55 seconds and 69 seconds in the 3- and 10-subclone settings, respectively, while  
1459 LICHeE took 0.326 and 0.93 seconds, respectively (Fig. S7). Conversely, Pairtree is faster than LICHeE  
1460 in settings with more subclones. For 30-subclone datasets, Pairtree takes a median 148 seconds, while  
1461 LICHeE takes 2,685 seconds. PhyloWGS was considerably slower with respect to wall-clock time than  
1462 LICHeE and Pairtree across all three settings. When runtime on individual datasets is examined, Pairtree  
1463 demonstrates a comparable or superior wall-clock time relative to PhyloWGS and LICHeE (Fig. S9).

1464 Datasets with 100 subclones warrant separate consideration. Pairtree took a median 23,827 seconds  
1465 of CPU time on 100-subclone cases (Fig. S6), but only a median 675 seconds of wall-clock time (Fig. S7).  
1466 LICHeE produced results for only 8% of these datasets, where it took a median 74,790 seconds of CPU  
1467 time. PhyloWGS yielded output for 62% of such datasets, taking median times of 86,400 seconds for  
1468 both CPU time and wall-clock time. The method's median times being equal to 24 hours reflects how  
1469 we handled incomplete runs. According to the (default) parameter settings used for these experiments,  
1470 PhyloWGS discards the first 1000 samples from its MCMC chain as burn-in samples not reflective of  
1471 the true posterior, then takes an additional 2500 posterior samples. If the method finished the 1000  
1472 burn-in samples within the 24-hour wall-clock period permitted, but completed fewer than the 2500  
1473 posterior samples, we used whatever partial set of posterior samples the algorithm produced to evaluate  
1474 its accuracy, while recording its runtime as 24 hours. The median times being 24 hours indicate that  
1475 most successful 100-subclone runs fell into this category. Conversely, the 68% of 100-subclone cases where  
1476 we recorded no output correspond to instances where PhyloWGS could not finish its initial 1000 burn-in  
1477 samples.

### 1478 10.5.3 Evaluating the performance costs of Pairtree's two stages

1479 The two primary steps composing the Pairtree algorithm are computing pairwise relations between sub-  
1480 clones and searching for trees. Tree search includes computing MAP subclonal frequencies for each tree  
1481 structure. The amount of computation needed to build the pairs tensor is fixed, as a distribution over  
1482 relations for every pair must be computed regardless of how many CPU cores are available. As relations  
1483 for each subclone pair are independent of all other subclones, the pairwise computations are embarrass-  
1484 ingly parallel, such that they can be trivially computed in parallel for all pairs. Thus, though the total  
1485 computational burden represented by CPU time is constant, the wall-clock time can be greatly reduced  
1486 by using more CPU cores, with  $N$  cores reducing the time needed for this stage nearly by a factor of  
1487  $N$ . By comparison, tree search requires that each MCMC chain acquire samples serially, such that any  
1488 one chain cannot be parallelized. Multiple chains, however, can execute in parallel, increasing CPU time  
1489 consumed in proportion to the number of chains, but with little effect on wall-clock time.

1490 In the Pairtree experiments illustrated throughout this paper, we used all available 40 CPU cores on  
1491 our compute nodes to calculate pairwise relations in parallel, and to run 40 parallel MCMC chains for tree  
1492 search. Doing so greatly inflated CPU time relative to wall-clock time, but likely was not necessary to  
1493 realize good results. Results of nearly equal quality could perhaps have been obtained from Pairtree using  
1494 fewer chains—while any one chain may become mired in pathological regions of tree space corresponding

1495 to a local optimum, such that multiple chains initialized from different positions can yield better samples,  
1496 we likely did not need all 40 chains to realize this benefit. Nevertheless, even if all 40 chains were necessary  
1497 to produce results of this quality, running those chains serially on a single CPU would have been feasible.  
1498 In this case, the wall-clock time would have been approximately equal to the CPU time. Amongst the  
1499 576 simulations, Pairtree’s longest run was on a 100-subclone, 100-cancer-sample dataset that took 1,110  
1500 seconds of wall-clock time (Fig. S7) and 36,606 seconds of CPU time (Fig. S6). Running all 40 chains  
1501 serially on a single CPU would thus have resulted in a wall-clock time of slightly over 10 hours.

1502 We can understand the relative computational costs of Pairtree’s two primary steps by comparing the  
1503 runtimes of the full Pairtree algorithm to the portion that computes the pairwise relations, denoted as  
1504 *pairs tensor*. By subtracting the pairs tensor runtime from that of full Pairtree, we reveal the cost of  
1505 tree search alone. Comparisons are most informative for the 100-subclone, 100-cancer-sample datasets,  
1506 where the runtimes are longest and differences are thus clearest. For instance, the single most costly  
1507 Pairtree run took 1,110 seconds of wall-clock time and 36,606 seconds of CPU time, as above (Figs. S6  
1508 and S7). Computing the pairs tensor alone took 81 seconds of wall-clock time and 2,666 seconds of  
1509 CPU time. Whether we consider CPU times or wall-clock times, we see 7% of Pairtree’s time went to  
1510 computing pairwise relations, while 93% went to tree search. If the number of CPU cores dedicated to  
1511 this run were cut tenfold to four CPUs rather than 40, we would expect the wall-clock cost of computing  
1512 pairwise relations to increase proportionally to 810 seconds, while the CPU time would remain constant.  
1513 Conversely, the wall-clock cost of tree search could be kept constant at 1,110 seconds by reducing the  
1514 number of MCMC chains to four, at a potential cost in result quality. In this instance, we would expect  
1515 Pairtree to take  $810 + 1,110 = 1,920$  seconds, with tree search consuming 58% of the total. Thus, the  
1516 relative burdens of computing the pairs tensor and performing tree search depend both on the number  
1517 of CPU cores used in parallel, and on the number of MCMC chains from which the user elects to sample  
1518 trees.

## 1519 **10.6 Multiple trees are often consistent with observed data, which Pairtree** 1520 **can accurately characterize**

1521 When building trees, algorithms draw on the subclonal frequencies of constituent subclones across cancer  
1522 samples and relationships between these frequencies to determine possible tree structures. Thus, to assess  
1523 method performance on simulated data, we can enumerate all tree structures consistent with the true  
1524 subclonal frequencies used to generate the data, yielding a distribution over trees. This distribution will

1525 include the true tree used to generate the data, as well as any other tree structures that are also consistent  
1526 with the subclonal frequencies. A perfect method would be able to recover this distribution exactly,  
1527 despite being given only noisy estimates of the true subclonal frequencies via the observed mutation  
1528 frequencies. To evaluate a method, we can then determine the extent to which its tree distribution  
1529 matches the true distribution of all trees consistent with the true subclonal frequencies.

1530 Amongst our 576 simulated datasets, if only one cancer sample is provided, there are usually multiple  
1531 trees consistent with the data (Fig. S10a), regardless of how many subclones are in the tree. This reaches  
1532 an extreme in our ten-subclone, single-sample simulations. This illustrates the importance of understand-  
1533 ing uncertainty in these reconstructions, rather than simply producing a single answer (Section 3.9)—the  
1534 perfect method should represent all of these trees as being equally consistent with the data, such that the  
1535 user should have no reason to prefer any one structure over the others. Drawing on more cancer samples  
1536 reduces this uncertainty, with most ten-sample datasets possessing only a single possible tree across the  
1537 three-, ten-, and 30-subclone settings (Fig. S10a). With 100 subclones, ten samples still permits little  
1538 uncertainty, with the number of possible trees rarely exceeding ten. Note, however, that in this simulated  
1539 setting, multiple samples are likely to be more powerful than they would be for real cancers. Here, each  
1540 sample had its subclonal frequencies generated independently from other samples, increasing the chance  
1541 that the sample induces tree structure constraints because its frequencies are different from all other  
1542 samples. In reality, samples are likely to have correlated frequencies, given that they may be taken from  
1543 similar spatial or temporal sites in the cancer that have similar population proportions.

1544 By computing the entropy of tree distributions, we can characterize how many high-confidence trees  
1545 exist in the distribution. Effectively, the entropy is a posterior-weighted count of the number of trees,  
1546 with the weights in the true tree distribution being uniform because all solutions are equally consistent  
1547 with the data. To determine how many high-confidence solutions was Pairtree was finding relative to  
1548 the number of possible solutions, we compared Pairtree's tree entropy for each simulated dataset to the  
1549 entropy of the true tree distribution (Fig. S10b). Pairtree's entropy generally tracked the true entropy  
1550 well, suggesting that Pairtree's uncertainty was usually consistent with the uncertainty in the true tree  
1551 distribution. Notably, in settings where the number of cancer samples was higher than the number  
1552 of subclones, there was only ever one true tree (Fig. S10a), while Pairtree's tree distribution entropy  
1553 exceeded the true distribution's entropy by more than  $5.9 \times 10^{-6}$  bits with only one exception across 181  
1554 simulations (Fig. S10b). These results demonstrate that, when the data is sufficiently high-resolution as  
1555 to permit only a single solution, Pairtree finds only a single solution.

1556 Though examining tree distribution entropies reveals the number of high-confidence trees Pairtree

1557 finds, it says nothing about the quality of those trees. To gain further insight, we can view a distribution  
1558 over trees as inducing a distribution over the *parents* of each subclone. For a given dataset, to compare  
1559 the Pairtree-computed tree distribution to the distribution of trees consistent with the true subclonal  
1560 frequencies, we can consider the joint Jensen-Shannon divergence between parent distributions induced  
1561 by these tree distributions, normalized to the number of subclones in the tree such that the divergence  
1562 will always lie between zero bits and one bit. We refer to this metric as the *parent JSD*. Even if the tree  
1563 distributions have no overlap—which could occur, for instance, if there is only a single true tree that  
1564 Pairtree fails to locate—the parent JSD nevertheless allows the distributions to have a small divergence if  
1565 they agree on parent choice for most subclones. We see that the parent JSD falls as the number of samples  
1566 increases for a given number of subclones (Fig. S10c), suggesting that Pairtree can efficiently exploit the  
1567 constraints provided by additional cancer samples to produce higher-quality trees. Moreover, when the  
1568 number of samples exceeds the number of subclones such that there is only one tree consistent with the  
1569 true subclonal frequencies (Fig. S10a), the parent JSD is effectively always zero, complementing the tree  
1570 entropy analysis (Fig. S10b) to show that the one tree Pairtree finds is almost perfectly consistent with the  
1571 true tree. Additionally, when the pairwise relation error is examined at a more granular level (Fig. S10d),  
1572 we see that for a given number of subclones and samples it is always less than the parent JSD. This suggests  
1573 that, even when Pairtree doesn't perfectly determine the parents of each subclone, the distributions over  
1574 relationships between subclones (e.g., ancestor-descendant or on-different-branches) are closer to the  
1575 truth. The quality difference between pairwise relation distributions and parent distributions is stark for  
1576 the 100-subclone setting. Though Pairtree only rarely finds the correct parents, demonstrated by the  
1577 parent JSDs that are close to one (Fig. S10c), the pairwise relation errors are much lower (Fig. S10d),  
1578 indicating that the higher-level relationships between subclones are closer to being correct.

## 1579 **10.7 Characteristics of simulated data**

### 1580 **10.7.1 Trees are dominated by small subclones**

1581 Examining statistics of simulated data illustrates factors that affect each clone-tree-reconstruction algo-  
1582 rithm's ability to recover good solutions. The nodes of each clone tree correspond to populations, with  
1583 subclones consisting of sub-trees made up of a population and all its descendants (Section 3.1). Thus,  
1584 a tree with  $K$  populations defines  $K$  subclones. Subclones are nested within trees—a subclone with  
1585 population  $i$  at its head and  $c$  total populations is also part of a subclone with  $i$ 's parent at its head  
1586 and  $c + 1$  total populations (excluding the root subclone that corresponds to the entire tree, which has

1587 no parental subclone). Characterizing subclone composition within simulated data is helpful, as several  
1588 properties of the simulated trees depend on how many populations compose each subclone.

1589 A fully linear tree with no branching that contains  $K$  populations would yield a uniform distribution  
1590 over subclones consisting of  $1, 2, \dots, K$  populations, with exactly one subclone of each size. Branching  
1591 within trees depletes the contribution of larger subclones, replacing them with smaller ones. Because  
1592 of how we constructed simulated tree structures (Section 6.4.2), we see that small subclones dominate  
1593 regardless of the number of populations within a tree (Fig. S11), with most subclones consisting of ten or  
1594 fewer populations in the 30- or 100-subclone trees. In the tree generation algorithm, we choose parents  
1595 for each population in turn, selecting the preceding population as parent with 75% probability, and  
1596 otherwise choosing a parent uniformly from the other nodes already in the tree. As a result, the length of  
1597 linear chains of populations within the tree roughly follows a geometric distribution. Linear chain length  
1598 deviates from the distribution, however, because a node may choose as its parent the end of a different  
1599 chain, allowing that chain to continue extending under a new geometric process.

