Kozlov et al. Genome Biology (2022) 23:37
https://doi.org/10.1186/s13059-021-02583-w

Genome Biology

METHOD Open Access

CellPhy: accurate and fast probabilistic inference of single-cell phylogenies from scDNA-seq data

Alexey Kozlov[1,5], Joao M. Alves[2,3,4], Alexandros Stamatakis[1] and David Posada[2,3,4*]

* Correspondence: dposada@uvigo.es
Alexey Kozlov and Joao M. Alves shared first-authorship.
[2]CINBIO, Universidade de Vigo, 36310 Vigo, Spain
[3]Department of Biochemistry, Genetics, and Immunology, Universidade de Vigo, 36310 Vigo, Spain
Full list of author information is available at the end of the article

Abstract

We introduce CellPhy, a maximum likelihood framework for inferring phylogenetic trees from somatic single-cell single-nucleotide variants. CellPhy leverages a finite-site Markov genotype model with 16 diploid states and considers amplification error and allelic dropout. We implement CellPhy into RAxML-NG, a widely used phylogenetic inference package that provides statistical confidence measurements and scales well on large datasets with hundreds or thousands of cells. Comprehensive simulations suggest that CellPhy is more robust to single-cell genomics errors and outperforms state-of-the-art methods under realistic scenarios, both in accuracy and speed. CellPhy is freely available at https://github.com/amkozlov/cellphy.

Keywords: Somatic phylogenetics, Genotype evolution, Single-cell genomics, Single-cell phylogenies, Cell lineage trees

Background

The study of single cells is revolutionizing biology, unveiling unprecedented levels of genomic and phenotypic heterogeneity within otherwise seemingly homogeneous tissues [1–6]. Understanding this somatic mosaicism has applications in multiple areas of biology due to its intrinsic connection to development, aging, and disease [7, 8]. However, the analysis of single-cell genomic data is not devoid of challenges [5, 9], including the development of more integrative, scalable, and biologically realistic models of somatic evolution that can handle the inherent noise of single-cell data—mainly amplification error and allelic dropout (ADO) [10]. Understanding how somatic cells evolve is one of the main applications of single-cell technologies. In particular, the reconstruction of cell phylogenies from single-cell DNA sequencing (scDNA-seq) can help us understand the mode and tempo of cell diversification and its mechanisms [11]. Thus, cell phylogenies can help disentangle key events in the ontogeny of differentiated cell types [12] and can be used to decipher human development at a remarkable resolution [13–17]. When geographical information is available, cell (and clonal) genealogies can

inform us about somatic expansions and cell migrations, which are of great relevance, for example, to understand tumor growth and metastasis [18–22].

Several methods have been proposed to reconstruct cell phylogenies from scDNA-seq data using single-nucleotide variants (SNVs) [23, 24]. OncoNEM [25] implements a nested-effects likelihood model to correct for observational noise, plus a simple heuristic search that maximizes the likelihood of the data across tree space. It reconstructs a tree of clones and mutations by assigning cells to the clones. OncoNEM assumes an infinite-site mutation (ISM) model, and as we will see below, it is very slow and can only analyze small datasets. Among the most popular, SCITE [26] assumes an infinite-site mutation (ISM) model and uses Markov Chain Monte Carlo (MCMC) to sample likelihoods/posterior probabilities. Apart from the cell phylogeny, SCITE can estimate a mutation tree to which it attaches the cells a posteriori. Conveniently, it can also infer false-positive (FP) and false-negative (FN) error rates from the data. infSCITE [27] extends SCITE to consider cell doublets, test the ISM's validity, and learn the FP rate from panel sequencing data. SiFit [28] implements a Markov finite-site evolution model and a heuristic ML tree search algorithm. It can also estimate FP and FN error rates, and in the reported simulations, outperforms OncoNEM and SCITE in terms of speed and accuracy. SCIPhI [29] jointly estimates the cell genotypes and their phylogenetic relationships, working directly with the read counts and considering amplification/sequencing errors, ADO, and loss of heterozygosity. SPhyR [30] and SASC [31] are ISM methods that exploit Dollo parsimony, which assumes that a mutation can be gained only once but lost multiple times. SCARLET [32] is another method that explicitly considers losses, uses a statistical model for the sequencing data, and leverages copy number profiles. More recently, ScisTree [33] implements an ISM maximum likelihood method whose input is binary/ternary genotype probabilities. In the simulations performed by the author, ScisTree's accuracy is similar to SCIPhI and SCITE, but it is much faster. Related approaches for single-cell SNV data include methods focused on cell clustering and clonal phylogeny [34–41], phylogenetic model selection [42, 43], mutation ordering [44], genotype correction [45], or longitudinal comparisons [46]. Methods also exist for inferring phylogenies from single-cell copy number variants [47], but these are out of the scope of this study.

All methods mentioned above for the specific problem of cell phylogeny reconstruction with SNVs were implemented de novo, yet they use relatively simple evolutionary models. Indeed, developing biologically realistic and yet computationally tractable models is one of the current challenges for single-cell phylogenetics [9]. Statistical phylogenetics is a well-developed field, with many sophisticated evolutionary models implemented into efficient computational programs. We reasoned that we could leverage such a framework to obtain more accurate somatic cell phylogenies. Thus, we implemented a novel, dedicated single-cell somatic SNV evolution model in an existing, successful framework for statistical phylogenetics, RAxML-NG [48]. We named the resulting software CellPhy. The main focus of CellPhy is the complete reconstruction of cell phylogenies, which can be the basis of subsequent evolutionary analyses. Our computer simulations and the analysis of empirical datasets show that CellPhy can reconstruct accurate single-cell SNV phylogenies across different biological scenarios and is substantially faster than the competing likelihood-based methods. We also evaluated a maximum parsimony-based tool TNT [49], an extremely fast approach for

phylogenetic reconstruction. Although TNT performed very well in error-free simulation scenarios, its accuracy quickly degraded in the presence of the typical biases observed in scDNA-seq data.

Results

CellPhy

We developed a probabilistic model for the phylogenetic analysis of single-cell diploid genotypes inferred from scDNA-seq experiments, called CellPhy. For tractability, we focused on single nucleotide variants (SNVs), arguably the most common genetic data obtained from somatic tissues nowadays. Current SNV evolutionary models for single-cells only consider the absence/presence of mutations regardless of the nucleotides involved [25, 26, 28]. Thus, they deal with at most a ternary state space where genotypes can only have 0, 1, or 2 mutant alleles (normal, heterozygous, or homozygous mutant, respectively). Instead, here we consider changes among all possible 16 phased DNA genotypes Γ = {A|A, A|C, A|G, A|T, C|A, C|C, C|G, C|T, G|A, G|C, G|G, G|T, T|A, T|C, T|G, T|T}, by extending the well-established finite-site continuous-time general-time-reversible Markov model of DNA sequence evolution with four states (GTR) [50] to 16 states. We named this new model GT16. In Fig. S1, we summarize a few characteristics of this model.

Because somatic evolution proceeds mainly by mitosis, where both daughter cells receive the same set of chromosomes and recombination can be safely ignored, a unique cell history is recorded in the same way in the maternal and paternal chromosomes. Therefore, we do not need to know whether a mutation occurred in the maternal or paternal chromosome to infer the cell phylogeny. While the GT16 model in CellPhy considers phased genotypes, nowadays, the vast majority of the empirical scDNA-seq datasets are unphased due to technical limitations. Conveniently, the GT16 model can also work with unphased genotypes (10 states) simply by considering the ambiguity of the phase in heterozygotes (see the "Methods" section). Our simulations consist of unphased genotypes to represent current scDNA-seq data.

Importantly, single-cell genotypes can be very noisy, mainly due to biases during whole-genome amplification (WGA) [10]. While previous methods rely on observational error models based on FP and FN rate parameters, we built an error model with two free parameters, the ADO rate (δ), and the amplification/sequencing error rate (ε). Compared with previous implementations, the advantage of this parameterization is that it can incorporate plausible situations such as an amplification/sequencing error converting a homozygous mutant into a heterozygous genotype. Due to its low probability of occurrence, we discard the possibility of observing more than one amplification/sequencing error at a given site. Still, we allow for the presence of both ADO and amplification/sequencing error in a single genotype. Instead of using a genotype error model, CellPhy can also use the Phred-scaled genotype likelihoods provided by single-cell variant callers.

We assume that the evolutionary history of a sample of cells can be appropriately portrayed as an unrooted binary tree and that all SNVs evolve in the same way and independently of each other. Given a set of single-cell SNV genotypes (provided by the user as a matrix in FASTA or PHYLIP format, or as a standard VCF file), CellPhy

leverages its error and genotype models to compute the tree likelihood as a product of the independent probabilities across SNVs, using the standard Felsenstein pruning algorithm [51, 52]. Conveniently, CellPhy does not need to assume any particular genotypic configuration at the root, like other programs. For example, SiFit assumes that the root of the tree is homozygous for the reference allele at all sites. Instead, the CellPhy tree can be easily rooted a posteriori using a particular set of cells as an outgroup (see [51]). For example, if we study tumor cells, the outgroup could be one or more healthy cells.

We implemented CellPhy's phylogenetic model in RAxML-NG [48], a popular maximum likelihood (ML) framework in organismal phylogenetics. Therefore, to obtain ML estimates of the model parameters (substitution rates, ADO, and amplification/sequencing errors) and the cell tree, CellPhy leverages the optimization routines and tree search strategies of RAxML [53] and RAxML-NG [48]. The latter, for example, is known to work particularly well on large trees [54]. The fact that CellPhy exploits RAxML-NG allows it to also calculate confidence values for the inner tree branches using either the standard [55] or transfer [56] bootstrap (BS) techniques. Moreover, CellPhy can perform ancestral state reconstruction [57] to obtain ancestral ML genotypes and map mutations onto the branches of the ML tree. CellPhy is freely available, together with documentation, tutorials, and example data at https://github.com/amkozlov/cellphy.

Validation and benchmarking

We benchmarked CellPhy against eight alternative methods for inferring cell phylogenies using SNV data (TNT, OncoNEM, SASC, SPhyR, infSCITE, SiFit, SCIPhI, and ScisTree), under multiple scenarios (Table S1). These methods represent various approaches regarding the ISM assumption, the input data, the error model, the optimization criterion, the search algorithm, or the statistical paradigm. We do not consider other clustering methods that do not explicitly reconstruct cell phylogenies or require additional data (e.g., bulk sequencing data). Since SCARLET requires copy-number profile data, and single-cell CNVs are out of the scope of this study, we did not include this method in the comparison.

Initial assessment of methods

We carried out an initial performance assessment to understand which methods were worth evaluating in detail. Three methods, OncoNEM, SASC, and SPhyR, already showed poor phylogenetic accuracy under ideal conditions (no errors) or relatively low noise levels (Fig. S2). OncoNEM produced highly unresolved trees (i.e., 50 to 90% polytomies), with low phylogenetic accuracy under these (and other) simple scenarios. In addition, OncoNEM is extremely slow and cannot handle large data sets. Similarly, SASC showed poor phylogenetic accuracy, with many polytomies, and also extremely long running times. SPhyR, on the other hand, was more accurate (and faster) than OncoNEM and SASC, albeit clearly worse than the remaining competitor tools despite being executed with the "true" simulated error rates. Unfortunately, SPhyR suffered from memory corruption issues for most (simulated and empirical) datasets. Thus, we did not further evaluate OncoNEM, SASC, and SPhyR.

Simulation 1: infinite-site model and low number of SNVs ("target-ISM")

For a simple scenario with samples of 40 cells and 250–1000 SNVs evolved under an infinite-site mutation model, phylogenetic accuracy decreased rapidly for all methods with increasing levels of genotype error or ADO (Figs. 1, S3–S4). CellPhy was the most accurate method overall, although closely followed by SiFit and, to a lesser extent, by infSCITE, which performed worse when genotype errors were prevalent. TNT, a parsimony-based inference tool, was as accurate as CellPhy, SiFit, or infSCITE without ADO and genotype error, but worse otherwise. The genotype coding strategy ("keep," "remove," and "missing"; see the "Methods" section) influenced accuracy only when genotype errors were present, with "missing" and "keep" being better than "remove" (see Figs. S3-S4).

