nature protocols

https://doi.org/10.1038/s41596-023-00913-9

Protocol

Check for updates

CONIPHER: a computational framework for scalable phylogenetic reconstruction with error correction

Supplementary Information

Contents

Supplementary Methods

1. Recommended preprocessing of mutations for CONIPHER

Mutation calling. CONIPHER is a method that takes processed mutation data as input. For mutation clustering and tree building on multi-sample, bulk, whole-exome sequencing data it is recommended to follow a pipeline similar to that used to process the TRACERx421 NSCLC cohort[1,2]. The details of the mutation preprocessing pipeline used in the TRACERx421 study were broadly similar to the TRACERx100 cohort[3], and a detailed description can be found in our companion manuscript[1]. In brief, somatic variant calling was performed using VarScan2[4] and MuTect[5] and mutations present at a variant count of < 10 are filtered out. In particular, multi-sample sequencing was leveraged in order to "force call" presence of a low-confidence mutation in tumour samples, if the mutation was called as confidently present in a distinct sample of the same tumour. It is strongly recommended to implement this latter step prior to running CONIPHER for more accurate results.

2. CONIPHER method additional details

This section describes the details of the CONIPHER method to reconstruct mutation clusters and tumour phylogenetic trees. It is noted that CONIPHER tree building is compatible with clustering performed using other methods provided the input is formatted as specified in this protocol. The CONIPHER mutation clustering method is designed as an extension of the existing algorithm PyClone (v.0.13.1)[6]. We find that the implemented extensions improve the performance of mutation clustering, based on a simulated ground truth dataset (Figure 2). CONIPHER clustering is therefore recommended, but not required.

2.1 Mutation clustering

The CONIPHER mutation clustering method takes as input a mutation table in long format, whereby each row describes one mutation in one sample (see PROCEDURE, Figure 4). The clustering comprises four main steps: (1) copy number preprocessing, (2) pre-clustering mutations based on presence/absence, (3) Dirichlet clustering using the existing PyClone algorithm[6], and (4), post-processing the inferred mutation clusters to remove clusters driven by subclonal copy number[3] and merge mutation clusters that were erroneously assigned to be subclonal. These steps are outlined briefly below.

Copy number preprocessing. The first step in CONIPHER mutation clustering is to estimate the PhyloCCF of each mutation. This calculation is performed as in previous work[3], whereby the standard cancer cell fraction (CCF) is computed by transforming the VAF by the expected mutation copy number and tumour purity[3,7–9] and subsequently correcting for subclonal copy number alterations[3] to generate the estimated PhyloCCF of each mutation.

Pre-clustering mutations. The second step is to separate mutations based on their presence or absence classification in each tumour sample into groups, whereby presence is defined as at least one mutant read. This step was found to improve over-clustering of mutations. In this step, an indel- and sample-specific VAF correction factor is implemented as previously described[3] in order to prevent potentially incorrectly estimated indel VAFs leading to separate mutation clusters. In brief, for each tumour sample, each indel PhyloCCF (and mutation copy number) is multiplied by that indel sample-specific correction factor. The correction factor is computed by dividing the median mutation PhyloCCF of ubiquitous SNVs (i.e., present in all tumour samples) by the median mutation PhyloCCF of ubiquitous indels, under the assumption that the majority of ubiquitous mutations (SNVs and indels) will be truncal. Any presence/absence group of mutations consisting of less than 5 mutations were not clustered using PyClone.

PyClone clustering. PyClone[6] is applied to each group of mutations independently. By default, CONIPHER runs PyClone (v.0.13.1) with 10,000 iterations and a burn-in of 1,000 mutations, as well as using the state where the reference prior was set to normal and the variant prior was set to “BB”. If ≥50 mutations are present in a presence/absence mutation group the maximum number of clusters parameter in PyClone is set to 10 for that group, and otherwise to the number of mutations in that group divided by 5 and rounded down to the nearest integer to avoid overclustering. All other parameters are set to default values.

Post-processing clusters. CONIPHER finally interrogates the clusters inferred by PyClone to check whether mutation clusters were driven by heterogeneous copy number events. First, mutations that have low PhyloCCF values, or are absent, in one or more samples are investigated to identify whether these could be driven by copy number loss events, as previously described[3]. Specifically, a statistical test is carried out to evaluate whether copy number loss coincides with lower PhyloCCF levels. If two copy number states only are present, a one-sided Wilcoxon test is used. If more than two copy number states are present, a one-sided general test of independence is performed (using the function independence_test() in R from the “coin” package[10] with default parameters). If the majority (>85%) of mutations in a cluster are determined to be driven by a subclonal copy number loss, the cluster is removed from further analysis.