## 1600 10.7.2 Tree construction becomes increasingly difficult with more subclones

1601 Large trees containing many subclones are more difficult to reconstruct than small trees. In part, this is  
1602 because the number of possible tree structures scales exponentially with the number of populations [20].  
1603 We must also consider, however, how relationships between subclones become more difficult to infer as the  
1604 number of subclones grows, which is a factor independent of tree structure. Given how we generated the  
1605 simulated data (Section 6.4.2), we can derive statistics of the simulated data, then use them to show how  
1606 the difficulty of inferring relationships between subclones changes according to the numbers of subclones  
1607 and cancer samples.

1608 In determining the proper placement of a population within a clone tree, two properties related  
1609 to population frequencies affect the difficulty of this task. Firstly, if a population  $k$  has a near-zero  
1610 population frequency  $\eta_{ks}$  in a cancer sample  $s$ , the VAFs associated with its mutations in that sample  
1611 will be difficult to distinguish from the VAFs of mutations in  $k$ 's parent, which we will denote as population  
1612  $j$ . This occurs because the VAFs for mutations that arose in each population are sampled based on the  
1613 subclonal frequencies of the populations' subclones (Section 6.4.2), which are computed from the sum  
1614 of the population frequencies composing the subclone (Section 6.3.1). Thus, when  $\eta_{ks} \approx 0$ , we have  
1615  $\phi_{ks} \approx \phi_{js}$ , and the VAFs in  $k$  and  $j$  will be nearly the same. Assuming there are no cancer samples other  
1616 than sample  $s$ , we could thus swap the positions of  $k$  and  $j$  in the tree without affecting tree likelihood—  
1617 both populations would have nearly the same subclonal frequency fit to them in the tree, which would

1618 fit the two sets of VAFs almost equally well. Larger population frequencies avoid this situation, making  
1619 clearer the proper ordering of parents and children.

1620 Intuitively, as more populations appear in a tree, the  $\eta_{ks}$  frequencies will become smaller on average,  
1621 as the unit mass apportioned by the Dirichlet distribution from which the frequencies are drawn must be  
1622 split amongst more entities. Indeed, by the properties of the Dirichlet distribution, for  $K$  subpopulations  
1623 in a sample  $s$  with  $[\eta_{0s}, \eta_{1s}, \dots, \eta_{Ks}] \sim \text{Dirichlet}(\alpha, \alpha, \dots, \alpha)$  (Section 6.4.2), we have  $\mathbb{E}[\eta_{ks}] = \frac{1}{K}$ . This  
1624 is evident when we examine the distribution over  $\eta_{ks}$  frequencies for each population in the simulated  
1625 trees (Fig. S12A), where the largest frequency observed across cancer samples for each population is typ-  
1626 ically close to 1 for trees with three subclones, but gets progressively smaller as the number of subclones  
1627 increases, with populations in 100-subclone trees dominated by small frequencies. To distinguish a pop-  
1628 ulation from its parent, it need have a non-negligible  $\eta_{ks}$  frequency in only one sample  $s$ , which is part  
1629 of why adding cancer samples is so helpful in resolving evolutionary relationships between populations,  
1630 and ultimately reconstructing an accurate clone tree.

1631 The second property related to population frequency that affects the difficulty of clone tree recon-  
1632 struction is the variance over cancer samples  $s$  in a subclone  $k$ 's frequencies  $\phi_{ks}$ . Suppose you are trying  
1633 to resolve the position of two subclones  $A$  and  $B$  in a tree, using the frequencies in cancer samples  $s$   
1634 and  $s'$ . To gain the greatest benefit from having two samples rather than only one, we want there to  
1635 be as much variance as possible in the subclonal frequencies between samples. The power of multiple  
1636 samples comes from these differences—for instance, if  $\phi_{As} > \phi_{Bs}$ , but  $\phi_{As'} < \phi_{Bs'}$ , we conclude that  
1637  $A$  cannot be the ancestor of  $B$ , and  $B$  cannot be the ancestor of  $A$ , since an ancestral subclone must  
1638 have a frequency at least as high as its descendants across every cancer sample. This is termed the  
1639 *crossing rule* [36], and leads to the conclusion that  $A$  and  $B$  must occur on separate tree branches. Un-  
1640 fortunately, as we observe only a noisy estimate of the subclonal frequencies through the VAFs, if the  
1641 subclonal frequencies for  $A$  and  $B$  are nearly the same in both samples, the noise in VAFs can obscure  
1642 this relationship. The less variance there is between  $\phi_{As}$  and  $\phi_{As'}$ , and between  $\phi_{Bs}$  and  $\phi_{Bs'}$ , the more  
1643 likely that  $|\phi_{As} - \phi_{Bs}| = |\phi_{As'} - \phi_{Bs'}| < \epsilon$  for some near-zero  $\epsilon$ , and the more difficult it will be to utilize  
1644 the crossing rule with our noisy observations.

1645 Suppose we have a subclone  $C$  composed of  $|C| \leq K$  populations, such that  $C \subseteq \{0, 1, \dots, K\}$ . As  
1646 before, given cancer sample  $s$ , we have population frequencies  $[\eta_{0s}, \eta_{1s}, \dots, \eta_{Ks}] \sim \text{Dirichlet}(\alpha, \alpha, \dots, \alpha)$   
1647 (Section 6.4.2), and  $\phi_{Cs} = \sum_{i \in C} \eta_{is}$ . By the properties of the Dirichlet distribution, we know that the  
1648 sum of Dirichlet-distributed variables is itself Dirichlet-distributed, such that



$$\left[ \sum_{i \in C} \eta_{is}, \eta_{(|C|+1)s}, \dots, \eta_{Ks} \right] \sim \text{Dirichlet}(|C|\alpha, \alpha, \dots, \alpha),$$

1649 where the first element of the vector represents the subclonal frequency  $\sum_{i \in C} \eta_{is} = \phi_{Cs}$ , and the final  
1650  $K - |C|$  elements represent the population frequencies of all populations not in subclone  $C$ . From this,  
1651 we get

$$\text{var}(\phi_{Cs}) = \frac{\frac{|C|}{K}(1 - \frac{C}{K})}{K\alpha + 1}.$$

1652 From the denominator, we see that variance is reduced either with more populations  $K$ , or with a larger  
1653 Dirichlet parameter  $\alpha$ . By plotting both the (theoretical) population standard deviation and (empirical)  
1654 sample standard deviation (Fig. S12B), we see that the latter conforms to the former, and that variance  
1655 is maximized for subclones with  $\frac{K}{2}$  populations, conferring the greatest benefit from multiple cancer  
1656 samples to populations near the root of the tree, such that they have half the populations as descendants.  
1657 Conversely, subclones with less variance in frequency across samples will either be at the very top of the  
1658 tree, with almost all populations as descendants, or at the bottom of the tree, with few populations as  
1659 descendants. Note that, in Fig. S12, the sample standard deviation appears less than the population  
1660 standard deviation, particularly in the three- and ten-subclone cases. This effect is exaggerated for those  
1661 settings because they include single-sample datasets with zero sample standard deviation, whereas the  
1662 30- and 100-subclone datasets do not.

### 1663 10.7.3 Simulated data often include subclones that are impossible to resolve

1664 If a population  $k$  has a near-zero population frequency  $\eta_{ks}$  across all cancer samples  $s$ , its position in a  
1665 clone tree relative to its parent  $j$  is difficult or impossible to resolve. Since  $k$ 's subclonal frequency  $\phi_{ks}$   
1666 is equal to the sum of the population frequencies of all populations in the subclone, when  $\eta_{ks} \approx 0$ , we  
1667 have  $\phi_{ks} \approx \phi_{js}$ . When this occurs, we will have two candidate trees that fit the data equally well—one  
1668 in which  $k$  is the parent of  $j$ , and one in which  $j$  is the parent of  $k$ . Both tree structures would permit  
1669 tree-constrained subclonal frequencies that fit the observed VAF data almost equally well. Well-behaved  
1670 algorithms should find both tree structures. Thus, populations whose frequencies are negligible across  
1671 all cancer samples lead to their subclonal frequencies being nearly equal across all cancer samples, which

1672 leads to ambiguity. In real data, we are unlikely to be faced with this situation. The observed VAFs  
1673 for two variants serve as noisy estimates of their subclones' subclonal frequencies. When the observation  
1674 noise exceeds the negligible differences in the subclonal frequencies, we will deem the two variants as  
1675 having originated from the same subclone, such that the variants are placed in a single cluster.

1676 Nevertheless, examining how often this situation occurs in simulated data is worthwhile, as it grants  
1677 insight into how well algorithms deal with ambiguity. Note that noisy observations of near-zero population  
1678 frequencies are not the only source of ambiguity—ambiguity can exist even given noise-free frequencies,  
1679 or with large population frequencies. All cases where tree enumeration using the noise-free subclonal  
1680 frequencies found multiple trees (Section 6.5.4) are demonstrations of this alternative ambiguity. Tree-  
1681 reconstruction algorithms should be able to deal with both sources of ambiguity by finding the full range  
1682 of solutions permitted for a dataset. With respect to our evaluation metrics, VAF loss (Section 3.4) does  
1683 not capture algorithms' performance in this respect, since it penalizes discrepancies between VAFs and  
1684 tree-constrained subclonal frequencies, and so algorithms can do well regardless of whether they find a  
1685 single good solution or multiple equivalent solutions. Relationship reconstruction error (Section 3.4),  
1686 however, properly reflects algorithms' performance in the face of ambiguity—in the example above in  
1687 which subclones  $j$  and  $k$  had nearly equal subclonal frequencies across all cancer samples, the solutions  
1688 recovered by a tree-reconstruction algorithm should show both that  $k$  could be an ancestor of  $j$ , and  $j$   
1689 could be an ancestor of  $k$ .

1690 To understand the role near-zero population frequencies play in introducing ambiguity, we must first  
1691 define a threshold  $\epsilon$  on population frequencies, such that we will say a population frequency  $\eta$  is near-  
1692 zero if  $\eta < \epsilon$ . This  $\epsilon$  should ideally be defined as a function of read depth, since depth determines  
1693 how precisely the observed VAFs reflect the underlying subclonal frequencies, and ultimately how small  
1694 population frequencies can get before they are swamped by noise. To set this threshold, consider a fixed  
1695 read depth of  $D = 200$ , such that with  $V$  variant reads and  $R$  reference reads we have  $D = V + R = 200$ .  
1696 By our simulation framework, we have  $V \sim \text{Binom}(D, \omega\phi)$ , yielding  $[E](V) = \omega\phi D$ . We will define a  
1697 non-negligible population frequency as that which produces a difference of one read in the mean read  
1698 counts. While this is a subtle difference, we must remember that, in tree search, the read counts for all  
1699 variants belonging to a cluster will be summed, exaggerating the difference in observations for the two  
1700 clusters. Thus, for populations  $j$  and  $k$ , we will assume we have subclonal frequencies  $\phi_j$  and  $\phi_k$  with  
1701  $\phi_j > \phi_k$ . Moreover, assume  $j$  is the parent of  $k$ , such that  $\phi_j = \phi_k + \eta_j$ . This gives us

$$\begin{aligned}\omega\phi_j D - \omega\phi_k D &\geq 1 \\ \phi_j - \phi_k &\geq \frac{1}{\omega D} \\ \eta_j &\geq \frac{1}{\omega D}\end{aligned}$$

1702 With  $\omega = \frac{1}{2}$ , this results in a non-negligible population frequency of  $\eta_j \geq 0.01$  for read depth  $D = 200$ .  
 1703 Conversely, we will define a near-zero population frequency as the complement of this, resulting in a  
 1704 threshold  $\epsilon = 0.01$ . To simplify the analysis, we will use this threshold regardless of read depth. With  
 1705 read depths  $D \in \{50, 200, 1000\}$  (Section 6.4.2), this choice of  $\epsilon$  will yield a greater difference in binomial  
 1706 mean for  $D = 1000$ , and a smaller difference for  $D = 50$ . Nevertheless, the conclusions we reach for fixed  
 1707  $\epsilon$  will be broadly applicable regardless of read depth.