Simulation 2: finite-site model and large number of SNVs ("WGS-FSM")

When we simulated larger data sets (100 cells, ~2000 SNVs) under a finite-site model of DNA evolution, overall phylogenetic accuracy increased for all methods. As before, all methods performed worse with higher levels of ADO or genotype error (Fig. 2). CellPhy was again most accurate overall, mainly when the data contained many

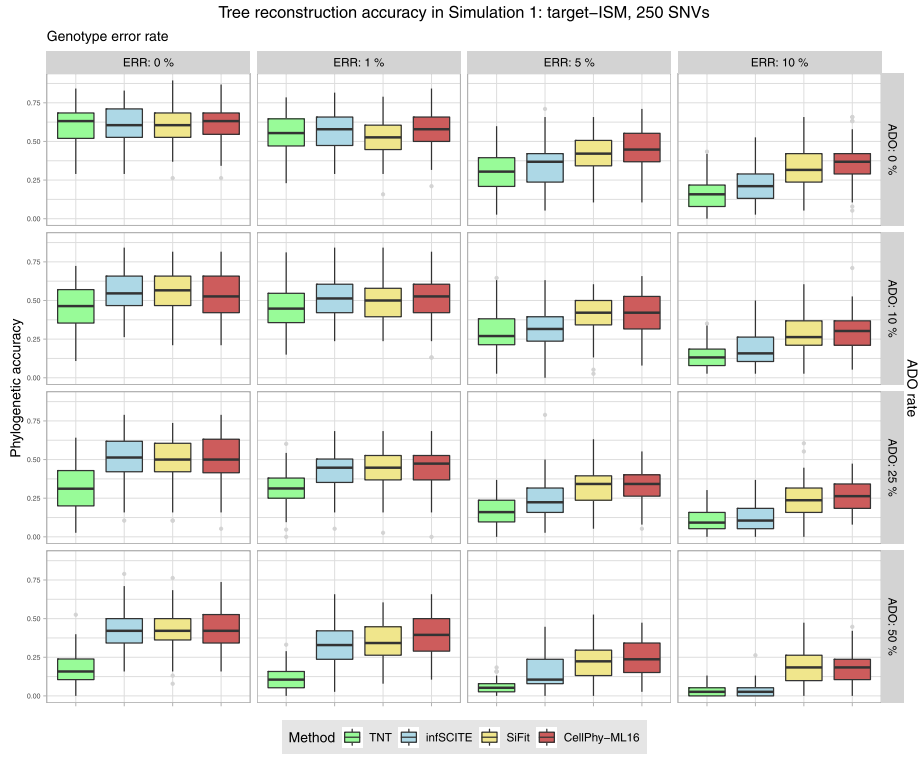

Fig. 1 Phylogenetic accuracy in simulation 1 ("target-ISM") with 250 SNVs. Datasets consisted of 40 cells. Accuracy was evaluated under different levels of genotype error (ERR), allele dropout (ADO), and genotype recoding strategy "missing." Phylogenetic accuracy is 1—nRF (see the "Methods" section). Boxplots were generated with the ggplot2 R package [58] (https://ggplot2.tidyverse.org) with default parameters. Lower and upper hinges correspond to the first and third quartiles. The upper whisker extends from the hinge to the largest value no further than 1.5 * IQR (IQR is the interquartile range or distance between the first and third quartiles). The lower whisker extends from the hinge to the smallest value at most 1.5 * IQR of the hinge. Data points beyond the end of the whiskers are called "outlier" points and plotted individually

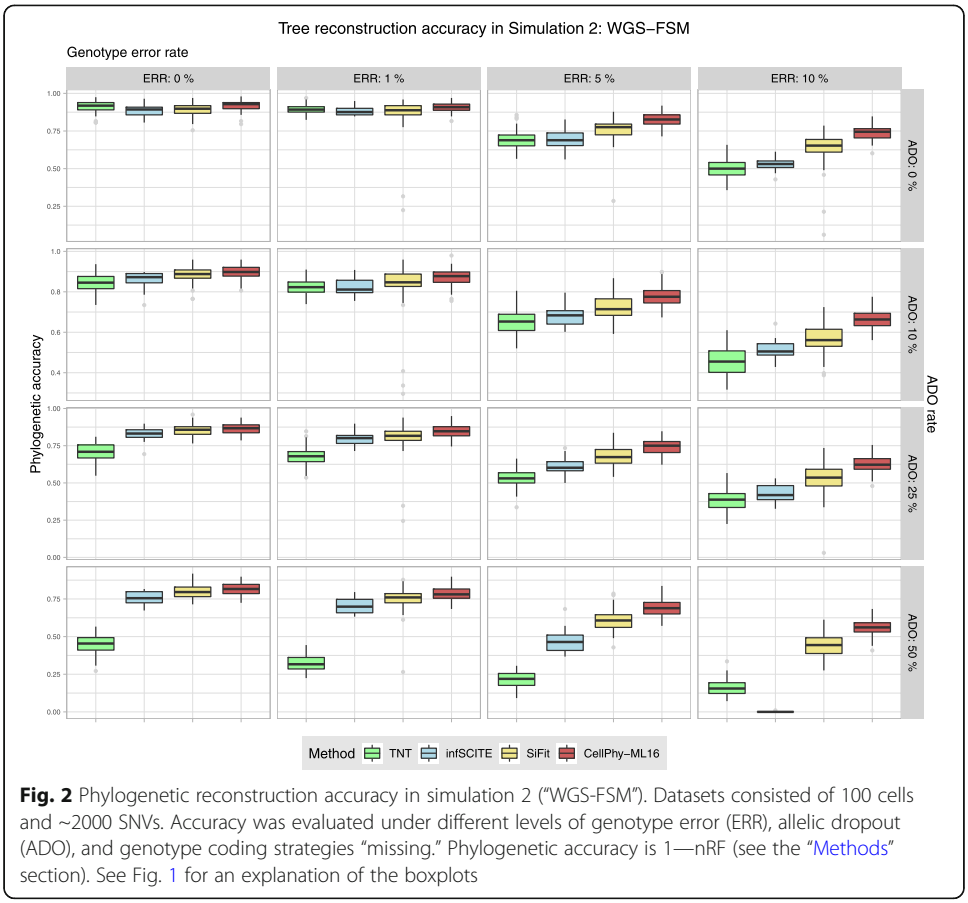

Fig. 2 Phylogenetic reconstruction accuracy in simulation 2 ("WGS-FSM"). Datasets consisted of 100 cells and ~2000 SNVs. Accuracy was evaluated under different levels of genotype error (ERR), allelic dropout (ADO), and genotype coding strategies "missing." Phylogenetic accuracy is 1—nRF (see the "Methods" section). See Fig. 1 for an explanation of the boxplots

genotype errors and ADO events. As in simulation 1, the "missing" coding strategy was slightly superior to "keep," particularly with higher genotype error rates, and substantially better than "remove" (data not shown). Thus, for subsequent simulations, we only considered the "missing" strategy.

Simulation 3: mutational signatures and large number of SNVs ("WGS-sig")

In simulation 3, we produced relatively large datasets (60 cells, 1000–4000 SNVs) under COSMIC trinucleotide mutational signatures 1 and 5, assuming an infinite-site model. The trends were as before, and CellPhy consistently outperformed the competing methods, especially with increasing levels of genotype error or ADO (Figs. S5-S6).

Simulation 4: genotype likelihoods from NGS read counts ("NGS-like")

Here, we simulated NGS data, and the input for tree inference consisted of read counts (SCIPhI and ScisTree), ML genotypes (CellPhy-ML and the remaining methods), or genotype likelihoods (CellPhy-GL). Across this scenario, with a realistic sequencing depth for single cells (5x), CellPhy performed much better than the competing methods, in particular under its genotype likelihood mode ("CellPhy-GL") (Fig. 3). At 30x and 100x, the accuracy differences were substantially more minor and CellPhy still performed as well as or better than the competing methods (Figs. S7-S8). ScisTree recovered highly inaccurate trees as the errors increased. Strikingly, the relative

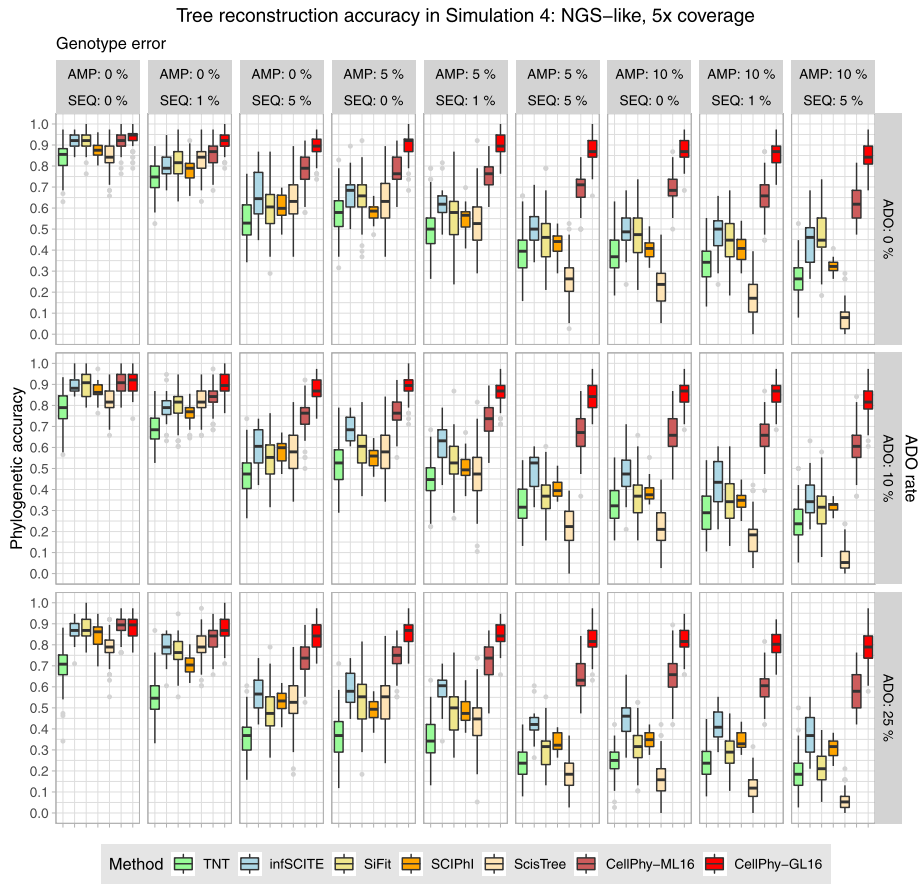

Fig. 3 Phylogenetic accuracy in simulation 4 ("NGS-like") at 5x. Datasets consisted of 40 cells and 1000–2000 SNVs. TNT, SiFit, and CellPhy-ML16 use the inferred genotypes, CellPhy-GL16 uses genotype likelihoods, and SCIPhI and ScisTree use read counts. Phylogenetic accuracy is 1—nRF (see the "Methods" section). AMP is the amplification error rate, SEQ is the sequencing error rate, and ADO is the allelic dropout rate. See Fig. 1 for an explanation of the boxplots

performance of SCIPhI and ScisTree worsened at 30× and fully degraded at 100× in the presence of single-cell noise.

Simulation 5: NGS doublets

In many single-cell experiments, cell doublets can be relatively common [24], so we also assessed their effect. As expected, the presence of cell doublets reduced phylogenetic accuracy for all methods. Still, CellPhy was the most accurate method, particularly in the presence of ADO and amplification errors (Fig. S9).

Simulation 6: NGS for large numbers of cells and SNVs

We assessed phylogenetic accuracy on large scDNA-seq datasets, with up to 1000 cells and 50,000 SNVs, without doublets. Here we could not evaluate infSCITE, as jobs were still running after 1 month. For the largest datasets, we could not obtain results for SCIPhI and ScisTree after running jobs for several days. CellPhy generally outperformed the competing methods, with rapidly increasing accuracy as a function of the number of SNVs, benefiting further from the use of genotype likelihoods (Fig. 4).

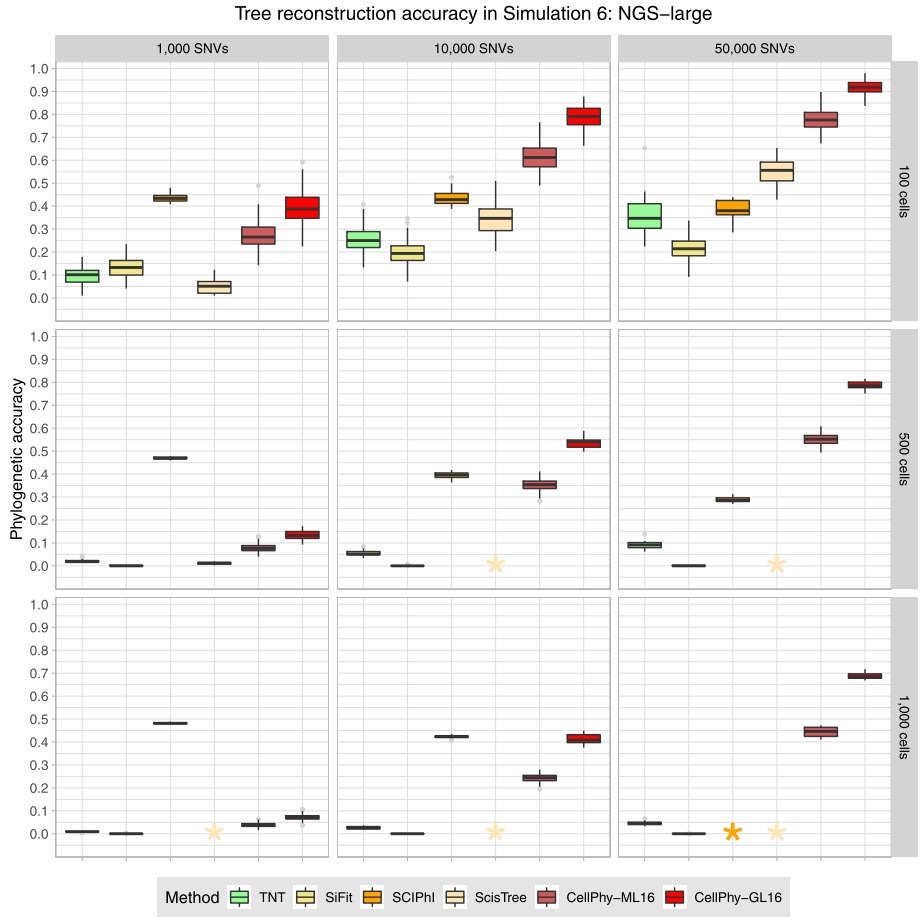

Fig. 4 Phylogenetic reconstruction accuracy in simulation 6 ("NGS-large"). Mutations were introduced according to signature S1, and the sequencing depth was 5x. Read counts were simulated with a 5% amplification error, 1% sequencing error, and 10% ADO. TNT, SiFit, and CellPhy-ML16 use the inferred genotypes, CellPhy-GL16, uses genotype likelihoods, and SCIPhI and ScisTree use read counts. Phylogenetic accuracy is 1—nRF (see the "Methods" section). For more than 1000 SNVs or more than 100 cells, only ten replicates were run for SiFit for reasonable running times. The (light) orange stars indicate that we could not obtain SCIPhI and ScisTree results for the largest datasets after running each replicate for >100 h. CellPhy was run using the ML genotypes error model (CellPhy-ML) and genotype likelihoods (CellPhy-GL). See Fig. 1 for an explanation of the boxplots.