Secondly, a cluster merging step is implemented, in which the regional subclonal copy number correction is re-evaluated: if the initial subclonal copy number correction results in an additional mutation cluster which would not be present without subclonal copy number correction, then this additional cluster is considered a false mutation cluster, and thereby merged such that no additional cluster is created.

2.2 Phylogenetic tree building: Determining cluster nesting

The CONIPHER tree building method similarly takes as input a mutation table in long format, whereby each row describes one mutation in one sample, with an assignment of each mutation to a cluster (Figure 4 and PROCEDURE). The tree building stage assumes mutation clustering has already been run, by either CONIPHER or another method.

The first stage of the CONIPHER tree building stage determines the nesting possibility of every pair of mutation clusters inferred in the clustering stage. This stage of the tree building method returns a nesting matrix.

Computing cluster confidence intervals. CONIPHER first computes confidence intervals summarising the distribution of the PhyloCCF values for each mutation cluster inferred in the clustering stage (Figure 1e). By default, these are computed via bootstrap sampling of the mutation PhyloCCF values and subsequently calculating the bootstrapped 99% upper and lower confidence intervals. In the case of a total tumour mutation burden (TMB) of less than 5000, 5000 bootstrapping iterations are run. In the case of a total TMB > 5000, the number of bootstrapping iterations run is the TMB value rounded up to the closest 1000.

Testing cluster distributions. Then, to test whether inferred mutation clusters have significantly different PhyloCCF distributions from one another, the mutation PhyloCCF distributions for each pair of clusters are compared with two one-sided Wilcoxon tests (Figure 1f). A significant difference is classified if the Wilcoxon p-value is <0.05 in either direction. The truncal cluster is inferred as the cluster that is able to nest all other clusters. All clusters that are not truncal are classified as subclonal mutation clusters.

Cluster clonality testing. In order to ensure that intra-tumour heterogeneity is not being over-estimated, CONIPHER next classifies each mutation cluster as 'absent', 'subclonal' or 'clonal' within each tumour sample, based on whether the confidence intervals (CIs) of the computed (bootstrapped) PhyloCCF distributions for each cluster overlap with the confidence intervals of the truncal cluster (Figure 1f). Specifically, firstly, we classify as 'clonal' every cluster in a tumour sample whose mutations have a PhyloCCF not significantly different than the PhyloCCF of the mutations in the truncal cluster within the same tumour sample (tested using a one-sided Wilcoxon test, p-value = 0.05). We also classify as clonal in each tumour sample every cluster whose 95% CI of the PhyloCCF of its mutations overlaps with the 95% CI of the PhyloCCF of the mutations in the truncal cluster, with a minimum threshold of 0.9 used for the lower 95% CI of truncal mutations.

Secondly, we define as ‘subclonal’ every mutation cluster in a tumour sample whose mean PhyloCCF across the corresponding mutations in that sample was strictly positive and not clonal (i.e., the mutation cluster does not pass the previous tests).

Lastly, any remaining mutation cluster is defined as absent in a tumour sample otherwise.

Merging clusters. If any subclonal cluster is classified as ‘clonal’ within every tumour sample, this subclonal cluster is merged with the truncal cluster, and cluster nesting is subsequently re-run. Additionally, CONIPHER searches for cases where subclonal copy number correction creates an additional cluster, and (optionally) merges this cluster with its parent. This correction is also performed in mutation clustering. Our algorithm includes this step additionally in the tree building stage as a failsafe.

Removing genomically localised clusters. A sampling strategy is performed to test whether the mutations within each subclonal cluster are more genomically localised than expected based on a background distribution based on clonal mutations (defined below). In such cases the subclonal cluster is considered to be driven by a missed subclonal copy number event rather than being driven by a subclone expansion and is removed (Figure 1f).

Specifically, let the number of mutations observed in each subclonal cluster be m_u, where $u \in \{1, ..., S_T\}$ and where S_T is the set of subclonal clusters for tree T. For a subclonal cluster u, m_u of the chromosome IDs for the truncal cluster mutations are sampled without replacement and the number of unique chromosome IDs summarised. This process is repeated 10,000 times to yield a background distribution, N_i, where $i \in \{1, ..., 10000\}$, for the expected number of chromosomes on which m_u subclonal cluster mutations would be expected to be found. Having established this background distribution, the observed number of chromosomes covered by the subclonal cluster, N_u, is compared to this simulated background distribution and determined to be significantly lower than expected by chance if (i) $p < 0.01$, where the empirical p-value is taken as $p = \sum_i \mathbb{1}_{N_i < N_u} / 10000$, and (ii) $(\sum_i N_i / 10000) / N_u > 2$. In such cases, the subclonal cluster u is removed.