1708 First, we will consider how many populations within each simulated dataset have population frequen-  
 1709 cies less than  $\epsilon = 0.01$  across all cancer samples  $s$ . Let  $\eta_{ks}$  denote the population frequency of population  
 1710  $k$  in cancer sample  $s$ . For  $K$  subpopulations, we have  $[\eta_{0s}, \eta_{1s}, \dots, \eta_{Ks}] \sim \text{Dirichlet}(\alpha, \alpha, \dots, \alpha)$ . By the  
 1711 properties of the Dirichlet distribution, we have

$$\begin{aligned}\eta_{ks} &\sim \text{Beta}(\alpha_{ks}, \sum_{j=0}^K \mathbb{1}_{j \neq k} \alpha_j s) \\ &= \text{Beta}(\alpha, K\alpha) .\end{aligned}$$

1712 Consequently, we since each cancer sample's population frequencies are independent of every other,  
 1713 for  $S$  cancer samples we get

$$\begin{aligned}p(\eta_{k1} < \epsilon, \dots, \eta_{kS} < \epsilon) &= \prod_{s=1}^S p(\eta_{ks} < 0.01) \\ &= \prod_{s=1}^S \int_0^\epsilon dx p(\eta_{ks} = x) \\ &= \prod_{s=1}^S \frac{\beta(\epsilon|\alpha, K\alpha)}{\beta(\alpha, K\alpha)} \\ &= \left[ \frac{\beta(\epsilon|\alpha, K\alpha)}{\beta(\alpha, K\alpha)} \right]^S\end{aligned}\tag{34}$$

1714 Here,  $\beta(\epsilon|\alpha, K\alpha)$  refers to the incomplete beta function, and  $\beta(\alpha, K\alpha)$  refers to the complete beta  
1715 function. Empirically, the proportion of simulated populations with near-zero population frequencies  
1716 across samples agrees with the result predicted above (Fig. S13). Datasets with 30 or 100 populations  
1717 and one or three cancer samples would have at least 38% of populations with near-zero population  
1718 frequencies in all cancer samples, rendering their positions in the tree difficult to resolve. This would  
1719 create excessive ambiguity, which is why we did not include such datasets in our simulated data.

1720 The relationship reconstruction error we used to evaluate method performance on simulated data  
1721 reflected how algorithms dealt with two sources of ambiguity: firstly, the multiple tree structures poten-  
1722 tially permitted by the noise-free frequencies (Section 10.6); and, secondly, the additional tree structures  
1723 permitted by populations with near-zero population frequencies. As we established above, if a population  
1724  $k$  has near-zero population frequencies across all cancer samples, the subclonal frequencies of  $k$  and its  
1725 true parent  $j$  will be almost equal, such that the noisy VAF observations will render difficult the task of  
1726 determining whether  $j$  is the parent of  $k$  or vice versa. Observe that 14% of populations in 100-subclone,  
1727 10-sample trees have noise-free population frequencies less than  $\epsilon = 0.01$  across cancer samples. In the  
1728 average tree, these would correspond to 14 populations with near-zero frequencies. Since each such pop-  
1729 ulation could be swapped with its parent while minimally affecting tree likelihood, these would generate  
1730  $2^{14} \approx 16,000$  additional trees. This assumes that none of the populations with near-zero frequencies have  
1731 edges between them; chains of two or more populations with near-zero frequencies would further increase  
1732 the number of potential tree configurations. We expect noisy observations to be the dominant source  
1733 of ambiguity. In the 100-subclone, 10-sample setting, none of the 36 simulated datasets permitted more  
1734 than 42 trees given the noise-free frequencies (Fig. S10), which is a value far smaller than the 16,000  
1735 trees we expect to be permitted by the noisy observations.

1736 This analysis also helps us understand how many cancer samples we must simulate to remove ambigu-  
1737 ity in tree search arising from noisy observations for a given number of subclones. Taking our threshold  
1738  $\epsilon = 0.01$ , we can ask how many cancer samples we need before  $p(\eta_{k1} < \epsilon, \dots, \eta_{kS} < \epsilon)$ . By solving for  
1739  $S$  in Eq. (34), we find that need 24 or more samples before the probability of a population frequency  
1740 being less than  $\epsilon$  across all samples falls below 1%. This has implications for variant clustering as well,  
1741 since a population's variants become distinguishable from other variants by the clustering algorithm only  
1742 when one or more cancer samples with non-negligible frequencies for the associated population render  
1743 the VAFs clearly distinct.

1744 To complement the above analysis concerning lone populations, we will also examine the probability  
1745 of simulated trees containing sub-trees that consist entirely of populations whose frequencies are less than

1746  $\epsilon = 0.01$ . We define a sub-tree to consist of a subset of the full tree's nodes, as well as all edges between  
1747 them, ensuring the sub-tree is connected. Thus, a sub-tree can correspond to a subclone (Section 3.1),  
1748 but is more general in that may omit parts of the subclone defined by the ancestral population at the root  
1749 of the sub-tree. For this analysis, we did not conduct an empirical examination of the simulated data,  
1750 but used only theoretical results derived from the Dirichlet distribution properties. Given a complete  
1751 tree composed of  $K$  populations as well as the root node 0, and a sub-tree composed of populations  
1752  $T \subseteq \{0, 1, \dots, K\}$  with size  $|T|$ , we have in cancer sample  $s$  the result

$$\begin{aligned} \sum_{i \in T} \eta_{is} &\sim \text{Dirichlet}\left(\sum_{i \in T} \alpha_i, \sum_{j \notin T} \alpha_j\right) \\ &= \text{Dirichlet}(|T|\alpha, (K - |T| + 1)\alpha) \end{aligned}$$

1753 Note that if the sub-tree  $T = \{j\} \cup \{k|k \text{ is descendent of } j\}$ , then  $T$  is equivalent to the subclone with  
1754 population  $j$  at its head, and  $\sum_{i \in T} \eta_{is} = \phi_{js}$ . By using the Dirichlet's marginal beta distribution, as in  
1755 the previous analysis, we can compute the probability of the arbitrary sub-tree  $T$  consisting exclusively  
1756 of populations whose summed frequencies across cancer samples are small, such that  $\sum_{i \in T} \eta_{is} < \epsilon = 0.01$   
1757 for every cancer sample  $s$  (Fig. S14). For instance, in the 100-subclone, single-sample case, we have a  
1758 6% probability of an arbitrary eleven-population sub-tree having a near-zero population frequency sum.  
1759 With  $|T|$  populations in such a sub-tree, there are  $(T + 1)!$  orderings of nodes in the sub-tree that would  
1760 permit nearly equal tree-constrained subclonal frequencies, and thus nearly equal tree likelihood. In the  
1761 eleven-population case, there would thus be  $(11 + 1)! = 4.79e8$  solution trees resulting from this single  
1762 ambiguous sub-tree.

1763 To compute the probability of observing such a case in the simulated trees, we must first consider how  
1764 many linear chains of  $J$  populations exist in a tree with  $K$  nodes, as each has an equal chance of being  
1765 assigned these small frequencies. If a tree is fully linear with no branching, there would be  $(K + 1) - J + 1$   
1766 chains of  $J$  nodes, such that our chain of 11 populations in a 100-subclone tree would have  $101 - 11 + 1 = 91$   
1767 sub-trees, assuming that tree was fully linear. This in turn yields a  $(100\% - 6\%)^{91} = 0.36\%$  chance that  
1768 we would not observe any near-zero-frequency 11-population chains in our tree—i.e., with near certainty,  
1769 we would encounter such a chain. Any degree of branching in a tree can reduce the number of node chains  
1770 of a given length, thereby lessening the chance we would see this scenario. Nevertheless, the probability  
1771 can remain considerable, which is another reason we omitted the many-subclones, few-samples cases from  
1772 our simulated data. Amongst the settings we included, we see, for instance, that in ten-subclone, single-

1773 sample trees, 6% of five-population chains will have small population frequency sums, yielding a 35%  
1774 chance that we would encounter such a case in a fully linear tree.

#### 1775 **10.7.4 Justifying our choice of the Dirichlet parameter for generating simulated data**

1776 In Sections 10.7.1 to 10.7.3, we saw that our choice of the Dirichlet parameter  $\alpha$  when generating simulated  
1777 data (Section 6.4.2) affects multiple aspects of simulated data.

- 1778 1. A smaller  $\alpha$  leads to more variance in population frequencies between samples, increasing the chance  
1779 that multiple samples will make clear the proper pairwise relations between subclones.
- 1780 2. A smaller  $\alpha$  also leads, however, to a greater probability of observing near-zero frequencies for a  
1781 population across all cancer samples, inhibiting tree-reconstruction algorithms' attempts to infer  
1782 the proper place for such populations in the tree. (We do not present results with alternative  $\alpha$   
1783 values here, but used these analyses to inform our choice of  $\alpha$ .)

1784 Our chosen  $\alpha = 0.1$  thus achieved a compromise between three factors.

- 1785 1. It led to sufficient variance in population frequencies between cancer samples for algorithms to  
1786 benefit from having access to multiple cancer samples.
- 1787 2. It avoided creating too many populations with near-zero frequencies across samples, which would  
1788 have created excessive ambiguity.
- 1789 3. Yet it created enough such populations so that we could evaluate how algorithms dealt with ambi-  
1790 guity stemming from this source.

#### 1791 **10.8 Impact of the infinite sites assumption**

1792 To simplify subclonal reconstruction, algorithms make the ISA, which posits that the genome is so large  
1793 as to be effectively infinite in size, meaning that each genomic site is mutated at most once during the  
1794 cancer's evolution. This implies that the same site can never be mutated twice by separate events, and  
1795 that it can never return to the wildtype. Moreover, two cells bearing the same mutation are assumed to  
1796 share a common ancestor in which that mutation occurred. Most clone tree reconstruction algorithms  
1797 make this assumption. Equivalently, ISA violations can be understood as violations of the four-gamete test  
1798 [38]. Under this assumption, the cancer phylogeny is a *perfect phylogeny*, such that descendant subclones  
1799 inherit all the mutations of their ancestors. Critically, the ISA allows us to characterize more subclones

1800 than we have cancer samples. In addition, the ISA is necessary to infer the pairwise relationships between  
1801 mutations from their frequencies (Section 6.1).

1802 Given complete genomes for each cancer cell, a perfect phylogeny can be constructed in linear time  
1803 [43], with mutations that deviate from the ISA detected via the four-gamete test [38]. However, the  
1804 bulk-tissue DNA sequencing data commonly used today do not provide complete genomes. Instead,  
1805 the samples consist of mixtures of different subclones, rendering NP-complete the construction of a  
1806 perfect phylogeny consistent with the exact subclonal frequencies of mutations across multiple samples  
1807 [44]. Nevertheless, the ISA implies relationships between mutation frequencies that can assist subclonal  
1808 reconstruction. Firstly, mutations in ancestral subclones must always have subclonal frequencies at least  
1809 as high as those in descendent subclones, across every observed cancer sample. Secondly, two mutations  
1810 on different tree branches can never have frequencies that sum to greater than one in any sample.

1811 Pairtree can often detect such violations and discard the offending mutations using its garbage rela-  
1812 tion (Section 6.1.3). Specifically, Pairtree’s pairwise-relation-based mutation clustering algorithm (Sec-  
1813 tion 10.1.3) could be trivially modified to use this information to temporarily remove mutations violating  
1814 the ISA. After building a clone tree using all other mutations, the ISA-violating mutations could be lay-  
1815 ered over the tree using a separate inference step. These extensions would also be relevant to scDNA-seq  
1816 settings (Section 10.9).

## 1817 **10.9 Using single-cell DNA sequencing data for building clone trees**

1818 Single-cell DNA sequencing (scDNA-seq) is becoming more popular for studying cancer evolution [45,  
1819 46]. In principle, scDNA-seq gives unambiguous knowledge of each cancer cell’s genotype, avoiding the  
1820 need to deconvolve the signal from many cell subpopulations that is inherent to bulk sequencing. How-  
1821 ever, scDNA-seq data is noisy, with amplification biases giving rise to inaccurate estimates of mutation  
1822 prevalence [47]. The same issues result in many mutations being missed altogether. As a result, bulk  
1823 sequencing will likely remain widely used for many years, including in initial clinical applications of clone  
1824 trees—bulk data gives a more complete depiction of a cancer’s mutation spectrum, and better estimates  
1825 of mutation prevalence.

1826 Nevertheless, scDNA-seq is likely to grow in popularity in the coming years. Pairtree can be extended  
1827 to construct clone trees from single-cell DNA sequencing (scDNA-seq) data. This can be accomplished by  
1828 modifying Pairtree’s pairwise relation framework to use binary valued information about the presence or  
1829 absence of mutations, rather than the mutation’s estimated subclonal frequencies. This would allow trees  
1830 to be built from mixtures of scDNA and bulk data, or from scDNA data alone [17]. Tree search would

1831 remain mostly unchanged, with modifications required only in defining a likelihood that incorporates  
1832 single-cell information.

1833       We have demonstrated that Pairtree can accurately recover clone trees with more subclones than  
1834 cancer samples by deconvolving bulk samples. This suggests the potential for using Pairtree with quasi-  
1835 bulk data, whereby single cells would be pooled together to reduce sequencing costs, then deconvolved  
1836 post-hoc using techniques inspired by compressed sensing. This deconvolution ability could also be useful  
1837 in detecting and resolving cell doublets.