Remarkably, SCIPhI showed a constant accuracy of ~0.4 across conditions. However, we noted that the SCIPhI trees contained many unresolved nodes, which decreases the possibility of false positives. Moreover, SCIPhI's accuracy decreased with more SNVs, highlighting a systematic bias.

Estimation of genotype error and ADO rates

Besides inferring the ML tree, CellPhy can calculate ML estimates for the genotyping error and the ADO rate of scDNA-seq datasets. Across the different simulation scenarios described above, CellPhy estimated the genotyping error quite accurately (MSE: 0.00003–0.002), with a slight over or underestimation when its actual value was below/above 5%, respectively (Fig. S10A). The ML estimates of ADO were more variable and tended to underestimate the actual value (MSE = 0.002–0.02), but were still generally

accurate, particularly at higher rates (Fig. S10B). As expected, in both cases, improved estimates were obtained with larger datasets comprising more SNVs.

Computational speed

We compared the speed of the different methods by recording the running times for six simulated and two empirical data sets (Fig. 5). TNT was the fastest method by at least two orders of magnitude, which is not surprising as parsimony scores are substantially cheaper to compute than likelihood scores. After TNT, CellPhy under both ML and GL models, and ScisTree—CellPhy being faster than ScisTree on larger data sets—were second-fastest, approximately one to two orders of magnitude faster than SiFit, SCIPhI, or infSCITE. For some of the most extensive data sets, including simulated and empirical data, infSCITE did not finish after several days. Regarding the methods discarded during simulation 1, OncoNEM and SASC were slower than infSCITE, and SPhyR crashed before finishing the speed benchmark due to memory allocation issues (data not shown). As expected, running 100 bootstrap replicates (+BS) increased computational time by 1–2 orders of magnitude for CellPhy, but only by one order of magnitude or less for TNT (Fig. S11). Still, the single-threaded version of CellPhy with 100 bootstrap replicates is faster than infSCITE. Furthermore, running CellPhy with several threads reduces its time-to-completion by one order of magnitude on a modern multicore machine (Fig. S11).

Application to single-cell data

Phylogenetic reconstruction of a colorectal cancer

We analyzed a single-cell WGS dataset (CRC24) produced in our lab, consisting of 24 cells collected from two primary tumor biopsies of a patient with colorectal cancer (CRC). After filtering out germline polymorphisms, SNVs in non-diploid regions, and low-quality variants (see the "Methods" section), we identified a total of 17,851 SNVs, of which 126 were exonic (Fig. S12). The high quality of these data is supported by the fact that the variant allele frequency distribution (VAF) derived from the single-cell

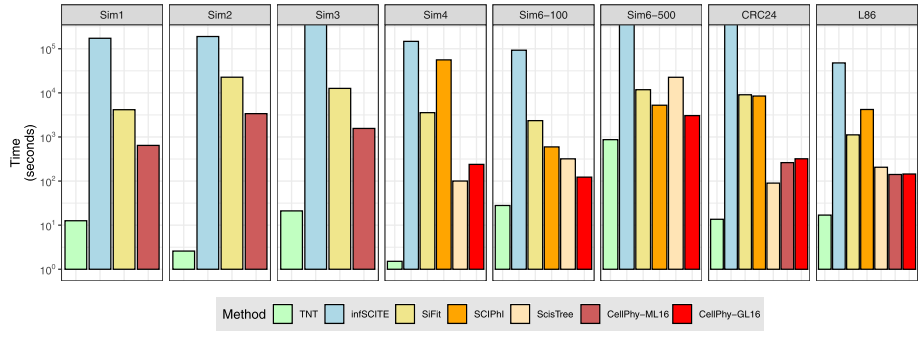

Fig. 5 Speed comparisons for simulated and real datasets. "Sim1" corresponds to a simulated single-cell dataset with 40 cells and 4753 SNVs. "Sim2" corresponds to a simulated dataset with 100 cells and 9935 SNVs. "Sim3" corresponds to a simulated dataset with 60 cells and 9982 SNVs. "Sim4" corresponds to a simulated dataset with 40 cells and 3986. "Sim6-100" corresponds to a simulated dataset with 100 cells and 1000 SNVs. "Sim6-500" corresponds to a simulated dataset with 500 cells and 1000 SNVs. CRC24 and L86 correspond to two empirical datasets (see the "Methods" section). Note the logarithmic time scale on the *y*-axis

SNVs is very similar to the VAF obtained from the bulk samples (Fig. S13). Some SNVs occurred in established CRC driver genes, such as *APC*, *BRAF*, *BRCA2*, *LRP1B*, and *MAP2K4*, although they were intronic. In the ML tree estimated by CellPhy using the genotype likelihoods ("CellPhy-GL" model) (Fig. 6A), cells tended to group according to their geographical location and phenotype, although not in a perfect fashion. Some of these relationships are well supported by the data, as reflected by several high bootstrap values, but others are not. This heterogeneous support illustrates one of the convenient features of CellPhy: it can provide phylogenetic confidence measurements for different parts of the tree. Interestingly, in the tumor interior region (TI), non-stem cells had longer branch lengths than stem cells, suggesting a potential difference in evolutionary rate between these two cell types [59–61]. We mapped the non-synonymous mutations onto the internal branches of the tree using a custom script (see the "Methods" section). We found that all tumor cells share the vast majority of these mutations (i.e., they are clonal), including variants affecting genes previously associated with CRC progression (e.g., *INPP1*, *CDC5L*, *ROR2*, *EXOSC5*) [62–65]. The tree topologies inferred by SiFit, SCIPhI, infSCITE, ScisTree, and TNT (Fig. S14) were distinct from the topology inferred by CellPhy (nRF=0.86, 0.59, 0.50, 0.64, and 0.68, respectively), but with a similar, albeit not identical, overall pattern regarding geography and cell type. Unfortunately, for the infSCITE, SiFit, SCIPhI, and ScisTree, the absence of branch support measures makes it impossible to determine which parts of the estimated trees can be trusted.

Revisiting the evolutionary history of a metastatic colorectal cancer

We also explored a published dataset from metastatic colorectal cancer patient CRC2 in Leung et al. [21]. In that study, after performing custom targeted sequencing of 182

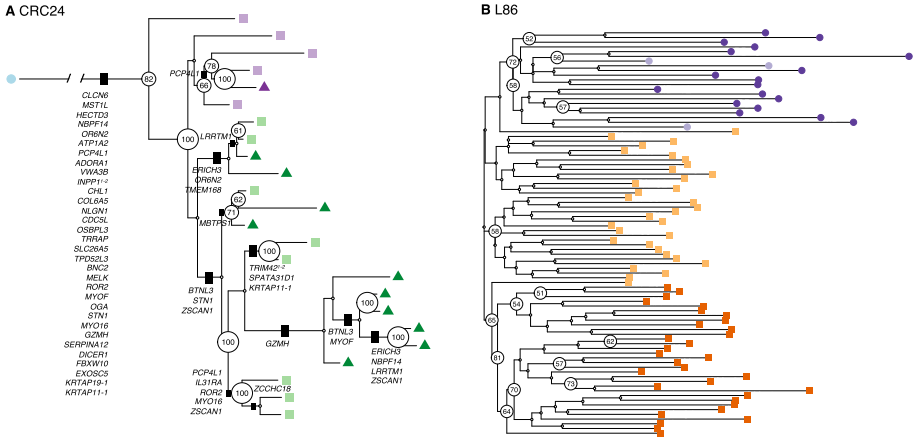

Fig. 6 CellPhy tree for the CRC24 and L86 datasets. **A** "CellPhy-GL" CRC24 tree. Distinct shapes and colors represent cell type: healthy (blue circle); tumor-non-stem from the tumor inferior (TI) region (dark green triangle), tumor-stem from TI region (light green square), tumor-non-stem from the tumor middle (TM) region (dark purple triangle), and tumor-stem from the TM region (light purple square). Only bootstrap support values above 50 are shown. Non-synonymous mutations are displayed on internal branches. **B** "CellPhy-GL" L86 tree. Distinct shapes and colors represent cell type: healthy diploid cells—from both primary and metastatic sites—(dark purple circle), healthy diploid cells missorted (light purple circle), primary tumor aneuploid cells (light orange square), and metastatic aneuploid cells (dark orange square). Only bootstrap values above 50 are shown

single cells sampled from primary and metastatic lesions, the authors derived a cell tree using SCITE. They inferred a polyclonal seeding of liver metastases (i.e., distinct populations of tumor cells migrated from the primary tumor towards the liver). However, their findings have been recently re-evaluated in two different studies. Zafar et al. [36] used the newly developed SiCloneFit, which relaxes the infinite-sites assumption, and proposed a polyclonal seeding of the metastases. More recently, Satas et al. [32] performed a joint analysis of SNVs and copy-number variants (CNVs) with SCARLET and concluded instead that a single clone seeded the liver metastasis. Our analysis focuses on cancer history, so we removed most of the healthy cells in the original dataset to speed up computation, ending with 86 cells (L86 data set) that we used for CNV and SNV calling. SC-Caller [66] identified 598 SNVs distributed over the genome portion not affected by CNVs, including several variants detected in the original study (e.g., *LINGO2*, *IL7R*, *MYH11*, *FUS*, *PTPRD*, *LRP1B*, *TSHZ3*), as well as other mutations affecting essential driver genes in CRC (e.g., *APC*, *TP53*, *NRAS*). Note that our set of SNVs only overlaps partially with the one in Leung et al., which is not surprising given that we used a different pipeline and a subset of their cells.

In the CellPhy tree, all metastatic cells clustered together with high support, indicating a monoclonal origin of the liver metastasis (Fig. 6B). Interestingly, some primary (PA-54 and PA-27) and metastatic aneuploid (MA-25 and MA-26) cells appear intermixed with healthy diploid cells; these correspond to cells mislabeled during FACS sorting, as previously noted by the authors. Furthermore, after mapping the non-synonymous mutations onto the internal branches of the CellPhy tree (Fig. S15), we found that all cancer cells harbor somatic variants affecting genes that can contribute to human intestinal neoplasia (i.e., *MYH11* and *STAG1*) [67, 68]. Apart from CellPhy, SCIPhI, and ScisTree also recovered a single metastatic clade (Fig. S16). Although in the SiFit tree, most metastatic cells cluster together, some of them appear intermixed with the primary tumor cells. In the infSCITE and TNT trees, the tumor cells did not form a clade, a result that does not appear to be realistic.

Applicability of CellPhy to non-cancer data and bulk clonal sequences

Finally, we used CellPhy to analyze two non-cancer WGS single-cell datasets. The first of these datasets (E15) consists of 242 somatic SNVs from 15 single neurons from a healthy individual [69]. In the CellPhy tree built using genotype likelihoods (Fig. S17A), different lineages seem to exhibit highly different evolutionary rates. However, most branches had low bootstrap support (<50%), suggesting that more SNVs are required before making reliable interpretations. All other methods recovered very distinct trees (Fig. S17B-F), and in the case of TNT, also with shallow bootstrap values. Unfortunately, infSCITE, SiFit, SCIPhI, and ScisTree do not assess branch support for cell trees.

The second dataset (LS140) consists of 140 single cell-derived human hematopoietic stem and progenitor colonies from a healthy individual [15]. In this case, because there is no single-cell genome amplification involved, we expect a minimal genotype error and no ADO. Remarkably, CellPhy ML estimates for this data set were zero for both error and ADO parameters. The CellPhy tree shows very high bootstrap values (Fig. S18), highlighting the quality of this dataset, which has a strong phylogenetic signal.

We expected this result given that the dataset has 127,884 SNVs and lacks single-cell biases. Moreover, our analysis confirms an early, well-supported divergence of the cell colonies into two distinct groups, together with a lack of geographical structure, hence reinforcing the idea of a continuous redistribution of stem cell pools at the whole-body level.

Discussion

We have developed CellPhy, a phylogenetic tool for analyzing single-cell SNV data inspired by existing models and methods in statistical phylogenetics. Unlike most of its competitors, CellPhy does not assume an infinite-site model of evolution, a widespread postulate in somatic evolution challenged by recent studies [27, 70]. CellPhy's evolutionary model explicitly considers all 16 possible phased DNA genotypes—but can work with both phased and unphased data—and can also account for their uncertainty by using genotype likelihoods as input. Furthermore, CellPhy can directly estimate single-cell error and ADO rates from the genotypes if the genotype likelihoods are unavailable. Finally, CellPhy can reconstruct ancestral states, predict mutations on tree branches, and provide a statistical measure of branch support. To benchmark CellPhy, we conducted computer simulations under different scenarios with varying degrees of complexity. Unlike in previous comparisons [25–28], we consider more realistic somatic genealogies by sampling sets of cells from a growing population. Demographic growth results in more difficult-to-reconstruct phylogenies, with shorter internal and longer terminal branches that require hundreds of SNVs to be accurately inferred.