2.3 Phylogenetic tree building: Determining a potential phylogenetic tree structure

The second stage of the CONIPHER tree building method is creating a tree structure from the nesting matrix, using an algorithm based on the established sum condition[11–13] and crossing rules[12,14]. Our method aims to identify the phylogenetic tree that best fits the inferred mutation cluster nestings and is supported by the most mutations possible (Figure 1g).

To begin with, the nesting matrix is represented as a directed ancestral graph, whereby each node corresponds to a mutation cluster, and each directed edge ($u \rightarrow v$) indicates that node u could be an ancestor of node v[15]. In order to obtain a tree from the ancestral graph, any edges from an ancestor node u to descendent node w are 'pruned' if $u \rightarrow v \rightarrow w$ and $u \rightarrow w$. This aims to remove multiple parent nodes leading to the same child node.

Following this pruning step, the resulting "test tree" is checked for any of the following issues: (i) graph cycles and (ii) PhyloCCF values of clusters at any particular tree level exceeding a set threshold, where we define the tree level of a cluster $u \in C_T$, where C_T is the set of clusters that are nodes on the tree T, as follows. Suppose the number of generations separating the trunk node of the tree u_0 and node u_j is j. Then we define the tree level of u_j as j. That is, the trunk will have tree level 0, its child will have tree level 1, etc. For issue (ii), clusters are labelled as issue clusters if the sum of PhyloCCFs at their tree level exceeds the upper confidence limit of the truncal cluster + a buffer set to 10% as default. If any of these issues are identified, clusters are iteratively removed from the tree, beginning from clusters with the smallest number of mutations to those with a higher number of mutations. For each removed cluster, the resulting tree structure is re-tested for issues. This procedure is carried out until a directed tree is obtained, with no issue clusters, and the result classified as the default tree.

2.4 Phylogenetic tree building: Enumerating alternative phylogenetic trees

The third stage of the phylogenetic tree building method is to identify all alternative phylogenetic tree structures $T \in \mathcal{T}$ that fit the PhyloCCF distribution nestings, and subsequently rank them according to how well they explain the data (Figure 1h). To do this, CONIPHER fixes the set of clusters comprising the default tree as the final set of tree nodes C_T and determines a subset of these clusters C_M that could be moved to descend from a different parent node in the tree without violating the sum condition. CONIPHER enumerates all combinations of clusters in subset C_M and iterates through these combinations by moving the current cluster combination and testing for cluster issues at each iteration (as described above in Supplementary Methods 2.3, Phylogenetic tree building: Determining a potential phylogenetic tree structure). In order to restrict the search space when the number of clusters is large, a constraint is set to move up to five clusters simultaneously, until all potential alternative phylogenetic trees have been identified.

CONIPHER next aims to summarise the multiple tree solutions using two methods (Figure 1i). First, the trees are ranked based on the extent to which the evolutionary constraints imposed by the tree structure are being violated. This is measured based on the amount of error introduced by nesting the cluster PhyloCCF distributions. The error metric used is the Sum Condition Error (SCE) and is computed as follows. Suppose we have a set of tumour samples S and a set of final tree clusters C_T. We denote the PhyloCCF of cluster $u \in C_T$ in sample $i \in S$ as F^i_u. Suppose parent cluster u has children nodes $v_1, ..., v_K$. Then, the Sum Condition[11] states that in each sample i, the sum of all children PhyloCCFs is constrained by the PhyloCCF of the parent node:

$$\sum_{k=1}^{K} F^i_{v_k} \leq F^i_u \ , \forall u \in C_T : u \rightarrow v, v \in C_T \ .$$

Hence for a particular alternative tree $T \in \boldsymbol{\mathcal{T}}$, suppose we have parent clusters $u \in C_P$ where $C_P \subset C_T$ is the set of parent nodes (i.e. all clusters which are not leaf nodes). Then, defining $F^i_{u,k} := \sum_{k=1}^{K} F^i_{v_k}$ as the sum of PhyloCCF values F of all children v_k of cluster u in sample i, we compute the sum condition error as the average amount that the sum condition is violated across all parent clusters $u \in C_P$ across all tumour samples $i \in S$.

$$SCE = \frac{1}{|S|} \frac{1}{|C_P|} \sum_{u \in C_P} \sum_{i \in S} \mathbb{1}_{F^i_u < F^i_{u,k}} (F^i_{u,k} - F^i_u)$$

For each tumour case, the SCE score is computed for all alternative phylogenetic trees, and ranked from lowest error to highest error. The tree generating the lowest SCE is ranked as the tree that fits the data best.