1838 **11 Supplementary figures**

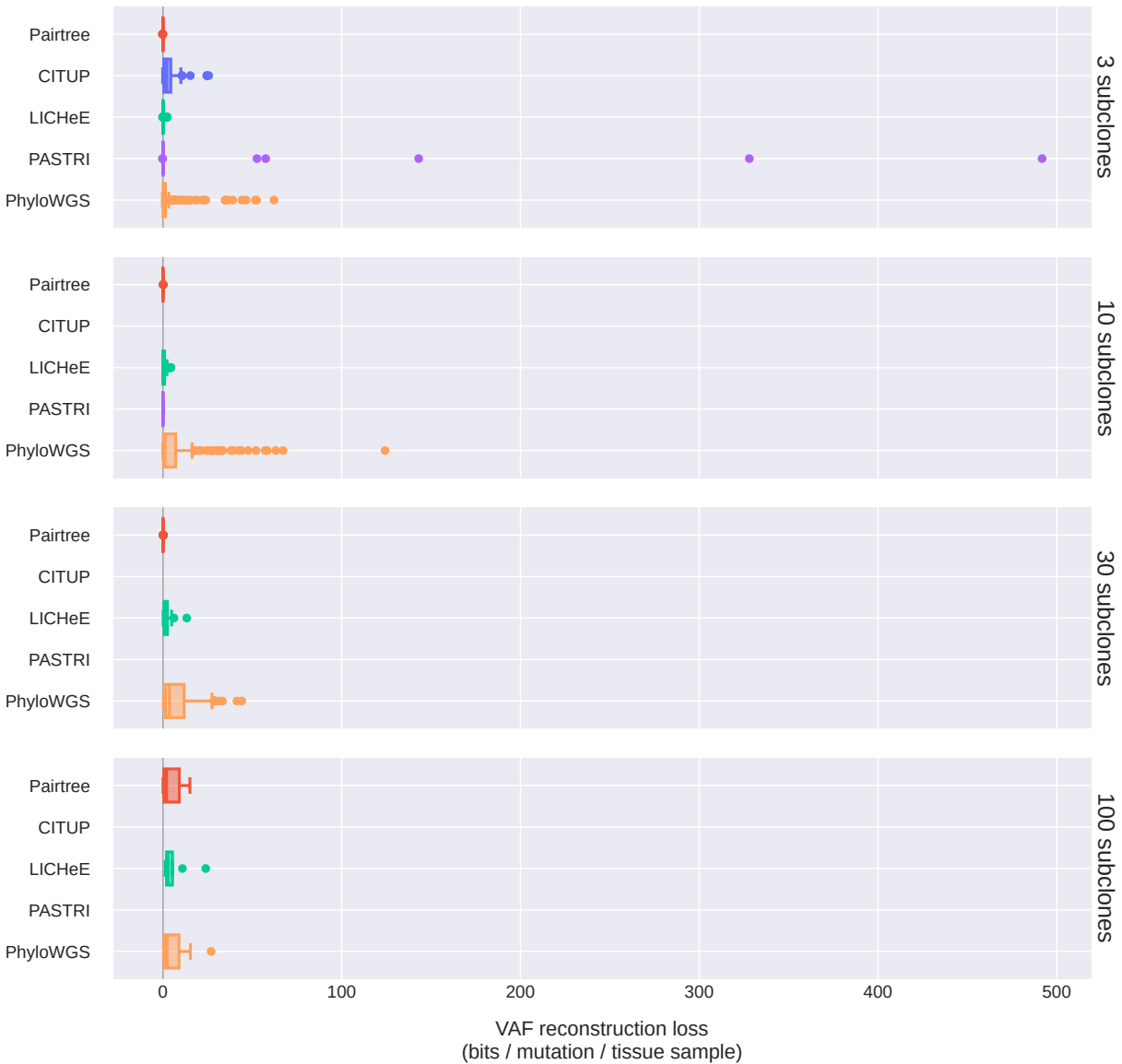


Figure S1: **Untruncated VAF reconstruction losses on 576 simulated datasets.** These results are the same as in Fig. 3b, but without axis truncation. As in the truncated plots, results reflect each method's performance on the subset of datasets where it succeeded in running.

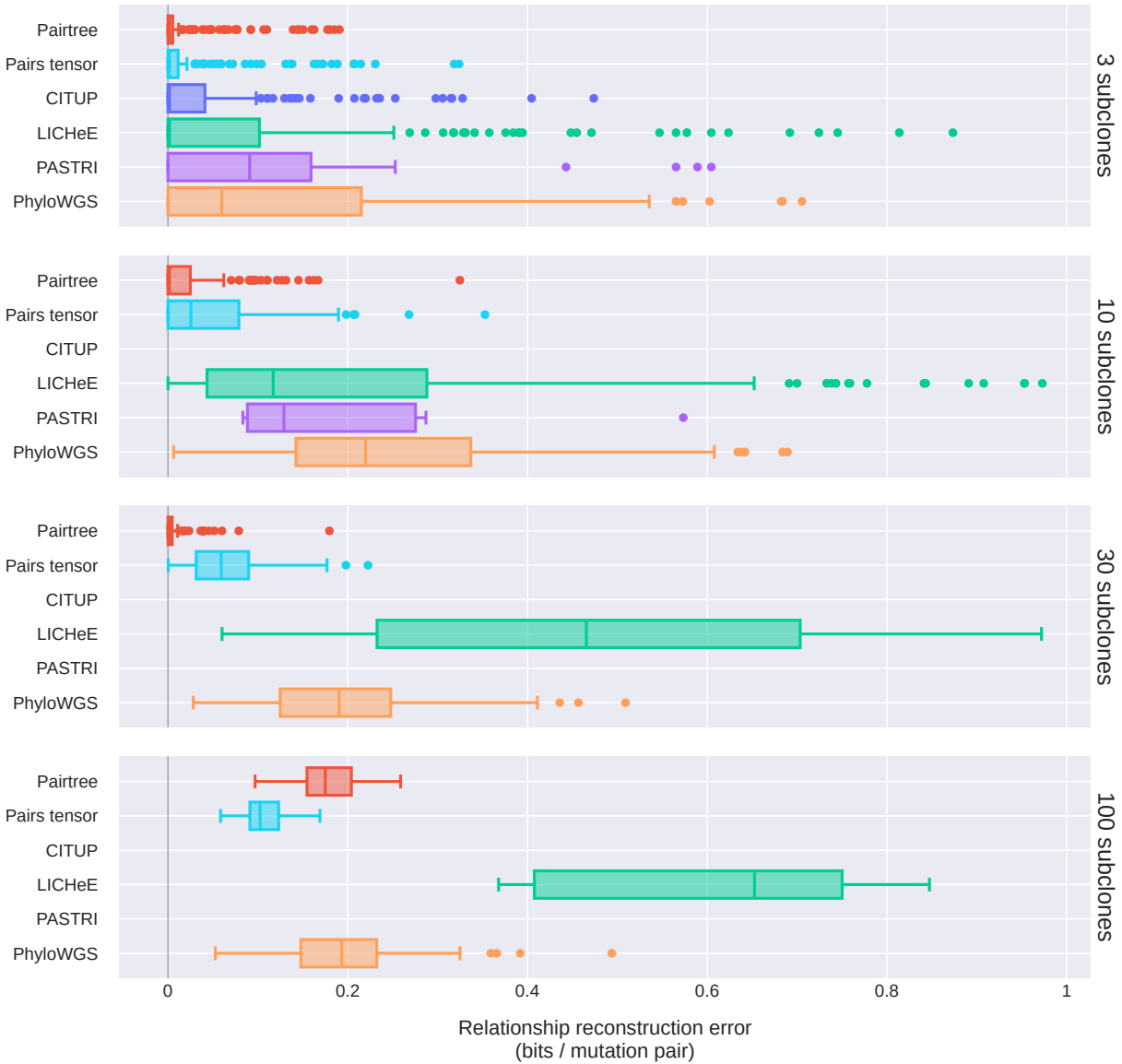


Figure S2: **Untruncated relationship reconstruction errors on 576 simulated datasets.** These results are the same as in Fig. 3c, but without axis truncation. As in the truncated plots, results reflect each method's performance on the subset of datasets where it succeeded in running.

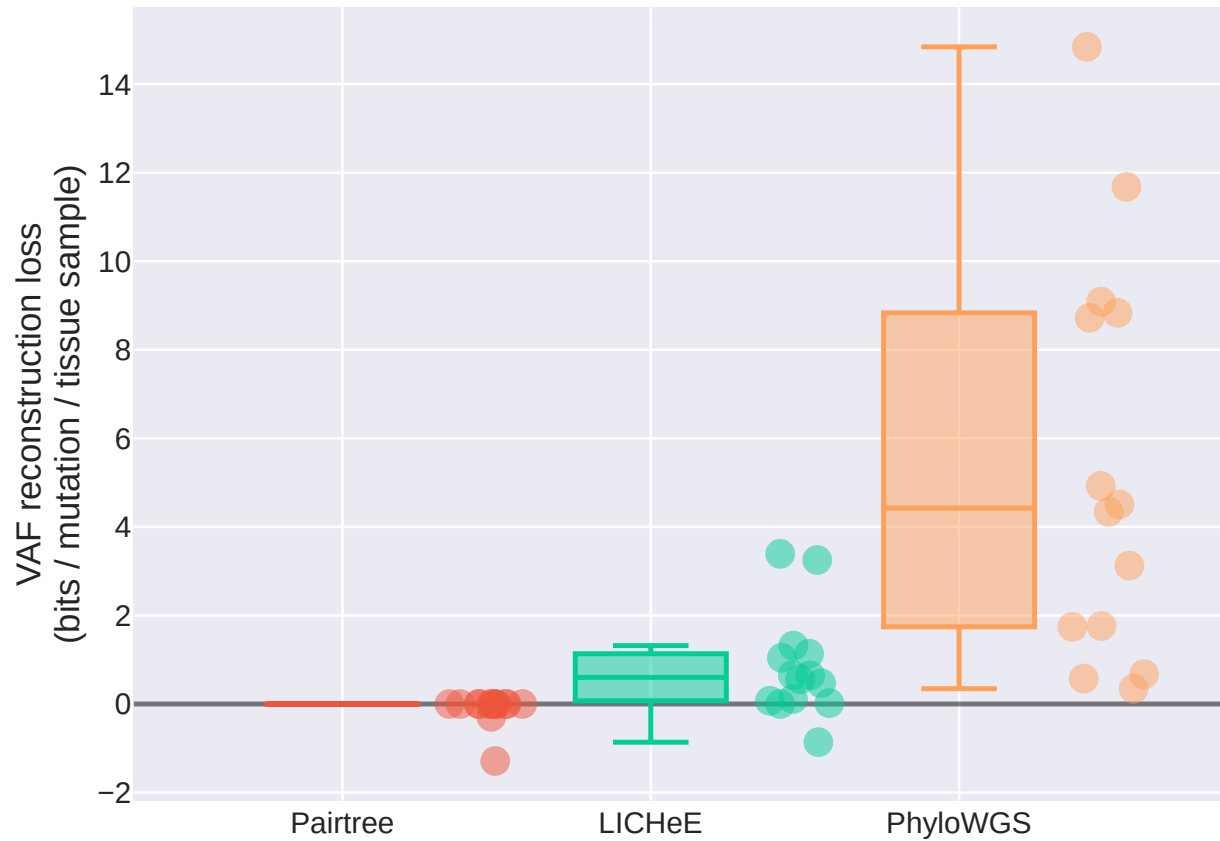


Figure S3: **Untruncated VAF reconstruction losses on 14 B-ALL datasets.** These results are the same as in Fig. 5, but without axis truncation. As in the truncated plots, results reflect each method's performance on the subset of datasets where it succeeded in running.