Overall, CellPhy was the most accurate method, under infinite- and finite-site mutation models with different mutational patterns (e.g., using COSMIC trinucleotide mutational signatures), mainly when genotype errors and ADO were common, which is often the case for scDNA-seq data. Our simulations suggest that accounting for SNV calling uncertainty is essential when sequencing depth is low to moderate, which is typically the case for single-cell WGS due to the sequencing costs. With a 5x sequencing depth, the ability to account for genotype uncertainty makes CellPhy substantially more accurate than its competitors. Notably, the accuracy of CellPhy does not come at the cost of speed. CellPhy is one to two orders of magnitude faster than SiFit, infSCITE, or SCIPhI. Although, as expected, parsimony-based TNT was by far the fastest tool, this comes at the cost of considerably worse accuracy under most scenarios. Moreover, CellPhy allows multi-threading, so running times could be considerably shorter if a multi-core computer—very common nowadays—is used, even with bootstrapping. CellPhy's bootstrap analyses illustrate the importance of explicitly considering phylogenetic uncertainty. Without such a measure, one cannot assess if the data support the estimated phylogeny. Further, the analysis of cell colonies shows that CellPhy can also be used to estimate trees from clonal sequences that do not necessarily correspond to amplified single-cells. As expected, and as shown in previous studies (e.g., [26, 28]), cell doublets can be detrimental for single-cell phylogenetic reconstruction. When doublets are likely to occur in the data, specific scDNA-seq doublet detection methods might be used to remove suspicious cells [71].

Our results suggest that CellPhy is an efficient tool for estimating phylogenetic trees from scDNA-seq data. It is important to note that to achieve high phylogenetic accuracy under realistic conditions (i.e., growing populations with long terminal branches),

CellPhy requires a reasonably high number of SNVs (hundreds to tens of thousands, depending on the number of cells). However, as can be seen in our benchmark, this limitation is common to all methods. It reflects the fundamental problem of poor signal-to-noise ratio: the faster the cell population grows, the more cells we have, and the higher the single-cell error and ADO rates are, the more SNVs are needed to recover the true cell relationships. We expect that further improvements in single-cell sequencing accuracy, variant calling sensitivity, and genotype phasing will yield datasets that are better suited for phylogenetic inference. Furthermore, some common simplifying assumptions of the classical phylogenetic models (reversibility, stationarity, context-independence) could be particularly problematic in the context of somatic evolution, which takes place at a much shorter temporal scale. Albeit methodologically and computationally demanding, eliminating those assumptions could improve accuracy on scDNA-seq datasets with a weak phylogenetic signal. Finally, another critical challenge for the future, particularly relevant for cancer data, will be developing models that simultaneously take into account SNVs and CNVs. While more or less simple methods have been proposed [32, 34, 40, 72], a full SNV + CNV phylogenetic Markov model is not trivial because copy number variants can overlap, breaking the site independence assumption typical of many phylogenetic models, including CellPhy.

Conclusions

The main focus of CellPhy, and its principal output, is the estimation of cell lineage trees. Our extensive simulations show that CellPhy can produce accurate cell phylogenies for small and large scDNA-seq SNV datasets within a reasonable timeframe. Cell phylogenies encode information regarding temporal diversification, demographic history, migration, and adaptation and can shed light on multiple aspects of cell biology [11]. CellPhy's model is a step forward towards more realistic models of somatic genotype evolution and can be applied to any population of somatic cells that divide asexually. Therefore, the use of CellPhy is not limited to cancer cells, as it could also be applied to normal cells, for example, to better understand human development [13–16].

Methods

The CellPhy model

Model of nucleotide substitution for phased/unphased diploid genotypes

We developed a substitution model for diploid genotypes akin to those typically used for DNA sequences in organismal phylogenetics ([see [51]). Specifically, we built a finite-site, continuous-time, Markov model of genotype evolution considering all possible 16 phased diploid states Γ= {A|A, A|C, A|G, A|T, C|A, C|C, C|G, C|T, G|A, G|C, G|G, G|T, T|A, T|C, T|G, T|T}, in which SNVs are independent and identically distributed. This model, named GT16, is defined by a rate matrix Q that contains the instantaneous transition rates $q_{X \leftrightarrow Y}$ among genotypes X and Y. For computational convenience, we assume a time-reversible process, in which case the Q matrix is the product of a symmetric exchangeability matrix R ($r_{X \leftrightarrow Y} = r_{Y \leftrightarrow X}$) and a diagonal matrix of stationary genotype frequencies π_X:

$$Q = R \cdot \ diag\left(\pi_{A|A}, \pi_{A|C}, ..., \pi_{T|T}\right) \tag{1}$$

We assume that only one of the two alleles in a genotype can change in an infinitesimal amount of time. Furthermore, because maternal and paternal chromosomes evolve independently in somatic cells, we also assume that the instantaneous transition rate of a given allele *a* in genotype *X* to allele *b* in genotype *Y* does not depend on the identity of the homologous allele *n* within the respective genotypes. In other words, we assume r(*na*↔*nb)* = r(*b*↔*a)*. Conveniently, these assumptions significantly reduce the number of free parameters in the *Q* matrix, as we only need to consider five nucleotide exchangeabilities (α = *r(A*↔*C)*, *β* = *r(A*↔*G)*, γ= *r(A*↔*T)*, *κ* = *r(C*↔*G)*, λ = *r(C*↔*T)*; *let* μ = *r(G*↔*T) = 1)* and 15 stationary-phased genotype frequencies ($\pi_{A|A}$, $\pi_{A|C}$, $\pi_{A|G}$, $\pi_{A|T}$, $\pi_{C|A}$, $\pi_{C|C}$, $\pi_{C|G}$, $\pi_{C|T}$, $\pi_{G|A}$, $\pi_{G|C}$, $\pi_{G|G}$, $\pi_{G|T}$, $\pi_{T|A}$, $\pi_{T|C}$, $\pi_{T|G}$; $\pi_{T|T}$= 1 - $\sum\pi_X$):

$$Q = \begin{array}{c} \\ A|A \\ C|C \\ G|G \\ T|T \\ A|C \\ A|G \\ A|T \\ C|G \\ C|T \\ G|T \\ C|A \\ G|A \\ T|A \\ G|C \\ T|C \\ T|G \end{array} \begin{array}{c} \begin{array}{cccccccccccccccc} A|A & C|C & G|G & T|T & A|C & A|G & A|T & C|G & C|T & G|T & C|A & G|A & T|A & G|C & T|C & T|G \end{array} \\ \begin{pmatrix} -q_{A|A} & 0 & 0 & 0 & \alpha\pi_{A|C} & \beta\pi_{A|G} & \gamma\pi_{A|T} & 0 & 0 & 0 & \alpha\pi_{C|A} & \beta\pi_{G|A} & \gamma\pi_{T|A} & 0 & 0 & 0 \\ 0 & -q_{C|C} & 0 & 0 & \alpha\pi_{A|C} & 0 & 0 & \kappa\pi_{C|G} & \lambda\pi_{C|T} & 0 & \alpha\pi_{C|A} & 0 & 0 & \kappa\pi_{G|C} & \lambda\pi_{T|C} & 0 \\ 0 & 0 & -q_{G|G} & 0 & 0 & \beta\pi_{A|G} & 0 & \kappa\pi_{C|G} & 0 & \mu\pi_{G|T} & 0 & \beta\pi_{G|A} & 0 & \kappa\pi_{G|C} & 0 & \mu\pi_{T|G} \\ 0 & 0 & 0 & -q_{T|T} & 0 & 0 & \gamma\pi_{A|T} & 0 & \lambda\pi_{C|T} & \mu\pi_{G|T} & 0 & 0 & \gamma\pi_{T|A} & 0 & \lambda\pi_{T|C} & \mu\pi_{T|G} \\ \alpha\pi_{A|A} & \alpha\pi_{C|C} & 0 & 0 & -q_{A|C} & \kappa\pi_{A|G} & \lambda\pi_{A|T} & 0 & 0 & 0 & 0 & 0 & 0 & \beta\pi_{G|C} & \gamma\pi_{T|C} & 0 \\ \beta\pi_{A|A} & 0 & \beta\pi_{G|G} & 0 & \kappa\pi_{A|C} & -q_{A|G} & \mu\pi_{A|T} & \alpha\pi_{C|G} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\pi_{T|G} \\ \gamma\pi_{A|A} & 0 & 0 & \gamma\pi_{T|T} & \lambda\pi_{A|C} & \mu\pi_{A|G} & -q_{A|T} & 0 & \alpha\pi_{C|T} & \beta\pi_{G|T} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \kappa\pi_{C|C} & \kappa\pi_{G|G} & 0 & 0 & \alpha\pi_{A|G} & 0 & -q_{C|G} & \mu\pi_{C|T} & 0 & \beta\pi_{C|A} & 0 & 0 & 0 & 0 & \lambda\pi_{T|G} \\ 0 & \lambda\pi_{C|C} & 0 & \lambda\pi_{T|T} & 0 & 0 & \alpha\pi_{A|T} & \mu\pi_{C|G} & -q_{C|T} & \kappa\pi_{G|T} & \gamma\pi_{C|A} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mu\pi_{G|G} & \mu\pi_{T|T} & 0 & 0 & \beta\pi_{A|T} & 0 & \kappa\pi_{C|T} & -q_{G|T} & 0 & \gamma\pi_{G|A} & 0 & \lambda\pi_{G|C} & 0 & 0 \\ \alpha\pi_{A|A} & \alpha\pi_{C|C} & 0 & 0 & 0 & 0 & 0 & \beta\pi_{C|G} & \gamma\pi_{C|T} & 0 & -q_{C|A} & \kappa\pi_{G|A} & \lambda\pi_{T|A} & 0 & 0 & 0 \\ \beta\pi_{A|A} & 0 & \beta\pi_{G|G} & 0 & 0 & 0 & 0 & 0 & 0 & \gamma\pi_{G|T} & \kappa\pi_{C|A} & -q_{G|A} & \mu\pi_{T|A} & \alpha\pi_{G|C} & 0 & 0 \\ \gamma\pi_{A|A} & 0 & 0 & \gamma\pi_{T|T} & 0 & 0 & 0 & 0 & 0 & 0 & \lambda\pi_{C|A} & \mu\pi_{G|A} & -q_{T|A} & 0 & \alpha\pi_{T|C} & \beta\pi_{T|G} \\ 0 & \kappa\pi_{C|C} & \kappa\pi_{G|G} & 0 & \beta\pi_{A|C} & 0 & 0 & 0 & 0 & 0 & \lambda\pi_{G|T} & 0 & \alpha\pi_{G|A} & 0 & -q_{G|C} & \mu\pi_{T|C} & 0 \\ 0 & \lambda\pi_{C|C} & 0 & \lambda\pi_{T|T} & \gamma\pi_{A|C} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha\pi_{T|A} & \mu\pi_{G|C} & -q_{T|C} & \kappa\pi_{T|G} \\ 0 & 0 & \mu\pi_{G|G} & \mu\pi_{T|T} & 0 & \gamma\pi_{A|G} & 0 & \lambda\pi_{C|G} & 0 & 0 & 0 & 0 & \beta\pi_{T|A} & 0 & \kappa\pi_{T|C} & -q_{T|G} \end{pmatrix} \end{array} \tag{2}$$

The *Q* matrix gives us the probabilities of changing from one genotype to another in an infinitely small time interval. When two genotypes differ at more than one nucleotide, these rates are 0. To calculate the tree likelihood, we first need to compute the change probabilities from the beginning to the end of each branch. The probabilities of changing from a given genotype to another along a branch of length *t* (in units of expected number of mutations per SNV site) are given by:

$$P(t) = e^{Qt} \tag{3}$$

The resulting matrix *P(t)* is known as the transition probability matrix, and all its entries are (small) positive values. For further details, we refer the reader to Felsenstein [51] and Yang [73].

Notably, the GT16 model can also work with unphased genotype data simply by assigning the same relative likelihood at the tips of the tree to the two possible resolutions of the phase for the observed genotype (see Table S2). This flexibility is relevant because current scDNA-seq techniques do not reveal the phase of the genotypes (i.e., we do not know which allele is located in the maternal or paternal chromosome). Therefore, in our simulations (see below), the simulated genotypes will always be unphased.

In addition to the GT16 model, we also implemented a model for unphased genotypes with only ten states, called GT10 (see Supplementary Note 2 for details). The GT10 model is computationally less expensive, approximately twice as fast as GT16. To maintain the reversibility assumption—otherwise, the calculations are much more complex—the GT10 model assumes that the probability of change between homozygous and heterozygous genotypes is equivalent. However, this is incorrect, as the

change from homozygote to heterozygote is twice as likely as the change from heterozygote to homozygote. Despite this theoretical flaw, GT10 and GT16 showed very similar tree inference accuracy in our experiments.

Single-cell genotype error model

We assume that the true genotypes are always diploid and biallelic. To incorporate errors in the observed genotypes arising during single-cell whole-genome amplification (scWGA) and sequencing, we consider two free parameters: the allelic dropout (ADO) rate (δ) and the amplification/sequencing error (ERR) rate (ε).

Allele dropout occurs during scWGA when one of the two alleles is not amplified and is absent from the sequencing library. ADO always implies a single allele in our model, as if both alleles drop, no reads would be available, and no genotype could be observed (i.e., it would be "missing"). Therefore, δ is the probability that the amplification of one or the other allele has failed and thus that we observe the homozygous genotype defined by the amplified allele. Given the phased genotype $a|b$, and with "_" indicating the dropped allele, we define the ADO rate as:

$$\delta = \mathrm{P}(\_|b|a|b) + \mathrm{P}(a|\_|a|b), \tag{4}$$

where $P\ (Y|X)$ is the probability of observing genotype Y after sequencing, given the true genotype X.