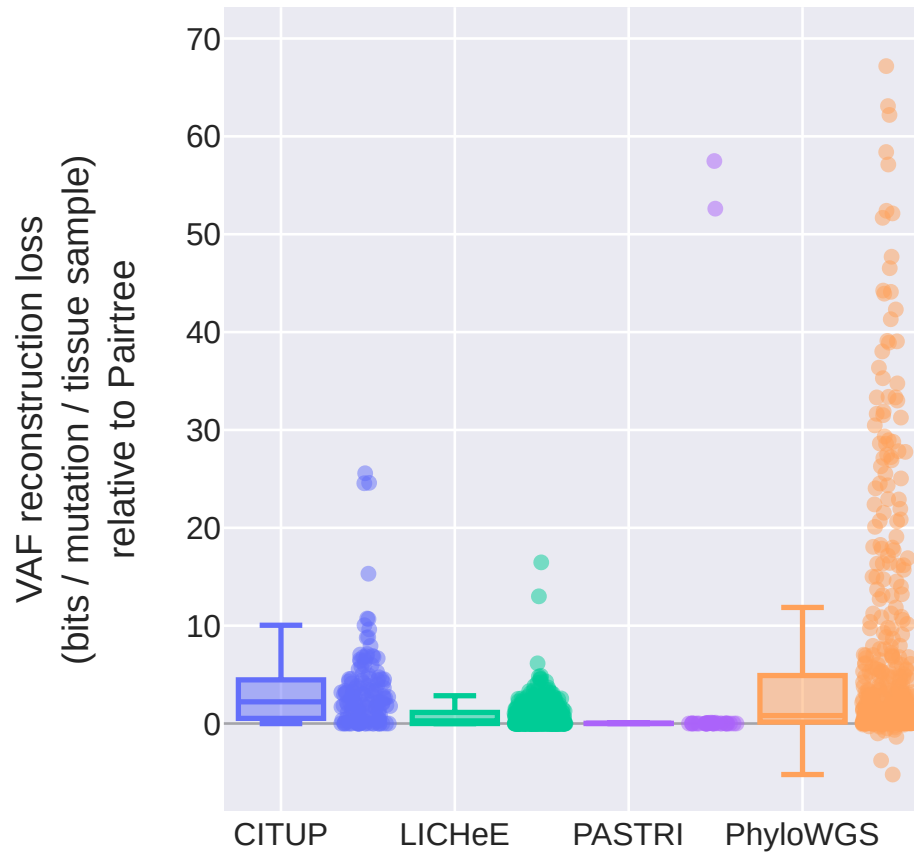
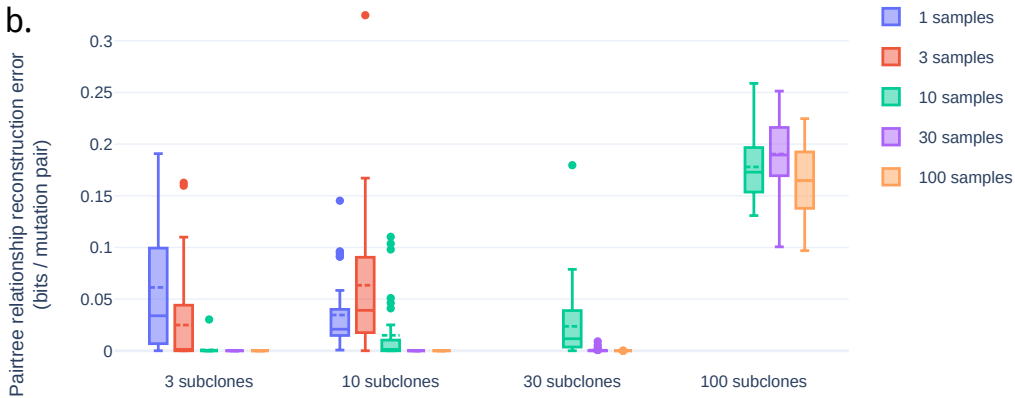


Figure S4: **VAF reconstruction loss of each method relative to Pairtree.** Each point represents a method's VAF reconstruction loss on a simulated dataset relative to Pairtree, with positive values indicating worse error. As each method failed on different simulations (Fig. 3a), values are reported only on datasets where a method produced a result.

a.



b.



c.

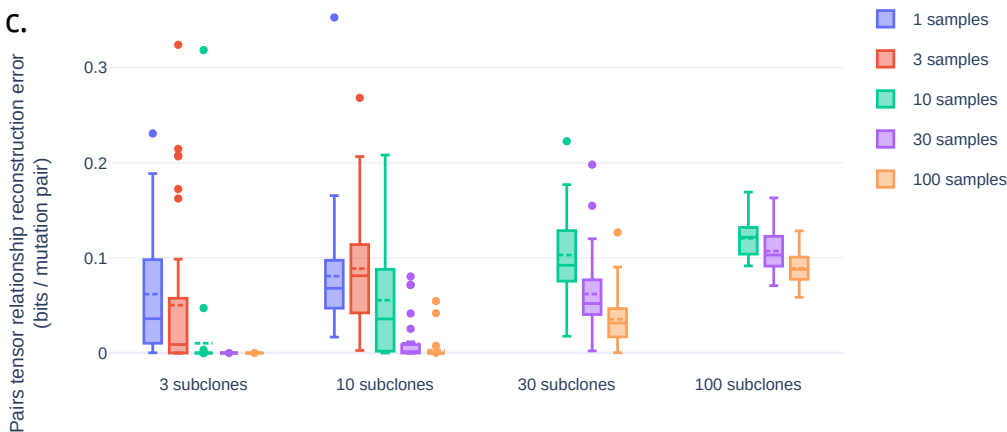


Figure S5: **Pairtree's performance on different numbers of subclones and cancer samples.** **a.** Pairtree's VAF reconstruction loss for each number of subclones and number of cancer samples. **b.** Pairtree's relationship reconstruction error for each number of subclones and number of cancer samples. **c.** Pairtree's Pairs Tensor's relationship reconstruction error for each number of subclones and number of cancer samples.

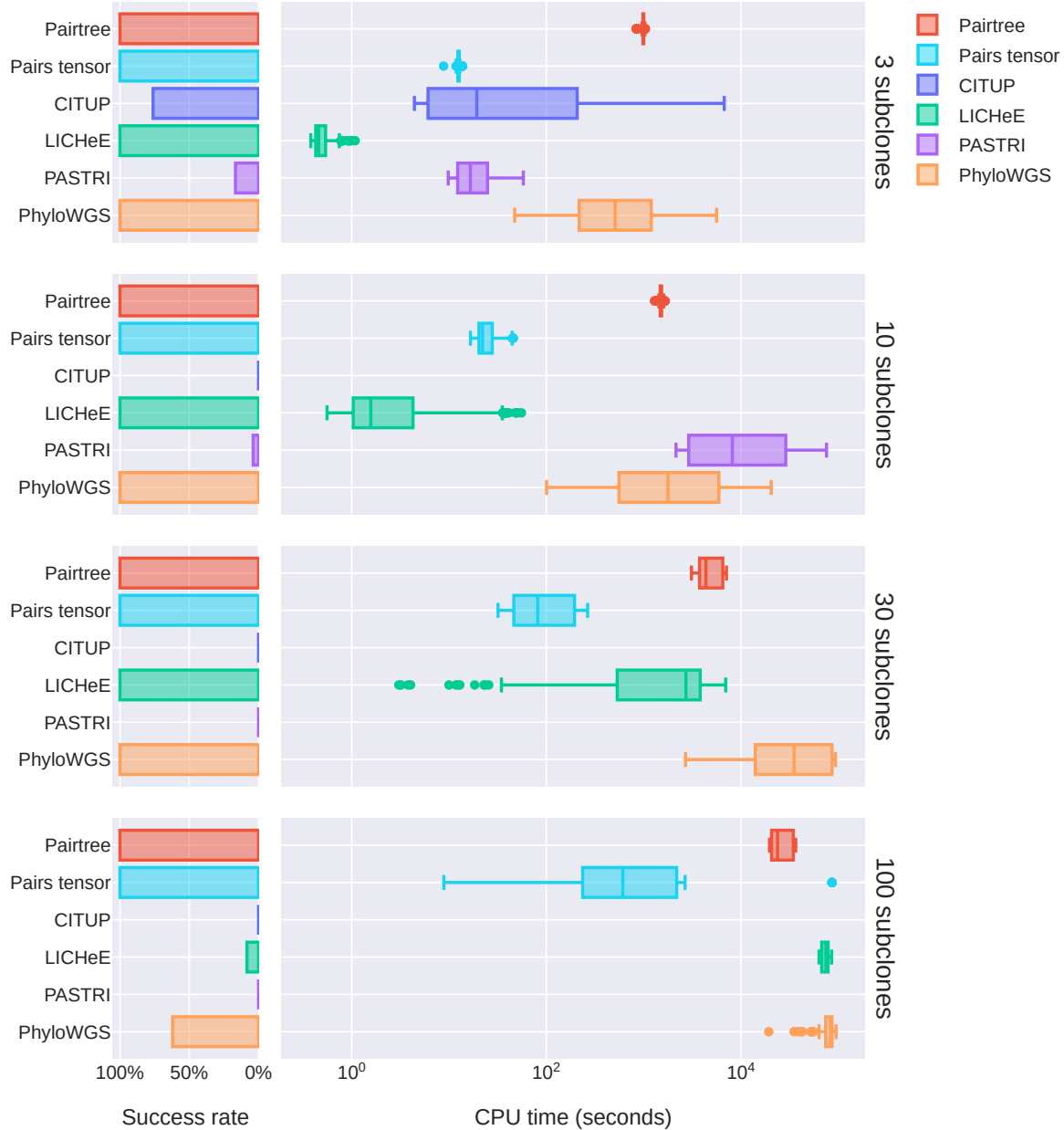


Figure S6: **Number of CPU seconds methods took to produce results.** Box mid-lines indicate medians. When using multiple CPU cores, these numbers can be much higher than elapsed wall-clock time (Fig. S7). Results for each method reflect only its performance on the datasets where it could produce a result.

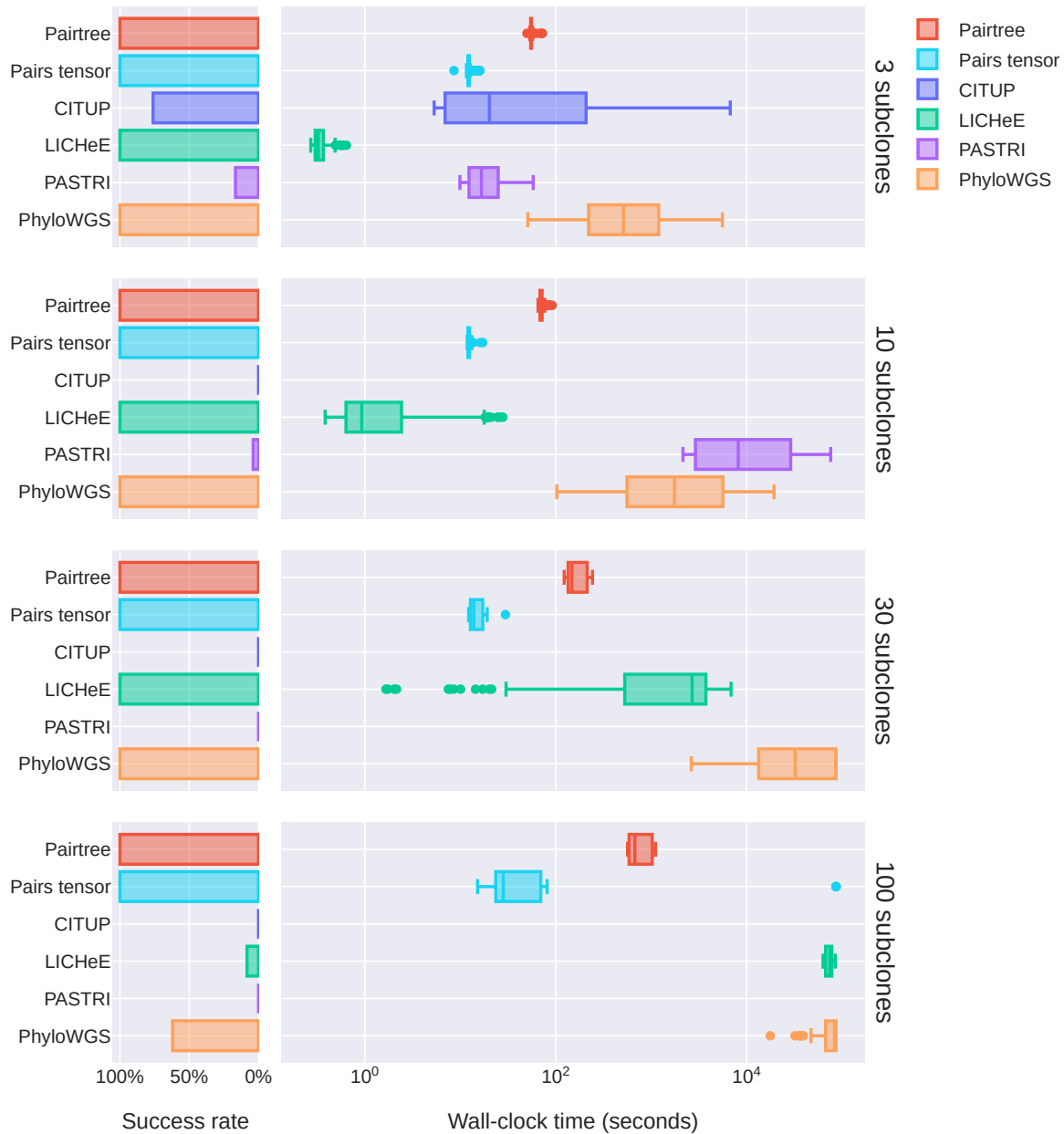


Figure S7: **Elapsed wall-clock seconds methods took to produce results.** Box mid-lines indicate medians. When using multiple CPU cores, these numbers can be much lower than the number of CPU seconds consumed (Fig. S6). Results for each method reflect only its performance on the datasets where it could produce a result.

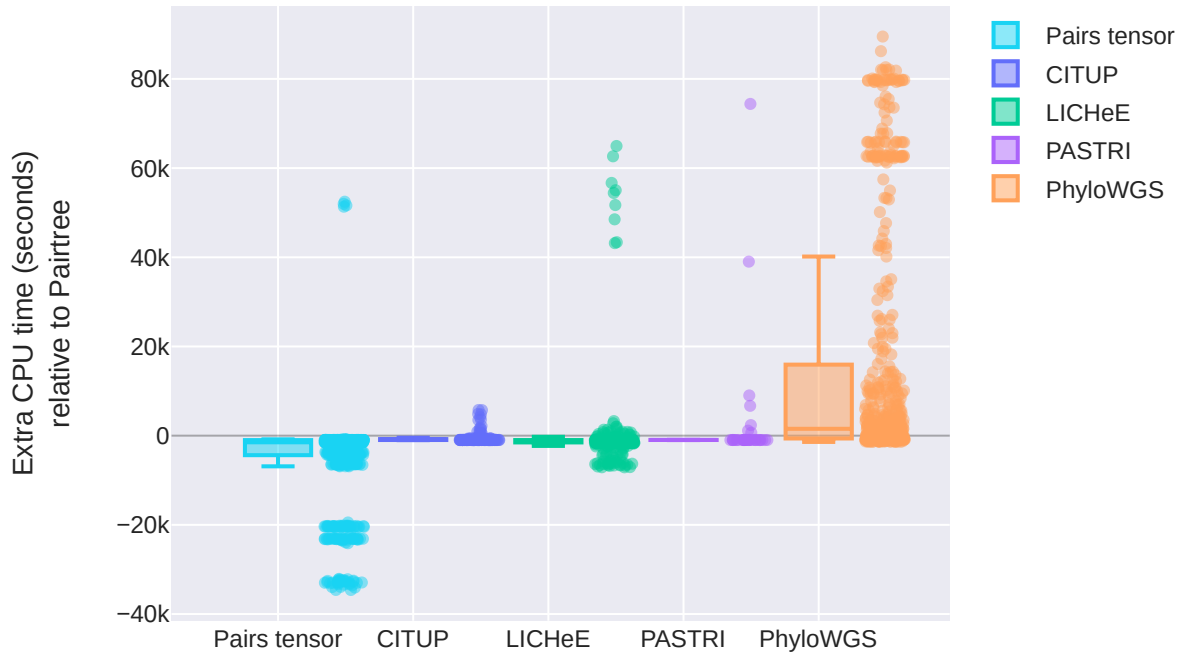


Figure S8: **Number of CPU seconds each method took to produce results relative to Pairtree.** Each point indicates the number of additional CPU seconds a method took on a dataset relative to Pairtree on that dataset. Points below zero indicate a method took less time than Pairtree on those datasets.



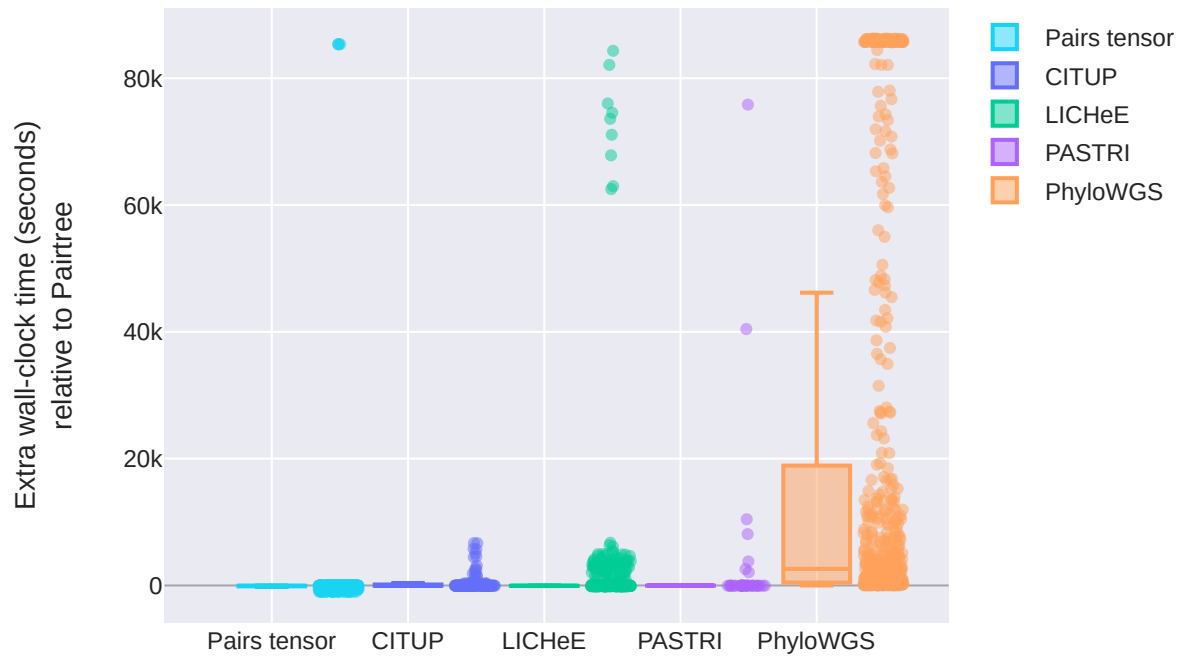
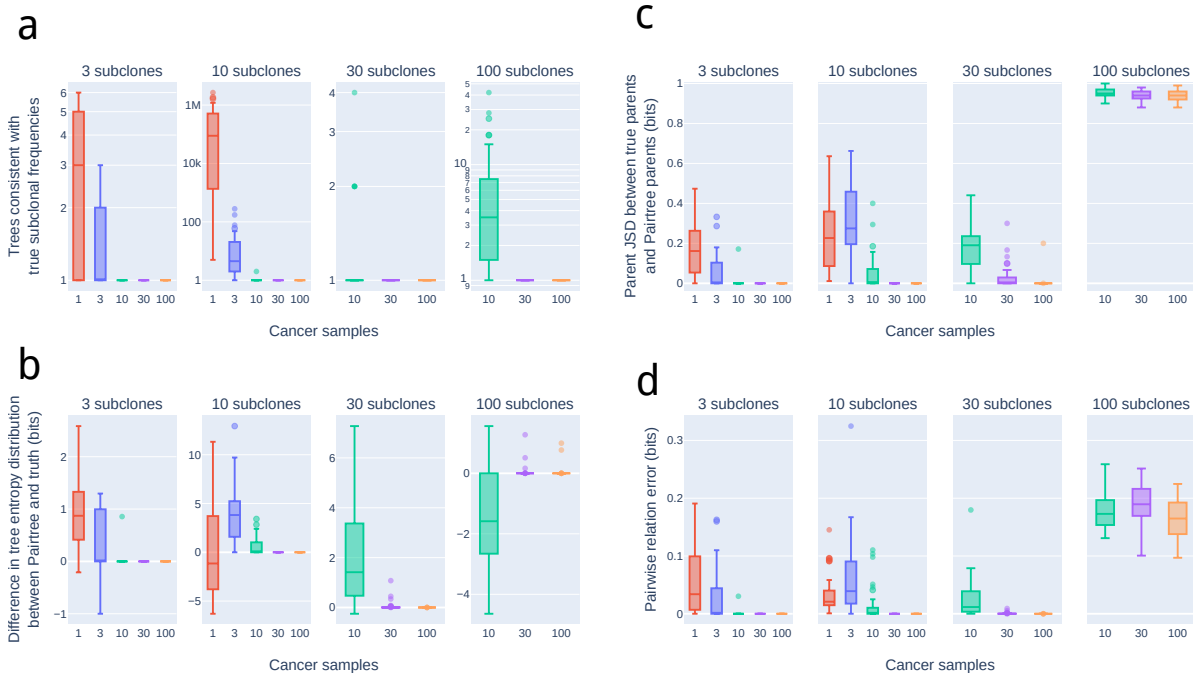


Figure S9: **Elapsed wall-clock seconds each method took to produce results relative to Pairtree.** Each point indicates the number of additional wall-clock seconds a method took on a dataset relative to Pairtree on that dataset. Points below zero indicate a method took less time than Pairtree on those datasets.



**Figure S10: Characteristics of the distributions over possible trees for the 576 simulated clone tree reconstruction problems.** Mid-lines in box plots indicate medians. **a.** Regardless of the number of subclones, with one cancer sample there are usually multiple trees consistent with the true subclonal frequencies. The highest median number of true trees (88,860) is reached for 10-subclone, single-sample reconstructions problems. Given ten or more samples, the tree becomes highly constrained, and there is usually only a single consistent tree. **b.** The entropies of the Pairee-recovered tree distribution and true tree distribution reflect how many high-confidence trees Pairee recovers relative to the number of possible trees. In general, Pairee recognizes when the true tree is highly constrained, and returns only one high-confidence tree. **c.** For a simulated dataset, a distribution over possible trees induces a distribution over parent choice for every population represented in the tree. Shown are the joint Jensen-Shannon divergence between parent distributions for Pairee relative to truth for each simulated dataset, normalized to the number of subclones in each tree. These divergences range between zero and one, with small values indicating that parent choices are nearly always correct. For a given number of subclones, Pairee generally exhibits lower divergences with more cancer samples, indicating it was able to use the information provided by those samples to improve its solution set. **d.** Relationship reconstruction errors show that, even when the parents chosen for subclones are sometimes incorrect (panel c), the relationship reconstructions can be more accurate. This is the same information as presented in Fig. 3b, but partitioned by number of cancer samples.