Genotype errors other than ADO can result from polymerase errors during scWGA, in the course of sequencing, variant calling, or generally represent an incorrect allele. Importantly, we assume a maximum of one ERR per genotype. Given that ε tends to be small, in the order of 10^{-3} to 10^{-5} [74], the probability of two ERR in one genotype, ε^2, is negligible. Specifically, ε is the probability that allele a will be observed as another allele $b \neq a$:

$$\varepsilon = \mathrm{P}(b|a) \tag{5}$$

Note that we allow for the presence of both ADO and ERR in the same observed genotype. Under these assumptions, if the true genotype is homozygous $a|a$, for phased genotypes $P\ (Y|X)$ becomes:

$$P(a|a|a|a) = 1-\varepsilon + {}^{1}/_{2}\delta\varepsilon \tag{6}$$

$$P(a|b|a|a) = (1-\delta)\cdot{}^{1}/6\varepsilon \tag{7}$$

$$P(b|b|a|a) = {}^{1}/_{6}\delta\varepsilon \tag{8}$$

Likewise, if the true genotype is heterozygous $a|b$, $P\ (Y|X)$ can be shown to be:

$$P(a|a|a|b) = {}^{1}/_{2}\delta + {}^{1}/_{6}\varepsilon - {}^{1}/_{3}\delta\varepsilon \tag{9}$$

$$P(c|c|a|b) = {}^{1}/_{6}\delta\varepsilon \tag{10}$$

$$P(a|c|a|b) = (1-\delta)\cdot{}^{1}/6\varepsilon \tag{11}$$

$$P(a|b|a|b) = (1-\delta) \cdot (1-\varepsilon) \quad (12)$$

For the remaining scenarios, given the assumptions of the error model, $P(Y|X)$ is zero. See Supplementary Material Note 1 for a detailed explanation.

Given the observed genotype Y for the SNV i and cell j, the likelihood of a genotype X being the true genotype at this position becomes:

$$L_i^j(X|Y) = P(Y|X), \forall X \in \Gamma \quad (13)$$

This follows from Bayes' theorem assuming equal prior genotype probabilities.

For missing data, we consider all possible genotypes to be equally likely, which is a standard assumption in likelihood-based phylogenetic inference. In this case:

$$L_i^j(X|missing) = P(missing|X) = 1, \forall X \in \Gamma \quad (14)$$

Phylogenetic likelihood

We consider the evolutionary history of cells as an unrooted binary tree, where τ is a tree topology, and t is a vector of branch lengths. We define the phylogenetic likelihood of the cell tree T as the conditional probability of the observed SNV matrix S given the substitution model M with parameters θ and T:

$$L(T, M, \theta) = P(S \mid T, M, \theta) \quad (15)$$

We assume that genomic sites evolve independently under model M. Therefore, the probability of observing the SNV matrix S is the product over the independent probabilities for all individual SNVs, S_i:

$$L(T, M, \theta) = \prod_{i=1}^{s} P(S_i|T, M, \theta) \quad (16)$$

where s is the total number of SNVs. For numerical reasons, that is, to avoid floating-point underflow, in practice, we calculate the log-likelihood score:

$$\log L(T, M, \theta) = \sum_{i=1}^{s} \log P(S_i|T, M, \theta) \quad (17)$$

We can efficiently compute the log-likelihood score of a given tree with the standard method called Felsenstein's pruning algorithm [52]. Let us place an additional imaginary node, called *virtual root*, into an arbitrary branch of the unrooted tree T at a random position along that branch. Without loss of generality, if we assume that this virtual root node has an index of 0 and that c is the number of cells, we can index the tip nodes from 1 to c by their respective cell numbers and index the internal nodes from c+1 to 2c–2 (note that an unrooted binary tree with c leaves has c–3 inner nodes, i.e., (2c–2) – (c+1) = c–3). Then, under the GT16 model, per-SNV probabilities can be computed as follows:

$$P(S_i|T, GT16, R, \pi) = \sum_{X \in \Gamma} L_i^0(X)\pi_x, i = 1 \ldots s \quad (18)$$

where X is a genotype, $\pi_x$is the stationary frequency of the genotype X, and $L_i^0$is the vector of genotype likelihoods at the virtual root, which we compute as follows:

$$L_i^v(X) = \sum_{Y \in \Gamma} P_{X,Y}(t_u) L_i^u(Y) \cdot \sum_{Y \in \Gamma} P_{X,Y}(t_w) L_i^w(Y), \forall X \in \Gamma, v \notin [1 \ldots c] \quad (19)$$

where Y is another genotype, u and w are both children nodes of v in the direction from the virtual root (Fig. S19) for the respective branch lengths.

We initialize the genotype likelihood vectors at the tip nodes ($L_i^1 .. L_i^c$) depending on the input type (see Section "Input data" below) and the error model. If the input is the genotype matrix S, in the absence of observational error, these tip likelihood vectors are:

$$L_i^v(X) = \{1 if S_i^v = X, 0 otherwise\}, v \in [1...c], i \in [1...s] \tag{20}$$

while if we include the error model, the tip likelihoods are computed according to the equations in the "single-cell genotype error model" section.

Otherwise, if the input consists of genotype likelihoods G, the tip likelihoods are:

$$L_i^v(X) = 10^{G_i^v(X)}, \forall X \in \Gamma, v \in [1...c], i \in [1...s] \tag{21}$$

The GT16 model will consider both phasing options equally likely when the input genotype likelihoods are unphased:

$$L_i^v(a|b) = L_i^v(b|a) = 10^{G_i^v(a/b)}, \forall a, b, v \in [1...c], i \in [1...s] \tag{22}$$

where a and b represent any two alleles.

Finally, if Phred-scaled likelihoods for REF/REF, REF/ALT, and ALT/ALT genotypes are provided (PL field in a VCF file), we calculate tip likelihoods as follows:

$$L_i^v(X) = L_i^v(a|b) = \begin{cases} 10^{-0.1 \cdot PL_i^v(0)} & \text{if } a = REF \wedge b = REF \\ 10^{-0.1 \cdot PL_i^v(1)} & \text{if} (a = REF \wedge b = ALT) \vee (a = ALT \wedge b = REF) \\ 10^{-0.1 \cdot PL_i^v(2)} & \text{if } a = ALT \wedge b = REF \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

The final tree can be rooted using an outgroup (see [51]). For the sake of completeness, we included a simple explanation of standard phylogenetic likelihood calculations on DNA sequence alignments in the Supplementary Material (Figs. S19-S20, Supplementary Note 3).

Implementation

Overview

We implemented CellPhy as a pipeline based on a modified version of RAxML-NG [48]. In addition to the core tree search functionality of RAxML-NG, the CellPhy pipeline offers VCF conversion, mutation mapping, and tree visualization using the *ape* [75] and the *ggtree* package [76]. Furthermore, CellPhy provides reasonable defaults for most parameters, which allows the user to run a "standard" CellPhy analysis by specifying just the input VCF file (or the genotype matrix). Alternatively, expert users can customize every aspect of the CellPhy analysis to fit their needs, as shown in the tutorial (https://github.com/amkozlov/cellphy/blob/master/doc/CellPhy-Tutorial.pdf). In the remainder of this section, we will also provide implementation details on each step of

the CellPhy pipeline. CellPhy code and documentation are freely available at https://github.com/amkozlov/cellphy.

Input data

CellPhy accepts two input types, a matrix of genotypes in FASTA or PHYLIP format or a standard VCF file (https://samtools.github.io/hts-specs/VCFv4.3.pdf). When the input is a genotype matrix, genotypes are encoded as shown in Table S2. When the input is a VCF, CellPhy can run in two distinct modes. The first mode ("CellPhy-ML") requires a VCF with at least the GT field (that stores the genotype calls), in which case CellPhy simply extracts the genotype matrix. The second mode ("CellPhy-GL") requires a VCF with the PL field (which stores the Phred-scaled genotype likelihoods) and uses the likelihood of each genotype instead. While commonly used variant callers for single-cell data (e.g., Monovar [77]) generate VCF files with a standard PL field, users should know that the PL definition may differ from its standard meaning in different callers. Indeed, SC-Caller [66], for instance, uses the PL field to store the likelihood of heterozygous and alternative homozygous genotypes and the likelihood of sequencing noise and amplification artifacts. On this basis, the PL field in VCF files stemming from SC-Caller needs to be converted to the standard PL format before CellPhy can be used (see Table S3 outlining CellPhy's compatibility with popular variant calling algorithms).

Phylogenetic tree search

CellPhy uses the broadly used and well-tested heuristic tree search strategy of RAxML [53] and RAxML-NG [48]. CellPhy's search algorithm starts by default with 20 different trees, ten obtained using a maximum parsimony-based randomized stepwise addition order routine and ten with a completely random topology. The ML tree search itself alternates between model parameter optimization and tree topology optimization phases. One of the most popular approaches for searching tree topologies is the so-called subtree pruning and regrafting (SPR) [78], which removes subtrees from the tree and subsequently places them into different branches to assess if the likelihood improves. Those SPR moves are applied iteratively until no SPR move further increases the tree's likelihood. In this case, CellPhy terminates and returns the best-found ML tree. For further details, please see [79] and references therein.

Model parameter optimization

CellPhy uses the L-BFGS-B method [80] to optimize genotype substitution rates and equilibrium base frequencies. ADO rate and amplification/sequencing error rate are optimized independently using Brent's method [81]. After each iteration of Brent's algorithm, CellPhy re-computes all per-genotype likelihoods according to Eq. 7 using the new values of the ADO rate $\delta`$ and the amplification/sequencing error ε'.

Branch support

CellPhy can compute confidence values for individual inner branches of the ML tree using two bootstrap (BS) techniques, the standard BS [55], and the transfer BS [56]. In the standard BS, the first step consists of generating many BS replicates, typically 100 to 1000, from the original dataset by randomly sampling SNV sites with replacement.

Then, an ML tree is estimated for each replicate. Finally, the support for each inner branch in the ML tree is computed as the percentage of BS trees that contain that branch, as outlined in the example provided in Fig. S21. The standard BS only considers exact matches (i.e., the branch in the ML tree and the BS trees must match exactly to be counted). In contrast, the transfer BS also considers inexact matches to account for the tree search uncertainty and vastness of the tree search space in phylogenetic analyses with many cells.

Mapping mutations onto the tree

CellPhy can show predicted mutations on the branches of the inferred cell tree. To this end, it performs marginal ancestral state reconstruction [57] to obtain the ML genotype for every SNV at every inner node of the tree. At the tips of the tree, occupied by the observed cell genotypes, depending on the input, CellPhy applies Eq. 4 to compute genotype likelihoods given the observed genotype y and estimated error rates (δ, ε) or directly uses the genotype likelihoods provided in the VCF file. Then, it compares ML genotypes between two nodes connected by a branch, and if they differ, a mutation is predicted on the corresponding branch. The mutation mapping output consists of two files, a branch-labeled tree in the Newick format and a text file with a list of predicted mutations (SNV names or positions) at each branch. We also provide a script (https://github.com/amkozlov/cellphy/blob/master/script/mutation-map.R) that automatically generates a plot with the mutations mapped onto the resulting phylogenetic tree, together with a tutorial that explains its use.

Computational efficiency

RAxML-NG was developed with a particular focus on high performance and scalability to large datasets. Hence, CellPhy capitalizes on numerous computational optimizations implemented therein, including highly efficient and vectorized likelihood calculation code, coarse- and fine-grained parallelization with multi-threading, checkpointing, and fast transfer bootstrap computation [82].

Benchmarking

We used computer simulations to benchmark the accuracy of CellPhy under different scenarios (Table S1) relative to state-of-the-art methods for single-cell phylogenies like OncoNEM [25], SPhyR [30], SASC [31], ScisTree [33], infSCITE [26, 27], SiFit [28], and SCIPhI [29]. For OncoNEM, SPhyR, SASC, infSCITE, and SiFit, the input is the matrix of observed reference/non-reference homozygous/heterozygous genotypes. ScisTree uses a matrix of genotype probabilities estimated from the simulated read counts for each site, while SCIPhI relies on a standard mpileup file with the simulated read counts. Also, we included in the comparison the standard phylogenetic method TNT [49]. TNT implements a maximum parsimony (MP) approach and attempts to find the tree/s that require the least number of mutations to explain the data. TNT is very popular in MP organismal phylogenetics, heavily optimized for computational speed and efficiency. It is not designed for single-cell NGS data and therefore assumes that the observed genotypes are error-free.

Simulation of genealogies and genotypes

For the simulation of single-cell diploid genotypes, we used CellCoal [83]. This program can simulate the evolution of a set of cells sampled from a growing population, introducing single-nucleotide variants on the coalescent genealogy under different models of DNA mutation. Furthermore, it can also introduce the typical errors of single-cell sequencing, specifically ADO, amplification, and sequencing errors, and doublets, either to the observed genotypes or directly into the read counts.

We designed six distinct simulation scenarios (simulations 1–6) representing different types of scDNA-seq datasets (Table S1), including variable numbers of cells (40–1000) and sites (1000–50,000). We simulated unphased genotype data in all cases, as current scDNA-seq techniques do not reveal the genotype phase. We chose a set of scenarios and parameter values that, in our opinion, are representative of different situations that researchers are likely to encounter. The cell samples were assumed to come from an exponentially growing population (growth rate equal to 1×10^{-4}) with a present-day effective population size of 10,000. Across scenarios, we set a constant value of 0.1 for the root branch. Note that the mutations in this branch are shared by all cells. We also defined an outgroup branch length of zero in all cases, so the healthy cell and the most recent common ancestor (MRCA) of the sample (a single healthy cell plus several tumor cells) have identical genotypes. The standard coalescent process results in an ultrametric tree, where all tips have the same distance from the MRCA of the sample. However, we introduced rate variation across cell lineages by multiplying the branch lengths of the resulting coalescent genealogy with scaling factors sampled from a Gamma distribution with a mean of 1.0.

Only in the first simulation scenario, we consider a fixed number of SNVs. In the remaining scenarios, the number of observed mutations resulted from applying a mutation rate of 1×10^{-6} [84], plus the different scDNA-seq errors. We explored different infinite- and finite-site mutation models at the single nucleotide or trinucleotide level. Except for the ISM scenarios (simulations 1 and 3), we introduced a variable mutation rate across sites using a Gamma distribution with a mean of 1.0.

We simulated unphased genotypes, as current scDNA-seq techniques do not reveal the phase. We generated the observed genotype matrices in two distinct ways. In the first three scenarios (simulations 1–3), we obtained the observed genotypes by directly adding sequencing/amplification errors (i.e., changing one or both alleles) and ADO to the simulated genotype matrices. In the other three (simulations 4–6), the generation of the observed genotypes was more complex. In this case, we first simulated read counts for each cell based on the true genotypes, considering different overdispersed sequencing depths, as well as amplification and sequencing errors. For simplicity, we assumed that maternal and paternal chromosomes are amplified with the same probability. Also, we consider that the number of reads is half for those genomic positions in which only one allele was amplified. In simulation 5, we introduced so-called doublets, that is, two cells that are erroneously sequenced together and that thus appear as a single cell in the sequencing data. For every combination of parameters, we generated 100 replicates. In total, we generated 19,400 cell samples.

Preliminary simulations

We carried out a set of preliminary simulations under favorable conditions, without errors or relatively low noise levels (ADO = 0.10 and amplification error= 0.01). We

simulated 40 cells with 250 SNVs, assuming a diploid ISM model [85]. Under this model, a given site can only accumulate a single mutation along with the genealogy, either in the maternal or paternal chromosome.

Simulation 1: infinite-site model, low number of SNVs ("target-ISM")

We started the full simulations with a simple scenario with 40 sampled cells and 250, 500, or 1000 SNVs, assuming a diploid ISM model [85]. We introduced genotype errors and ADO at different rates (Table S1).

Simulation 2: finite-site model, large number of SNVs ("WGS-FSM")

In this case, we simulated a larger number of SNVs, more typical of whole-genome sequencing (WGS) experiments. The number of sampled cells was 100. The mutation model, in this case, was a non-reversible version of the finite-site General Time Reversible Markov model [50], that we called GTnR, assuming a set of single-nucleotide instantaneous rates extrapolated (essentially, we pooled the same mutation in the same rate independently of the 5′ and 3′ context) from the trinucleotide mutational signature 1 at COSMIC (https://cancer.sanger.ac.uk/cosmic/signatures):

$$Q_{GTnR} = \begin{bmatrix} 0 & 0.03 & 0.12 & 0.04 \\ 0.11 & 0 & 0.02 & 0.68 \\ 6.68 & 0.02 & 0 & 0.11 \\ 0.04 & 0.12 & 0.13 & 0 \end{bmatrix} \quad (24)$$

The overall mutation rate was set to 1×10^{-6}, which resulted in about 2000 *true* SNVs (see Table S1). However, since ADO events and genotype errors can introduce false negatives and false positives, the number of *observed* SNVs ranged between 1531 and 10,000. The mutation rates varied across sites according to a Gamma distribution (+G) with shape parameter and mean equal to 1.0 (i.e., moderate among-site rate heterogeneity).

Simulation 3: mutational signatures and large number of SNVs ("WGS-sig")

This scenario is similar to the previous one, with 60 cells and assuming a trinucleotide ISM model, with COSMIC signatures 1 and 5. The former is a ubiquitous signature in human cells with a predominance of C>T transitions in the NCG trinucleotide context and is related to the spontaneous deamination of 5-methylcytosine [86]. The latter is also a typical age-related signature with a predominance of T>C substitutions in the ATN trinucleotide context, related to transcriptional strand bias [87].

Simulation 4: genotype likelihoods from NGS read counts ("NGS-like")

In this scenario, and the next two, we simulated NGS read counts from the simulated genotypes. The number of sampled cells was 40, with 10,000 sites and the same mutation model (GTnR+G) and mutation rates as in Simulation 3. We explored three sequencing depths (5x, 30x, and 100x), three ADO rates (0, 0.05, 0.10), three amplification error rates (0, 0.05, 0.10), and three sequencing error rates (0, 0.01, 0.05). We assumed that amplification and sequencing errors among the four nucleotides were equally likely. From the read counts, CellCoal can also simulate the likelihood for all ten possible unphased genotypes at each SNV site, in this case under a 4-template

amplification model [83]. The input for CellPhy was either the ten genotype likelihoods at each SNV site (CellPhy-GL mode) or the unphased genotype with the maximum likelihood (ML) value (CellPhy-ML mode). The input for SCIPhi was the read counts, while for the rest of the programs, it was the ML unphased genotype. In the rare case of tied ML genotypes, we chose one of them at random.

Simulation 5: NGS doublets ("NGS-doublet")

In this case, we intended to explore the effects of doublets in the data. Settings were very similar to those for Simulation 4 but, for simplicity, we fixed the sequencing depth to 5x and explored two amplification error rates (0, 0.05), two sequencing error rates (0, 0.01), and four doublet rates (0, 0.05, 0.10, 0.20).

Simulation 6: NGS for large numbers of cells and SNVs ("NGS-large")

Finally, to assess the scalability of the tools, we simulated scenarios with 100, 500, or 1000 cells and with 1000, 10,000, or 50,000 sites. Given the mutation rate, a large number of cells, and, most importantly, the amplification and sequencing error rates, almost all sites were observed as SNVs. Settings were very similar to those specified for Simulation 5 but, for simplicity, we fixed the sequencing depth to 5x and explored only one amplification (0.05) and one sequencing (0.01) error value. We only analyzed the first 20 replicates in this scenario due to the high computational cost and prohibitive running times for several competing tools.

Settings for the phylogenetic analyses

Coding DNA into ternary genotypes Our simulations produce unphased DNA genotypes with ten possible states. However, except for CellPhy, existing tools work with an alphabet composed of 0 (homozygous for the reference allele), 1 (heterozygous), 2 (homozygous for the alternative allele), and 3 (missing genotype). Therefore, we had to encode the simulated DNA genotypes into ternary genotypes (0–3). For this, we used the true reference allele, considering that, in real life, we usually know which allele is the reference. For the sake of simplicity, we did not introduce germline mutations. Importantly, our simulations do not necessarily produce bi-allelic SNVs, as in the finite-site model multiple mutations can coincide, and amplification and sequencing errors can also result in new alleles called. CellPhy does not limit the number of alleles at an SNV site, but competing tools handle multi-allelic sites differently. We explored three ways of coding sites with more than two alleles into ternary genotypes:

- Option *keep*: transform all heterozygotes to "1" and all homozygotes for the alternative allele/s to "2". In this case, all simulated sites are held, regardless of the number of observed alleles.
- Option *remove*: eliminate sites from the data with more than two alleles. The final genotype matrix includes only bi-allelic sites.
- Option *missing*: keep only those genotypes that contain the reference allele and/or the major (most common) alternative allele. All other genotypes (containing minor alternative alleles) are considered as missing data ("3"). Therefore, the final ternary

genotype matrix includes the same number of sites as the original DNA genotype matrix.

We considered all three encoding options only in simulations 1 and 2. In the remaining simulations, we used only the "missing" option, as it maximized accuracy in most cases.

TNT settings We performed the TNT analyses using a binary data matrix in TNT format for all simulated and empirical datasets. We allowed 1000 trees to be retained for each run and performed tree searches by setting *mult = replic 100*. We stored all equally parsimonious trees and used additional ttags to store branch lengths and bootstrap support values.

OncoNEM settings We ran OncoNEM following the recommended settings in the OncoNEM vignette. We set the false positive rate as the actual genotype error for each scenario and performed a tree search for 200 iterations. Because of its heavy computational requirements and poor performance, we only run it in Simulation 1.

SASC settings We generated binary matrices as input files and set *-k* option to 1. Additionally, the false positive rate was set as the actual genotype error, and the false negative rate was set as the actual ADO rate. The command line was: *sasc -n <CELLS> -m <SNVS> -k 1 -a <ADO> -b <ERR> -E data.CellID -l -i data.sasc -p 24*. Similar to OncoNEM, due to its bad performance, SASC was only run partially in Simulation 1.

SPhyR settings We produced binary input files using a custom script and ran the kDPFC version with the false positive rate (*-a*) set to the simulated genotype error and the false negative rate (*-b*) set to the simulated ADO rate. The command line was *kDPFC data.sphyr -a <ERR> -b <ADO> -k 1 -t 24 > data.sphyr.result*. Afterwards, we used the *visualize* program to generate the output tree. The command line was *visualize data.sphyr.result -c data.snv_labels -t data.cell_labels > data.sphyr.dot*. Because it continuously crashed for most datasets, SPhyR was only run partially in simulation 1.

infSCITE settings For simulations 1–3, we ran infSCITE using a ternary data matrix composed of 0, 1, 2, and 3, as described above, and set the false positive rate (*-fd*) to be the actual genotype error for each simulated scenario. We set the false-positive rate for simulations 4–5 as the sum of the simulated sequencing and amplification error rates. For the empirical analyses, we set the false positive rate to *1e−05*. We set the remaining parameters to the default values in all runs and obtained results after running an MCMC chain with 5 million steps, a fair trade-off between runtime and apparent MCMC convergence (the best tree score barely changed after 1M iterations). We additionally set the *-transpose* option to return a tree where the single-cell samples are the leaf labels. The command line was *infSCITE -i data.infSCITE -n <SNVS> -m <CELLS> -r 1 -l 5000000 -fd <ERR> -ad 1.46e-1 -cc 1.299164e-05 -transpose -e 0.2 -o data.infSCITE.Tree*

SiFit settings For all simulated and empirical datasets, we ran SiFit for 200,000 iterations. The command line was *java -jar /SiFit.jar -m <CELLS> -n <SNVS> -r 1 -iter 200000 -df 1 -ipMat data.sf -cellNames data.names*.

SCIPhI settings Since SCIPhI requires read counts to perform joint variant calling and phylogenetic reconstruction, we ran it only for simulations 4–6. We set the mean error rate (-wildMean) for each scenario as the sum of the true sequencing and amplification error rates. The command line was *sciphi -o data.Result --in sampleNames -u 1 --ncf 0 --md 1 --mmw 4 --mnp 1 --ms 1 --mc 1 --unc true -l 200000 --seed $RANDOM data.mpileup --ese 0 --wildMean <ERR>.* To make the results comparable, for the empirical datasets we used samtools to generate mpileups for the positions previously identified by SC-Caller. These were in turn used as input for SCIPhI: *sciphi -o <SET>.SCIPhI --in <SET>-SampleNames.txt -u 0 --ncf 0 --md 0 --mmw 4 --mnp 1 --unc true -l 200000 --seed 421 <SET>.SCCaller-Positions.mpileup*.

ScisTree settings We extracted allele counts from VCF files using a custom script (available in the CellPhy's GitHub repository) and obtained genotype probabilities using the *scprob* program provided with ScisTree. We ran ScisTree v1.2.0.6 with default parameters, which we called as follows: *scistree <genotype-probability-matrix>*.

CellPhy settings For simulations 1–6, we performed a heuristic tree search starting from a single parsimony-based tree under the GT16 model. The command line for runs with ML genotypes as input was *cellphy.sh RAXML --search --msa data.phy --model GT16+FO+E --tree pars{1}*. The command line for runs where the input was genotype likelihoods (VCF) was *cellphy.sh RAXML --search --msa data.vcf --model GT16+FO --tree pars{1}*. For all empirical datasets except LS140, to take advantage of the genotype likelihood model, we used an *in-house* bash script (sc-caller-convert.sh, distributed together with CellPhy) to convert the PL field from our SC-Caller VCFs. In short, following SC-Caller authors' suggestion, we used the highest likelihood score of the first two values in the PL field (i.e., sequencing noise, amplification artifact) as the Phred-scaled genotype likelihood of the reference homozygous (0/0) genotype, and the remaining values as the likelihood for heterozygous (0/1) and alternative homozygous (1/1) genotypes, respectively. Afterward, we ran CellPhy using the following command line *cellphy.sh RAXML --all --msa data.vcf --model GT16+FO --bs-metric fbp,tbe --bs-trees 100* to perform an all-in-one analysis (ML tree inference and bootstrapping based on 100 bootstrap trees). For LS140, since we only had the genotype matrix available and these data were generated without whole-genome amplification, we ran CellPhy without the single-cell error model using the following command line *cellphy.sh RAXML --all --msa data.vcf --model GT16+FO --prob-msa off --bs-metric fbp,tbe --bs-trees 100*.

Evaluation of phylogenetic accuracy

We defined *phylogenetic accuracy* as one minus the normalized Robinson-Foulds (nRF) distance [88] between the inferred tree and the (true) simulated tree. This normalization divides the (absolute) RF distance by the total number of (internal)

branches in *both* trees. Hence, the nRF distance is a convenient metric from zero to one that reflects the proportion of branches (bipartitions of the data) correctly inferred.

Running time comparisons

We characterized the computational efficiency of CellPhy by comparing running times for all methods on six datasets from simulations 1–6 (sim1-ADO:0.50, ERR:0.10, sim2-ADO:0.10, ERR:0.05, sim3-ADO:0.15, ERR:0.10, Signature1, sim4-Number of cells:40, ADO:0.25, Amp error:0.10, Seq error: 0.05; sim6-100-Number of Cells:100, ADO:0.10, Amp error:0.05; Seq error:0.01; and sim6-500-Number of Cells:500, ADO:0.10, Amp error:0.05; Seq error:0.01) and two empirical datasets (CRC24 and L86) described below. We measured running times using the Linux/Unix "time" command, as follows: *{time ./cellphy.sh RAXML --search --msa data.vcf --model GT16+FO --tree pars{1} --prefix data-out --threads 1 ; } 2> data-CellPhyGL16.time*. We ran all analyses on a single core from an Intel Xeon E5-2680 v3 Haswell Processor 2.5 GHz with 128 Gb of RAM. For the bootstrap benchmark, the CellPhy command line was *{time ./cellphy.sh RAXML --all --msa data.vcf --model GT16+FO --tree pars{1} --bs-trees 100 --prefix data-out-Boot --threads 1 ; } 2> data-CellPhyGL16.Boot.time.* As for TNT, additional ttags were used to perform and store 100 bootstrap replicates.