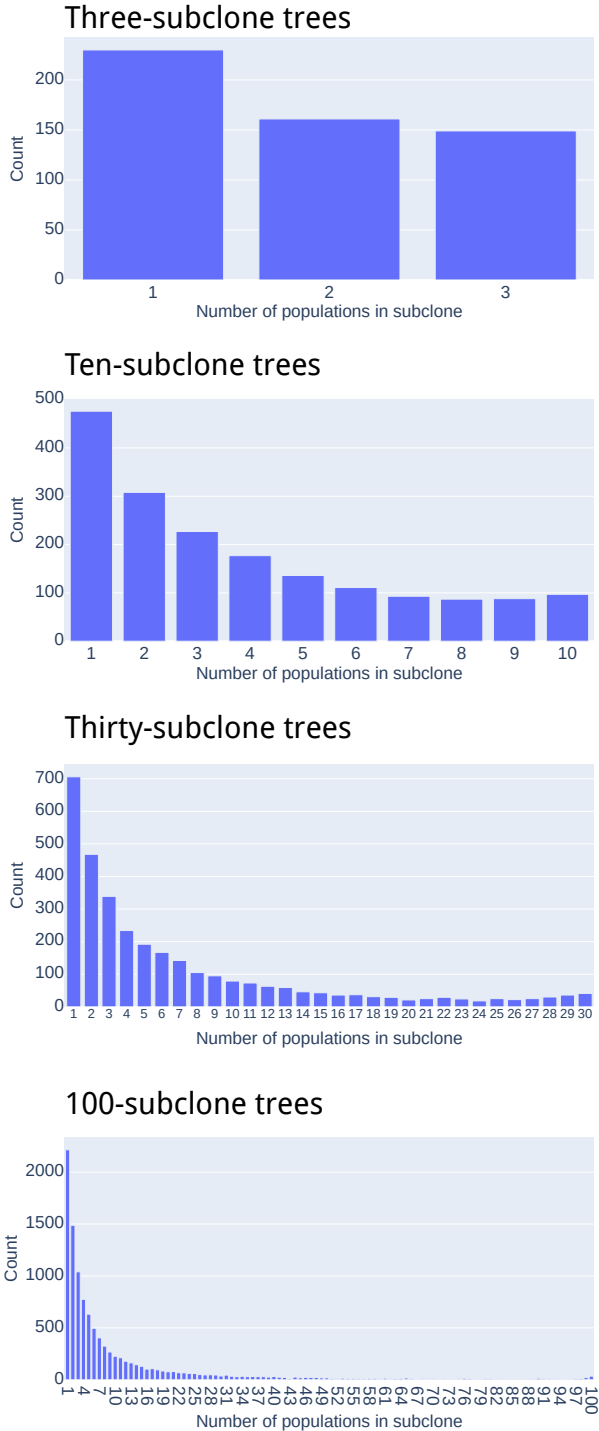
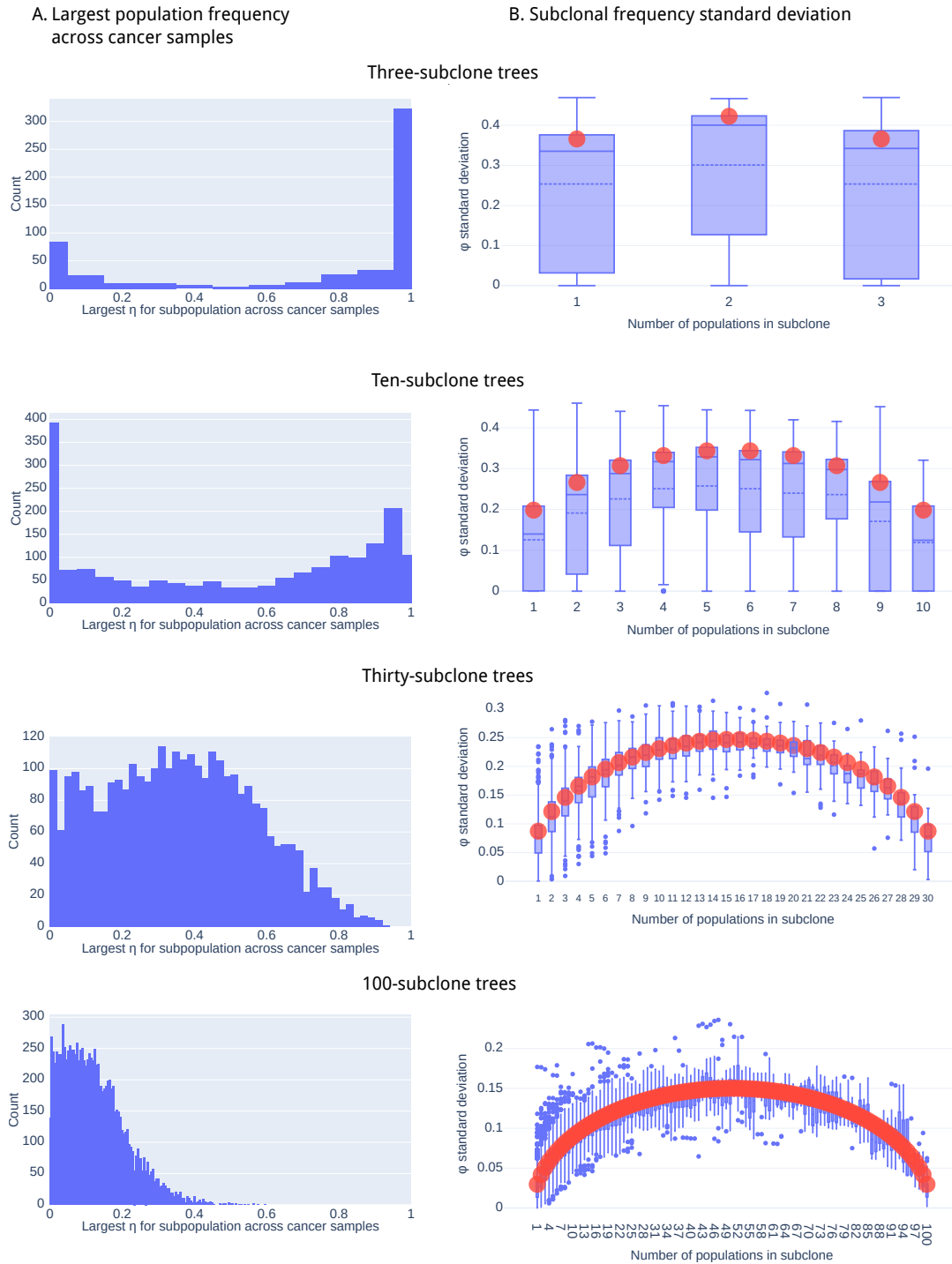


Figure S11: **Prevalence of different subclone sizes within simulated trees.** Subclone size indicates the number of subpopulations present within a subclone, reflecting the number of subpopulations that are descendants of the subpopulation that initiated the subclone.



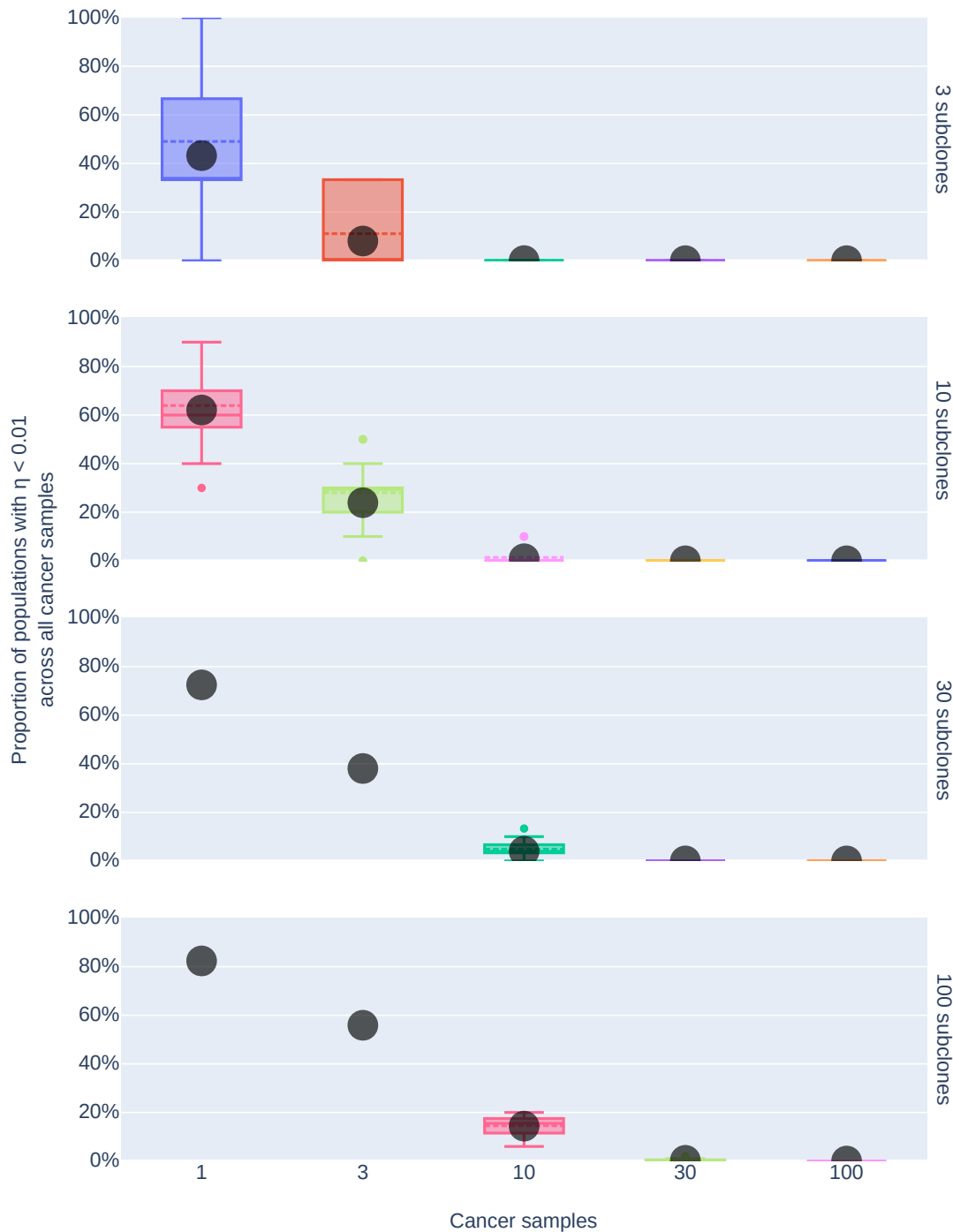


Figure S13: **Proportion of populations with small population frequencies in all cancer samples.** Proportion of populations  $k$  with population frequencies  $\eta_{ks} < 1\%$  across all cancer samples  $s$ . Box plots show the empirical proportions measured in (noise-free) simulated data, with solid line indicating the median and dashed line showing the mean. Grey circles show the predicted proportions derived from Dirichlet distribution properties.



Figure S14: **Probability that sub-trees will consist entirely of populations with small frequencies in all cancer samples.** Probability that sub-tree containing given number of populations will have population frequencies  $\eta_{k,s} < 1\%$  for all populations  $k$  in the sub-tree across all cancer samples  $s$ , computed using properties of Dirichlet distribution. A sub-tree consists of a subset of nodes from the full-tree and all edges between those nodes. By this definition, all subclones are sub-trees, but a sub-tree need not be a subclone.

## 1839 References

- 1840 1. Dentre, S. C. *et al.* Pervasive intra-tumour heterogeneity and subclonal selection across cancer types.  
1841 *Accepted at Cell* (2021).
- 1842 2. Hanahan, D. & Weinberg, R. a. Hallmarks of cancer: the next generation. *Cell* **144**, 646–74. ISSN:  
1843 1097-4172. <http://www.ncbi.nlm.nih.gov/pubmed/21376230> (Mar. 2011).
- 1844 3. Gerstung, M. *et al.* The evolutionary history of 2,658 cancers. *Nature* **578**, 122–128. ISSN: 1476-4687.  
1845 <https://www.nature.com/articles/s41586-019-1907-7> (2020) (Feb. 2020).
- 1846 4. Espiritu, S. M. G. *et al.* The Evolutionary Landscape of Localized Prostate Cancers Drives Clinical  
1847 Aggression. *Cell* **173**, 1003–1013.e15. ISSN: 0092-8674. [http://www.sciencedirect.com/science/](http://www.sciencedirect.com/science/article/pii/S009286741830309X)  
1848 [article/pii/S009286741830309X](http://www.sciencedirect.com/science/article/pii/S009286741830309X) (2018) (May 3, 2018).
- 1849 5. Jamal-Hanjani, M. *et al.* Tracking the Evolution of Non–Small-Cell Lung Cancer. *New England*  
1850 *Journal of Medicine* **376**, 2109–2121. ISSN: 0028-4793, 1533-4406. [http://www.nejm.org/doi/10.](http://www.nejm.org/doi/10.1056/NEJMoa1616288)  
1851 [1056/NEJMoa1616288](http://www.nejm.org/doi/10.1056/NEJMoa1616288) (2018) (June 2017).
- 1852 6. Gudem, G. *et al.* The evolutionary history of lethal metastatic prostate cancer. *Nature* **520**, 353–  
1853 357. ISSN: 1476-4687 (Apr. 16, 2015).
- 1854 7. Sakamoto, H. *et al.* Evolutionary Origins of Recurrent Pancreatic Cancer. *bioRxiv*, 811133. [https:](https://www.biorxiv.org/content/10.1101/811133v1)  
1855 [//www.biorxiv.org/content/10.1101/811133v1](https://www.biorxiv.org/content/10.1101/811133v1) (2019) (Oct. 31, 2019).
- 1856 8. Alves, J. M., Prado-López, S., Cameselle-Teijeiro, J. M. & Posada, D. Rapid evolution and biogeo-  
1857 graphic spread in a colorectal cancer. *Nature Communications* **10**. Number: 1 Publisher: Nature  
1858 Publishing Group, 5139. ISSN: 2041-1723. [https://www.nature.com/articles/s41467-019-](https://www.nature.com/articles/s41467-019-12926-8)  
1859 [12926-8](https://www.nature.com/articles/s41467-019-12926-8) (2020) (Nov. 13, 2019).
- 1860 9. Hu, Z. *et al.* Quantitative evidence for early metastatic seeding in colorectal cancer. *Nature Genetics*,  
1861 1. ISSN: 1546-1718. <https://www.nature.com/articles/s41588-019-0423-x> (2019) (June 17,  
1862 2019).
- 1863 10. Dobson, S. M. *et al.* Relapse-Fated Latent Diagnosis Subclones in Acute B Lineage Leukemia Are  
1864 Drug Tolerant and Possess Distinct Metabolic Programs. *Cancer Discovery*. Publisher: American  
1865 Association for Cancer Research Section: Research Articles. ISSN: 2159-8274, 2159-8290. [https :](https://cancerdiscovery.aacrjournals.org/content/early/2020/03/12/2159-8290.CD-19-1059)  
1866 [//cancerdiscovery.aacrjournals.org/content/early/2020/03/12/2159-8290.CD-19-1059](https://cancerdiscovery.aacrjournals.org/content/early/2020/03/12/2159-8290.CD-19-1059)  
1867 (2020) (Feb. 21, 2020).