Analysis of empirical data

In-house single-cell WGS data from a colorectal cancer patient (CRC24) We obtained a fresh frozen primary tumor and normal tissues from a single colorectal cancer patient (CRC24). We isolated EpCAM+ cells with a BD FACSAria III cytometer from two tumoral regions (tumor inferior (TI) section and tumor middle section (TM)) and classified them as stem or non-stem according to the stemness markers at the cell surface (stem: EpCAM+/Lgr5+/CD44-/CD166-; non-stem: EpCAM+/Lgr5-/CD44-/CD166-). We amplified the genomes of 24 cells with Ampli1 (Silicon Biosystems) and built whole-genome sequencing libraries using the KAPA (Kapa Biosystems) library kit. Each library was then sequenced at ~ 6× on an Illumina Novaseq 6000 at the National Center of Genomic Analysis (CNAG-CR; https://www.cnag.crg.eu/).

Retrieval of publicly available datasets (L86, E15, and LS140) We also analyzed three public data sets with 86, 15, and 140 cells (L86, E15, and LS140, respectively). The L86 dataset consists of targeted sequencing data from 86 cells from a metastatic colorectal cancer patient (CRC2 in [21]) that we downloaded from the Sequence Read Archive (SRA) in FASTQ format, together with paired healthy-tumor bulk cell population samples (accession number: SRP074289). The E15 dataset consists of WGS data from 15 neurons [69] from a healthy donor, downloaded from the SRA in FASTQ format, together with a bulk cell population from heart tissue (accession number: SRP041470). The LS140 dataset consists of 140 single cell-derived human hematopoietic stem and progenitor colonies from a healthy individual [15]. For this dataset, we directly downloaded the substitution calls from the Mendeley data archive (https://data.mendeley.com/datasets/yzjw2stk7f/1).

NGS data processing and variant calling We aligned single-cell and bulk reads to the human reference GRCh37 using the MEM algorithm in the BWA software [89]. The mapped reads were then independently processed for all datasets by filtering reads displaying low mapping quality, performing local realignment around indels, and removing PCR duplicates. For the tumor bulk samples (i.e., CRC24 & L86), we obtained SNV calls using the paired-sample variant-calling approach implemented in the MuTect2 software [90]. For the E15 dataset, we ran HaplotypeCaller from the Genome Analysis Toolkit (GATK) [91] software on the bulk sample from the heart tissue to identify and remove all germline variants.

In parallel, we used the single-cell SC-caller software [66] to retrieve single-cell SNV calls. In short, for each single-cell BAM, we ran SC-Caller together with the corresponding healthy bulk DNA as input under default settings. Since different amplification methods were used to generate each dataset, we defined the bias estimation interval (*--lamb*) as the average amplicon size of each amplification method—10,000 for MDA-based protocols (L86, E15) and 2000 for Ampli1 (CRC24). Besides, since the actual genomic targets of the L86 dataset were not available, we ran SC-caller on the entire exome. We applied a series of heavy filters (see below) to remove potential off-target calls. We additionally estimated copy-number variants (CNVs) for each single-cell dataset. For the sc-WGS datasets (CRC24 and E15), we obtained CNV calls with the Ginkgo software [92] using variable-length bins of around 500 kb. For the L86 dataset, we determined CNVs using CNVPanelizer, an algorithm specifically designed to infer copy number states from targeted sequencing data.

We filtered our raw single-cell VCFs by excluding short indels, SNVs with a flag other than "true" in the SO format field (i.e., showing weak evidence of being a true somatic mutation), and variable sites with an alternative read count < 3. We also excluded variable sites in which the ML genotype estimate was above 120 (Phred-scaled). Such uncertainty in the genotype call was usually associated with sites experiencing an apparent disparity in the proportion of both alleles (i.e., allelic bias). Moreover, as we are primarily interested in analyzing diploid genomic regions, we removed those SNVs located within CN variable regions.

For each dataset, we then merged single-cell VCFs using the bcftools software [93] and applied a "consensus" filter to only retain sites present in at least one cell and the bulk tumor sample, or in two cells. For the E15 dataset, we limited this "consensus" filter to somatic sites observed in at least two cells, as we classified as germline all variants observed in the bulk sample. Finally, we removed positions missing (i.e., not covered by any read) in more than 50% of the cells and SNVs comprising more than one alternative allele. For the L86 dataset, we filtered out off-target SNVs located outside exonic regions. For the LS140 dataset, we converted the binary genotype matrix into a VCF by transforming 0, 1, and NA values into 0/0, 0/1, and ./., respectively. Afterward, we removed all duplicated (non-biallelic) positions and indels.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s13059-021-02583-w.

Additional file 1. Supplementary Tables S1-S3, Figures S1-S22, and Notes S1-S3 [15, 21, 29, 51, 52, 66, 69, 73, 77, 91, 94–98].

Additional file 2. Review history.

Acknowledgements
We want to thank Debora Chantada, Pilar Alvariño, and Sonia Prado for their help in obtaining the CRC24 dataset and Phylogenomics lab members for their comments.

Review history
The review history is available as Additional file 2.

Peer review information
Barbara Cheifet was the primary editor of this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

Authors' contributions
A.K. implemented the model within RAxML-NG and ran part of the simulations. J.M.A. ran part of the simulations and performed the empirical analyses. A.S. supervised the phylogenetic implementation. D.P. conceived the CellPhy model, developed the error model, defined the experimental design, and supervised the obtention of the single-cell data. All authors contributed to manuscript writing. The authors read and approved the final manuscript.

Authors' information
Twitter handles: @JMFAlves (Joao M. Alves); @AlexisCompBio (Alexandros Stamatakis); @dposada_ (David Posada)

Funding
This work was supported by the European Research Council (ERC-617457- PHYLOCANCER awarded to D.P.) and by the Spanish Ministry of Science and Innovation - MICINN (PID2019-106247GB-I00 awarded to D.P.). D.P. receives further support from Xunta de Galicia and from FCT (PTDC/BIA-EVL/32030/2017). J.M.A. is supported by an AXA Research Fund Postdoctoral Fellowship and an AECC Investigator 2020 fellowship. This work was also financially supported by the Klaus Tschira Foundation (A.K. and A.S.).

Availability of data and materials
The source code of CellPhy is freely available at https://github.com/amkozlov/cellphy [99] under the GNU Affero GPL version 3 license. The source code version used in the manuscript can be downloaded from the Zenodo repository (https://doi.org/10.5281/zenodo.5345921) [100].
Conversion scripts and input/output files for the empirical analyses are available at https://github.com/amkozlov/cellphy-paper.
We deposited the raw single-cell whole-genome sequencing data for patient CRC24 at the National Center for Biotechnology Information (NCBI) as BioProject PRJNA789841 [101]. We additionally analyzed several published single-cell data sets [21, 69]. Raw sequencing data for these are available from the SRA database under accession numbers SRP074289 (L86) and SRP041470 (E15). We generated the genotype matrix for the LS140 dataset [15] from the substitution calls available at the Mendeley data archive (https://data.mendeley.com/datasets/yzjw2stk7f/1).

Declarations

Ethics approval and consent to participate
Tissue samples from patient CRC24 were obtained at the Galicia Sur Health Research Institute (IISGS) Biobank, member of the Spanish National Biobank Network (N° B.0000802). This study was approved by a local Ethical and Scientific Committee (CAEI Galicia 2014/015).

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Computational Molecular Evolution Group, Heidelberg Institute for Theoretical Studies, 69118 Heidelberg, Germany. [2]CINBIO, Universidade de Vigo, 36310 Vigo, Spain. [3]Department of Biochemistry, Genetics, and Immunology, Universidade de Vigo, 36310 Vigo, Spain. [4]Galicia Sur Health Research Institute (IIS Galicia Sur), SERGAS-UVIGO, Vigo, Spain. [5]Institute for Theoretical Informatics, Karlsruhe Institute of Technology, 76128 Karlsruhe, Germany.

Received: 6 May 2021 Accepted: 20 December 2021
Published online: 26 January 2022

References
1. Gawad C, Koh W, Quake SR. Single-cell genome sequencing: current state of the science. Nat Rev Genet. 2016;17:175–88.
2. Tanay A, Regev A. Scaling single-cell genomics from phenomenology to mechanism. Nature. Nature Publishing Group, a division of Macmillan Publishers Limited. All Rights Reserved.; 2017;541:331.
3. Macaulay IC, Voet T. Single cell genomics: advances and future perspectives. PLoS Genet. 2014;10:e1004126.
4. Baslan T, Hicks J. Unravelling biology and shifting paradigms in cancer with single-cell sequencing. Nat Rev Cancer. 2017;17:557–69.

5. Stegle O, Teichmann SA, Marioni JC. Computational and analytical challenges in single-cell transcriptomics. Nat Rev Genet. 2015;16:133–45.
6. Lim B, Lin Y, Navin N. Advancing cancer research and medicine with single-cell genomics. Cancer Cell. 2020;37:456–70.
7. Marioni JC, Arendt D. How single-cell genomics is changing evolutionary and developmental biology. Annu Rev Cell Dev Biol. 2017;33:537–53.
8. Wiedmeier JE, Noel P, Lin W, Von Hoff DD, Han H. Single-cell sequencing in precision medicine. Precision Medicine. Cancer Therapy. 2019:237–52.
9. Lähnemann D, Köster J, Szczurek E, McCarthy DJ, Hicks SC, Robinson MD, et al. Eleven grand challenges in single-cell data science. Genome Biol. 2020;21:31.
10. Navin NE. Cancer genomics: one cell at a time. Genome Biol. 2014;15:452.
11. Stadler T, Pybus OG, Stumpf MPH. Phylodynamics for cell biologists. Science [Internet]. 2021;371. Available from: https://doi.org/10.1126/science.aah6266
12. Wagner DE, Klein AM. Lineage tracing meets single-cell omics: opportunities and challenges. Nat Rev Genet. 2020;21:410–27.
13. Coorens THH, Moore L, Robinson PS, Sanghvi R, Christopher J, Hewinson J, et al. Extensive phylogenies of human development inferred from somatic mutations. Nature. 2021;597:387–92.
14. Lodato MA, Woodworth MB, Lee S, Evrony GD, Mehta BK, Karger A, et al. Somatic mutation in single human neurons tracks developmental and transcriptional history. Science. 2015;350:94–8.
15. Lee-Six H, Øbro NF, Shepherd MS, Grossmann S, Dawson K, Belmonte M, et al. Population dynamics of normal human blood inferred from somatic mutations. Nature. 2018;561:473–8.
16. Park S, Mali NM, Kim R, Choi J-W, Lee J, Lim J, et al. Clonal dynamics in early human embryogenesis inferred from somatic mutation. Nature. 2021;597:393–7.
17. Evrony GD, Hinch AG, Luo C. Applications of single-cell DNA sequencing. Annu Rev Genomics Hum Genet. 2021;22:171–97.
18. Quinn JJ, Jones MG, Okimoto RA, Nanjo S, Chan MM, Yosef N, et al. Single-cell lineages reveal the rates, routes, and drivers of metastasis in cancer xenografts. Science [Internet]. 2021;371. Available from: https://doi.org/10.1126/science.abc1944
19. Naxerova K, Jain RK. Using tumour phylogenetics to identify the roots of metastasis in humans. Nat Rev Clin Oncol. 2015;12:258–72.
20. Hong WS, Shpak M, Townsend JP. Inferring the origin of metastases from cancer phylogenies. Cancer Res. 2015;75:4021–5.
21. Leung ML, Davis A, Gao R, Casasent A, Wang Y, Sei E, et al. Single-cell DNA sequencing reveals a late-dissemination model in metastatic colorectal cancer. Genome Res. 2017;27:1287–99.
22. Alves JM, Prado-López S, Cameselle-Teijeiro JM, Posada D. Rapid evolution and biogeographic spread in a colorectal cancer. Nat Commun. 2019;10:5139.
23. Kuipers J, Jahn K, Beerenwinkel N. Advances in understanding tumour evolution through single-cell sequencing. Biochim Biophys Acta Rev Cancer. 2017;1867:127–38.
24. Zafar H, Navin N, Nakhleh L, Chen K. Computational approaches for inferring tumor evolution from single-cell genomic data. Current Opinion in Systems Biology. 2018;7:16–25.
25. Ross EM, Markowetz F. OncoNEM: inferring tumor evolution from single-cell sequencing data. Genome Biol. 2016;17:69.
26. Jahn K, Kuipers J, Beerenwinkel N. Tree inference for single-cell data. Genome Biol. 2016;17:86.
27. Kuipers J, Jahn K, Raphael BJ, Beerenwinkel N. Single-cell sequencing data reveal widespread recurrence and loss of mutational hits in the life histories of tumors. Genome Res. 2017;27:1885–94.
28. Zafar H, Tzen A, Navin N, Chen K, Nakhleh L. SiFit: inferring tumor trees from single-cell sequencing data under finite-sites models. Genome Biol. 2017;18:178.
29. Singer J, Kuipers J, Jahn K, Beerenwinkel N. Single-cell mutation identification via phylogenetic inference. Nat Commun. 2018;9:5144.
30. El-Kebir M. SPhyR: tumor phylogeny estimation from single-cell sequencing data under loss and error. Bioinformatics. 2018;34:i671–9.
31. Ciccolella S, Ricketts C, Soto Gomez M, Patterson M, Silverbush D, Bonizzoni P, et al. Inferring cancer progression from single-cell sequencing while allowing mutation losses. Bioinformatics. 2021;37:326–33.
32. Satas G, Zaccaria S, Mon G, Raphael BJ. SCARLET: single-cell tumor phylogeny inference with copy-number constrained mutation losses. Cell Systems. 2020;10:323–32.e8.
33. Wu Y. Accurate and efficient cell lineage tree inference from noisy single cell data: the maximum likelihood perfect phylogeny approach. Bioinformatics. 2020;36:742–50.
34. Chen Z, Gong F, Wan L, Ma L. RobustClone: a robust PCA method for tumor clone and evolution inference from single-cell sequencing data. Bioinformatics. 2020;36:3299–306.
35. Sadeqi Azer E, Rashidi Mehrabadi F, Malikić S, Li XC, Bartok O, Litchfield K, et al. PhISCS-BnB: a fast branch and bound algorithm for the perfect tumor phylogeny reconstruction problem. Bioinformatics. 2020;36:i169–76.
36. Zafar H, Navin N, Chen K, Nakhleh L. SiCloneFit: Bayesian inference of population structure, genotype, and phylogeny of tumor clones from single-cell genome sequencing data. Genome Res. 2019;29:1847–59.
37. Roth A, McPherson A, Laks E, Biele J, Yap D, Wan A, et al. Clonal genotype and population structure inference from single-cell tumor sequencing. Nat Methods. 2016;13:573–6.
38. Malikic S, Jahn K, Kuipers J, Sahinalp SC, Beerenwinkel N. Integrative inference of subclonal tumour evolution from single-cell and bulk sequencing data. Nat Commun. 2019;10:2750.
39. Salehi S, Steif A, Roth A, Aparicio S, Bouchard-Côté A, Shah SP. ddClone: joint statistical inference of clonal populations from single cell and bulk tumour sequencing data. Genome Biol. 2017;18:44.
40. Chen Z, Gong F, Wan L, Ma L. BiTSC2: Bayesian inference of tumor clonal tree by joint analysis of single-cell SNV and CNA data [Internet]. bioRxiv. 2020 [cited 2021 Aug 11]. p. 2020.11.30.380949. Available from: https://www.biorxiv.org/content/10.1101/2020.11.30.380949v1
41. Baghaarabani L, Goliaei S, Foroughmand-Araabi M-H, Shariatpanahi SP, Goliaei B. Conifer: clonal tree inference for tumor heterogeneity with single-cell and bulk sequencing data [Internet]. Research Square; 2021. Available from: https://www.researchsquare.com/article/rs-263502/v1