- 1868 11. Hu, Z., Li, Z., Ma, Z. & Curtis, C. Multi-cancer analysis of clonality and the timing of systemic  
1869 spread in paired primary tumors and metastases. *Nature Genetics* **52**. Number: 7 Publisher: Nature  
1870 Publishing Group, 701–708. ISSN: 1546-1718. [https://www.nature.com/articles/s41588-020-](https://www.nature.com/articles/s41588-020-0628-z)  
1871 0628-z (2020) (July 2020).
- 1872 12. Zahir, N., Sun, R., Gallahan, D., Gatenby, R. A. & Curtis, C. Characterizing the ecological and  
1873 evolutionary dynamics of cancer. *Nature Genetics* **52**. Number: 8 Publisher: Nature Publishing  
1874 Group, 759–767. ISSN: 1546-1718. <https://www.nature.com/articles/s41588-020-0668-4>  
1875 (2020) (Aug. 2020).
- 1876 13. Williams, M. J. *et al.* Quantification of subclonal selection in cancer from bulk sequencing data.  
1877 *Nature Genetics*, 1. ISSN: 1546-1718. <http://www.nature.com/articles/s41588-018-0128-6>  
1878 (2018) (May 28, 2018).
- 1879 14. Pogrebniak, K. L. & Curtis, C. Harnessing Tumor Evolution to Circumvent Resistance. *Trends in*  
1880 *Genetics* **34**, 639–651. ISSN: 01689525. [https://linkinghub.elsevier.com/retrieve/pii/](https://linkinghub.elsevier.com/retrieve/pii/S0168952518300921)  
1881 S0168952518300921 (2018) (Aug. 2018).
- 1882 15. Jiang, Y., Qiu, Y., Minn, A. J. & Zhang, N. R. Assessing intratumor heterogeneity and tracking  
1883 longitudinal and spatial clonal evolutionary history by next-generation sequencing. *Proceedings of*  
1884 *the National Academy of Sciences* **113**, E5528–E5537. ISSN: 0027-8424, 1091-6490. [http://www.](http://www.pnas.org/content/113/37/E5528)  
1885 [pnas.org/content/113/37/E5528](http://www.pnas.org/content/113/37/E5528) (2017) (Sept. 13, 2016).
- 1886 16. Malikic, S., McPherson, A. W., Donmez, N. & Sahinalp, C. S. Clonality inference in multiple tumor  
1887 samples using phylogeny. *Bioinformatics* **31**, 1349–1356. ISSN: 1460-2059, 1367-4803. [https://](https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btv003)  
1888 [academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btv003](https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btv003)  
1889 (2019) (May 1, 2015).
- 1890 17. Malikic, S., Jahn, K., Kuipers, J., Sahinalp, S. C. & Beerenwinkel, N. Integrative inference of sub-  
1891 clonal tumour evolution from single-cell and bulk sequencing data. *Nature Communications* **10**.  
1892 Number: 1 Publisher: Nature Publishing Group, 2750. ISSN: 2041-1723. [https://www.nature.com/](https://www.nature.com/articles/s41467-019-10737-5)  
1893 [articles/s41467-019-10737-5](https://www.nature.com/articles/s41467-019-10737-5) (2020) (June 21, 2019).
- 1894 18. Deshwar, A. G., Vembu, S. & Morris, Q. Comparing nonparametric Bayesian tree priors for clonal  
1895 reconstruction of tumors. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*,  
1896 20–31. ISSN: 2335-6936 (2015).
- 1897 19. Popic, V. *et al.* Fast and scalable inference of multi-sample cancer lineages. *Genome Biology* **16**.  
1898 ISSN: 1465-6906. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4501097/> (2017) (2015).



- 1899 20. Satas, G. & Raphael, B. J. Tumor phylogeny inference using tree-constrained importance sampling.  
1900 *Bioinformatics* **33**, i152–i160. ISSN: 1367-4803. [https://academic.oup.com/bioinformatics/  
1901 article/33/14/i152/3953987/Tumor-phylogeny-inference-using-tree-constrained](https://academic.oup.com/bioinformatics/article/33/14/i152/3953987/Tumor-phylogeny-inference-using-tree-constrained) (2017)  
1902 (July 15, 2017).
- 1903 21. Salcedo, A. *et al.* A community effort to create standards for evaluating tumor subclonal recon-  
1904 struction. *Nature Biotechnology* **38**. Number: 1 Publisher: Nature Publishing Group, 97–107. ISSN:  
1905 1546-1696. <https://www.nature.com/articles/s41587-019-0364-z> (2020) (Jan. 2020).
- 1906 22. Dentre, S. C., Wedge, D. C. & Van Loo, P. Principles of Reconstructing the Subclonal Architecture  
1907 of Cancers. *Cold Spring Harbor Perspectives in Medicine*, a026625 (2017).
- 1908 23. Kuipers, J., Jahn, K., Raphael, B. J. & Beerenwinkel, N. Single-cell sequencing data reveal widespread  
1909 recurrence and loss of mutational hits in the life histories of tumors. *Genome Research* **27**, 1885–  
1910 1894. ISSN: 1088-9051. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5668945/> (2020)  
1911 (Nov. 2017).
- 1912 24. Singer, J., Kuipers, J., Jahn, K. & Beerenwinkel, N. Single-cell mutation identification via phylo-  
1913 genetic inference. *Nature Communications* **9**, 5144. ISSN: 2041-1723. [https://www.nature.com/  
1914 articles/s41467-018-07627-7](https://www.nature.com/articles/s41467-018-07627-7) (2020) (Dec. 4, 2018).
- 1915 25. Bonizzoni, P., Ciccolella, S., Della Vedova, G. & Soto, M. Does relaxing the infinite sites assumption  
1916 give better tumor phylogenies? An ILP-based comparative approach. *bioRxiv*. [http://biorxiv.  
1917 org/lookup/doi/10.1101/227801](http://biorxiv.org/lookup/doi/10.1101/227801) (2019) (Dec. 3, 2017).
- 1918 26. Ciccolella, S. *et al.* Inferring Cancer Progression from Single-cell Sequencing while Allowing Mutation  
1919 Losses. *bioRxiv*. <http://biorxiv.org/lookup/doi/10.1101/268243> (2019) (Apr. 13, 2018).
- 1920 27. Satas, G., Zaccaria, S., Mon, G. & Raphael, B. J. SCARLET: Single-Cell Tumor Phylogeny Inference  
1921 with Copy-Number Constrained Mutation Losses. *Cell Systems* **10**, 323–332.e8. ISSN: 2405-4712.  
1922 <http://www.sciencedirect.com/science/article/pii/S2405471220301150> (2020) (Apr. 22,  
1923 2020).
- 1924 28. Roth, A. *et al.* PyClone: statistical inference of clonal population structure in cancer. *Nature methods*  
1925 (2014).
- 1926 29. Miller, C. A. *et al.* SciClone: Inferring Clonal Architecture and Tracking the Spatial and Temporal  
1927 Patterns of Tumor Evolution. *PLoS Computational Biology* **10** (ed Beerenwinkel, N.) e1003665. ISSN:

- 1553-734X. JSTOR: {PMC}4125065. <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4125065/>  
1928  
1929 (Aug. 2014).
- 1930 30. Gillis, S. & Roth, A. PyClone-VI: Scalable inference of clonal population structures using whole  
1931 genome data. *bioRxiv*. Publisher: Cold Spring Harbor Laboratory Section: New Results, 2020.08.31.276212.  
1932 <https://www.biorxiv.org/content/10.1101/2020.08.31.276212v1> (2020) (Sept. 1, 2020).
- 1933 31. Tarabichi, M. *et al.* A practical guide to cancer subclonal reconstruction from DNA sequencing.  
1934 *Nature Methods* **18**. Number: 2 Publisher: Nature Publishing Group, 144–155. ISSN: 1548-7105.  
1935 <https://www.nature.com/articles/s41592-020-01013-2> (2021) (Feb. 2021).
- 1936 32. Hastings, W. K. Monte Carlo Sampling Methods Using Markov Chains and Their Applications.  
1937 *Biometrika* **57**. Publisher: [Oxford University Press, Biometrika Trust], 97–109. ISSN: 0006-3444.  
1938 <https://www.jstor.org/stable/2334940> (2020) (1970).
- 1939 33. Jahn, K., Kuipers, J. & Beerenwinkel, N. Tree inference for single-cell data. *Genome Biology* **17**,  
1940 86. ISSN: 1474-760X. <http://dx.doi.org/10.1186/s13059-016-0936-x> (2016) (2016).
- 1941 34. Jia, B., Ray, S., Safavi, S. & Bento, J. Efficient Projection onto the Perfect Phylogeny Model.  
1942 *arXiv:1811.01129 [cs]*. arXiv: 1811.01129. <http://arxiv.org/abs/1811.01129> (2018) (Nov. 2,  
1943 2018).
- 1944 35. Sundermann, L. K., Wintersinger, J., Rättsch, G., Stoye, J. & Morris, Q. Reconstructing tumor  
1945 evolutionary histories and clone trees in polynomial-time with SubMARine. *PLOS Computational*  
1946 *Biology* **17**. Publisher: Public Library of Science, 1–28. <https://doi.org/10.1371/journal.pcbi.1008400> (2021).  
1947
- 1948 36. Deshwar, A. G. *et al.* PhyloWGS: Reconstructing subclonal composition and evolution from whole  
1949 genome sequencing of tumors. *Genome Biology* **16**, 35 (2015).
- 1950 37. Satas, G. & Raphael, B. *PASTRI source code* Aug. 16, 2019. [https://github.com/raphael-](https://github.com/raphael-group/PASTRI)  
1951 [group/PASTRI](https://github.com/raphael-group/PASTRI) (2020).
- 1952 38. Hudson, R. R. & Kaplan, N. L. Statistical Properties of the Number of Recombination Events  
1953 in the History of a Sample of DNA Sequences. *Genetics* **111**, 147–164. ISSN: 0016-6731. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1202594/>  
1954 (2019) (Sept. 1985).
- 1955 39. Campbell, P. J. *et al.* Pan-cancer analysis of whole genomes. *Nature* **578**. Number: 7793 Publisher:  
1956 Nature Publishing Group, 82–93. ISSN: 1476-4687. [https://www.nature.com/articles/s41586-](https://www.nature.com/articles/s41586-020-1969-6)  
1957 [020-1969-6](https://www.nature.com/articles/s41586-020-1969-6) (2021) (Feb. 2020).

- 1958 40. Riedmiller, M. & Braun, H. *A direct adaptive method for faster backpropagation learning: The*  
1959 *RPROP algorithm in IEEE international conference on neural networks* (1993), 586–591.
- 1960 41. Hinton, G. *Neural Networks for Machine Learning: Lecture 6a: Overview of min-batch gradient*  
1961 *descent*. 2014. [https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture\\_slides\\_lec6.](https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf)  
1962 [pdf](https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf).
- 1963 42. Bei, J. & Bento, J. *libprojectppm source code* original-date: 2018-10-26T22:41:45Z. Apr. 8, 2019.  
1964 <https://github.com/bentoayr/Efficient-Projection-onto-the-Perfect-Phylogeny-Model>.
- 1965 43. Gusfield, D. Efficient algorithms for inferring evolutionary trees. *Networks* **21**, 19–28. ISSN: 1097-  
1966 0037. <http://onlinelibrary.wiley.com/doi/10.1002/net.3230210104/abstract> (2017)  
1967 (Jan. 1, 1991).
- 1968 44. El-Kebir, M., Oesper, L., Acheson-Field, H. & Raphael, B. J. Reconstruction of clonal trees and  
1969 tumor composition from multi-sample sequencing data. *Bioinformatics (Oxford, England)* **31**, i62–  
1970 70. ISSN: 1367-4811 (June 15, 2015).
- 1971 45. Stewart, C. A. *et al.* Single-cell analyses reveal increased intratumoral heterogeneity after the onset  
1972 of therapy resistance in small-cell lung cancer. *Nature Cancer* **1**. Number: 4 Publisher: Nature  
1973 Publishing Group, 423–436. ISSN: 2662-1347. [https://www.nature.com/articles/s43018-019-](https://www.nature.com/articles/s43018-019-0020-z)  
1974 [0020-z](https://www.nature.com/articles/s43018-019-0020-z) (2020) (Apr. 2020).
- 1975 46. Miles, L. A. *et al.* Single cell mutational profiling delineates clonal trajectories in myeloid malignan-  
1976 cies. *bioRxiv*. Publisher: Cold Spring Harbor Laboratory Section: New Results, 2020.02.07.938860.  
1977 <https://www.biorxiv.org/content/10.1101/2020.02.07.938860v1> (2020) (Feb. 9, 2020).
- 1978 47. Lähnemann, D. *et al.* Eleven grand challenges in single-cell data science. *Genome Biology* **21**, 31.  
1979 ISSN: 1474-760X. <https://doi.org/10.1186/s13059-020-1926-6> (2020) (Feb. 7, 2020).