42. Sadeqi Azer E, Haghir Ebrahimabadi M, Malikić S, Khardon R, Sahinalp SC. Tumor phylogeny topology inference via deep learning. iScience. 2020;23:101655.
43. Weber LL, El-Kebir M. Distinguishing linear and branched evolution given single-cell DNA sequencing data of tumors. Algorithms Mol Biol. 2021;16:14.
44. Gao Y, Gaither J, Chifman J, Kubatko L. A phylogenetic approach to inferring the order in which mutations arise during cancer progression [Internet]. bioRxiv. 2021 [cited 2021 Aug 10]. p. 2020.05.06.081398. Available from: https://www.biorxiv.org/content/10.1101/2020.05.06.081398v5
45. Miura S, Huuki LA, Buturla T, Vu T, Gomez K, Kumar S. Computational enhancement of single-cell sequences for inferring tumor evolution. Bioinformatics. 2018;34:i917–26.
46. Ramazzotti D, Angaroni F, Maspero D, Ascolani G. Longitudinal cancer evolution from single cells. bioRxiv [Internet]. biorxiv.org. 2020; Available from: https://www.biorxiv.org/content/10.1101/2020.01.14.906453v2.abstract.
47. Mallory XF, Edrisi M, Navin N, Nakhleh L. Assessing the performance of methods for copy number aberration detection from single-cell DNA sequencing data. PLoS Comput Biol. 2020;16:e1008012.
48. Kozlov AM, Darriba D, Flouri T, Morel B, Stamatakis A. RAxML-NG: a fast, scalable, and user-friendly tool for maximum likelihood phylogenetic inference. Bioinformatics. 2019;35:4453–5.
49. Goloboff PA, Catalano SA. TNT version 1.5, including a full implementation of phylogenetic morphometrics. Cladistics. 2016;32:221–38.
50. Tavaré S. Some probabilistic and statistical problems in the analysis of DNA sequences. Lectures on mathematics in the life sciences. 1986;17:57–86.
51. Felsenstein J. Inferring phylogenies. Sinauer associates Sunderland, MA; 2004.
52. Felsenstein J. Evolutionary trees from DNA sequences: a maximum likelihood approach. J Mol Evol. 1981;17:368–76.
53. Stamatakis A. RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. Bioinformatics. 2014;30:1312–3.
54. Zhou X, Shen X-X, Hittinger CT, Rokas A. Evaluating fast maximum likelihood-based phylogenetic programs using empirical phylogenomic data sets. Mol Biol Evol. 2018;35:486–503.
55. Felsenstein J. Confidence limits on phylogenies: an approach using the bootstrap. Evolution. 1985;39:783–91.
56. Lemoine F, Domelevo Entfellner J-B, Wilkinson E, Correia D, Dávila Felipe M, De Oliveira T, et al. Renewing Felsenstein's phylogenetic bootstrap in the era of big data. Nature. 2018;556:452–6.
57. Yang Z, Kumar S, Nei M. A new method of inference of ancestral nucleotide and amino acid sequences. Genetics. 1995;141:1641–50.
58. Wickham H. ggplot2: elegant graphics for data analysis: Springer; 2016.
59. Hartenstein V. Stem cells in the context of evolution and development. Dev Genes Evol. 2013;223:1–3.
60. Reya T, Morrison SJ, Clarke MF, Weissman IL. Stem cells, cancer, and cancer stem cells. Nature. 2001;414:105–11.
61. Kreso A, Dick JE. Evolution of the cancer stem cell model. Cell Stem Cell. 2014;14:275–91.
62. Li J, Zhang N, Zhang R, Sun L, Yu W, Guo W, et al. CDC5L promotes hTERT expression and colorectal tumor growth. Cell Physiol Biochem. 2017;41:2475–88.
63. Ma SSQ, Srivastava S, Llamosas E, Hawkins NJ, Hesson LB, Ward RL, et al. ROR2 is epigenetically inactivated in the early stages of colorectal neoplasia and is associated with proliferation and migration. BMC Cancer. 2016;16:508.
64. Pan H, Pan J, Song S, Ji L, Lv H, Yang Z. EXOSC5 as a novel prognostic marker promotes proliferation of colorectal cancer via activating the ERK and AKT pathways. Front Oncol. 2019;9:643.
65. Li S-R, Gyselman VG, Lalude O, Dorudi S, Bustin SA. Transcription of the inositol polyphosphate 1-phosphatase gene (INPP1) is upregulated in human colorectal cancer. Mol Carcinog. 2000;27:322–9.
66. Dong X, Zhang L, Milholland B, Lee M, Maslov AY, Wang T, et al. Accurate identification of single-nucleotide variants in whole-genome-amplified single cells. Nature Methods. 2017;14:491–3.
67. Alhopuro P, Phichith D, Tuupanen S, Sammalkorpi H, Nybondas M, Saharinen J, et al. Unregulated smooth-muscle myosin in human intestinal neoplasia. Proc Natl Acad Sci U S A. 2008;105:5513–8.
68. Romero-Pérez L, Surdez D, Brunet E, Delattre O, Grünewald TGP. STAG mutations in cancer. Trends Cancer Res. 2019;5:506–20.
69. Evrony GD, Lee E, Mehta BK, Benjamini Y, Johnson RM, Cai X, et al. Cell lineage analysis in human brain using endogenous retroelements. Neuron. 2015;85:49–59.
70. Demeulemeester J, Dentro SC, Gerstung M, Van Loo P. Biallelic mutations in cancer genomes reveal local mutational determinants [Internet]. bioRxiv. 2021; Available from: https://www.biorxiv.org/content/10.1101/2021.03.29.437407v1.abstract.
71. Weber LL, Sashittal P, El-Kebir M. doubletD: detecting doublets in single-cell DNA sequencing data. Bioinformatics. 2021;37:i214–21.
72. Zhu Y, Sengupta S, Wei L, Yang S, Ji Y. Tumor Evolution Decoder (TED): unveiling tumor evolution based on mutation profiles of subclones or single cells [Internet]. bioRxiv. 2019; Available from: http://biorxiv.org/lookup/doi/10.1101/633610.
73. Yang Z. Molecular evolution: a statistical approach. Oxford: Oxford University Press; 2014.
74. Huang L, Ma F, Chapman A, Lu S, Xie XS. Single-cell whole-genome amplification and sequencing: methodology and applications. Annu Rev Genomics Hum Genet. 2015;16:79–102.
75. Paradis E, Schliep K. ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R. Bioinformatics. 2019;35:526–528.
76. Yu G, Smith DK, Zhu H, Guan Y, Lam TT. ggtree: an r package for visualization and annotation of phylogenetic trees with their covariates and other associated data. Methods in Ecology and Evolution. 2017;8:28–36.
77. Zafar H, Wang Y, Nakhleh L, Navin N, Chen K. Monovar: single-nucleotide variant detection in single cells. Nat Methods. 2016;13:505–7.
78. Robinson DF. Comparison of labeled trees with valency three. J Combin Theory Ser B. 1971;11:105–19.
79. Kozlov O. Models, optimizations, and tools for large-scale phylogenetic inference, handling sequence uncertainty, and taxonomic validation [Dr-Ing.]. In: Stamatakis A, Posada D, editors. Karlsruher Institut für Technologie (KIT); 2018.
80. Fletcher R. Practical methods of optimization [Internet]. 2000. Available from: https://doi.org/10.1002/9781118723203

81. Brent RP. An algorithm with guaranteed convergence for finding a zero of a function. The Computer Journal. 1971;14: 422–5.
82. Lutteropp S, Kozlov AM, Stamatakis A. A fast and memory-efficient implementation of the transfer bootstrap. Bioinformatics. 2020;36:2280–1.
83. Posada D. CellCoal: coalescent simulation of single-cell sequencing samples. Mol Biol Evol. 2020;37:1535–42.
84. Martincorena I, Campbell PJ. Somatic mutation in cancer and normal cells. Science. 2015;349:1483–9.
85. Kimura M. The number of heterozygous nucleotide sites maintained in a finite population due to steady flux of mutations. Genetics. 1969;61:893–903.
86. Alexandrov LB, Nik-Zainal S, Wedge DC, Aparicio SAJR, Behjati S, Biankin AV, et al. Signatures of mutational processes in human cancer. Nature. 2013;500:415–21.
87. Alexandrov LB, Jones PH, Wedge DC, Sale JE, Campbell PJ, Nik-Zainal S, et al. Clock-like mutational processes in human somatic cells. Nat Genet. 2015;47:1402–7.
88. Robinson DF, Foulds LR. Comparison of phylogenetic trees. Math Biosci. 1981;53:131–47.
89. Li H. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM [Internet]. arXiv [q-bio.GN]. 2013; Available from: http://arxiv.org/abs/1303.3997.
90. Cibulskis K, Lawrence MS, Carter SL, Sivachenko A, Jaffe D, Sougnez C, et al. Sensitive detection of somatic point mutations in impure and heterogeneous cancer samples. Nature Biotechnology. 2013;31:213–9.
91. Poplin R, Ruano-Rubio V, DePristo MA, Fennell TJ. Scaling accurate genetic variant discovery to tens of thousands of samples. BioRxiv [Internet]. biorxiv.org. 2018; Available from: https://www.biorxiv.org/content/10.1101/201178v3.abstract.
92. Garvin T, Aboukhalil R, Kendall J, Baslan T, Atwal GS, Hicks J, et al. Interactive analysis and assessment of single-cell copy-number variations. Nat Methods. 2015;12:1058–60.
93. Li H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. Bioinformatics. 2011;27:2987–93.
94. Hård J, Al Hakim E, Kindblom M, Björklund ÅK, Sennblad B, Demirci I, et al. Conbase: a software for unsupervised discovery of clonal somatic mutations in single cells through read phasing. Genome Biol. 2019;20:68.
95. Luquette LJ, Bohrson CL, Sherman MA, Park PJ. Identification of somatic mutations in single cell DNA-seq using a spatial model of allelic imbalance. Nat Commun. 2019;10:3908.
96. Lähnemann D, Köster J, Fischer U, Borkhardt A, McHardy AC, Schönhuth A. Accurate and scalable variant calling from single cell DNA sequencing data with ProSolo. Nat Commun. 2021;12:6744.
97. Roch S. A short proof that phylogenetic tree reconstruction by maximum likelihood is hard [Internet]. IEEE/ACM Transactions on Computational Biology and Bioinformatics. 2006. p. 92–4. Available from: https://doi.org/10.1109/tcbb.2006.4
98. Stamatakis A, Kozlov AM, Kozlov A. Efficient maximum likelihood tree building methods. hal.archives-ouvertes.fr; 2020; Available from: https://hal.archives-ouvertes.fr/hal-02535285/
99. Kozlov A, Alves JM, Stamatakis A, Posada D. CellPhy: accurate and fast probabilistic inference of single-cell phylogenies from scDNA-seq data. Github. 2021; https://github.com/amkozlov/cellphy.
100. Kozlov A, Alves JM, Stamatakis A, Posada D. CellPhy v0.9.1 source code. Zenodo. 2021. https://doi.org/10.5281/zenodo.5345921.
101. Kozlov A, Alves JM, Stamatakis A, Posada D. CellPhy: accurate and fast probabilistic inference of single-cell phylogenies from scDNA-seq data. Datasets. Bioproject. (2021) https://www.ncbi.nlm.nih.gov/bioproject/PRJNA789841

Publisher's Note