CONIPHER additionally computes a second metric summarising the solution space of potential alternative phylogenies: the edge probability score. This metric returns a probability score of each alternative tree, based on the total probability of all the edges $(u \rightarrow v)$ comprising that tree. Specifically, to determine the probability of parent node u leading to child node v, we compute the empirical probability, e_v, of each edge $(u \rightarrow v)$ across all alternative trees $T \in \mathcal{T}$:

$$e_v = \frac{1}{|\mathcal{T}|} \sum_{\mathcal{T}} \mathbb{1}_{(u \rightarrow v)}$$

e_v can be interpreted as the fraction of alternative trees containing that edge. Then, to summarise the total edge probability score for each alternative tree $T \in \mathcal{T}$, CONIPHER calculates the log product of the edge probabilities of the edges comprising that tree, weighted by the number of mutations on that edge. Specifically, suppose cluster $v \in C_T$ is supported by m_v mutations. Then the edge probability score, L, for tree $T \in \mathcal{T}$ is calculated as follows:

$$L_T = \sum_{v \in C_T} m_v log(e_v).$$

For each tumour case, the edge probability score is computed for all alternative phylogenetic trees, and a rank is produced, from highest to lowest edge probability. The tree generating the highest edge probability is interpreted as the tree that summarises the solution space the best.

2.5 Phylogenetic tree building: Estimating clone proportions

Subclone proportions within each tumour sample are automatically generated by CONIPHER (Figure 1j). This computation is performed using a top-down approach. Firstly, all mutation clusters that have been classified as ‘clonal’ within a particular tumour sample (as described in

Supplementary Methods 2.2, Phylogenetic tree building: Determining cluster nesting) are assigned to have a PhyloCCF = 100 within this sample. Then, the subclone proportions are computed for each tumour sample independently, in a top-down manner as follows. Beginning at the truncal cluster and iterating through parent nodes, ordered by tree level, we compute the clone proportion of the current parent node u in sample i as the difference in PhyloCCF values (mean mutation PhyloCCF per cluster) between the sum of all child clusters and the current parent node: $F^{i}_{u} - \sum_{k=1}^{K} F^{i}_{v_k}$. If the sum condition is violated, children PhyloCCF values are rescaled to sum exactly to the PhyloCCF of the parent node. This procedure is repeated from the trunk to the leaf nodes of the tree, within each tumour sample, until all subclone proportions are determined.

3. Exploring removed clusters in the TRACERx421 cohort

To explore whether our removal of genomically clustered clones in the CONIPHER clustering step led to an enrichment for removal of hypermutation events, we investigated the trinucleotide context of the mutations in the reconstructed tumour phylogenetic trees in the TRACERx421 primary NSCLC cohort[1] (n = 403 trees) that fall within one of the following 4 groups, (1) kept truncal mutations (n_{mut} = 195377), (2) kept subclonal mutations (n_{mut} = 107537), (3) excluded mutations in copy number driven clusters (genomically localised clusters) (n_{mut} = 1695), and (4) excluded mutations in tree removed clusters (n_{mut} = 7345) (Supplementary Figure 1a, b). We ran deconstructSigs[16] on each of these groups in order to deconvolve the mutational signatures explaining the mutational profiles in each group (Supplementary Figure 1c). Additionally, deconstructSigs was run on kept truncal mutations individually for each tumour in the TRACERx421 primary cohort of phylogenetic trees (n=403), and the mean signature weights output across all tumours was computed. We observed that the mutations excluded as part of copy number removed clusters exhibited a mutational signature distribution that reflected that of the kept truncal mutations, indicating that these clusters likely reflect truncal mutations in a genomic region subject to a missed subclonal copy number event.

4. Benchmarking the performance of CONIPHER using simulations

To assess the performance of CONIPHER, we used a simulation framework and simulated datasets introduced within the TRACERx 421 study[1]. The TRACERx simulation framework was introduced to model the evolution of somatic SNVs, as well as the evolution of other frequent genetic alterations, such as truncal/subclonal somatic copy number alterations (SCNAs including gains, losses, copy-neutral losses of heterozygosity (LOHs), etc.) and truncal/subclonal whole genome duplications (WGDs)[17]. Importantly, the framework specifically models the interactions between these different events, allowing us to model complex but frequent evolutionary events such as SNVs with different mutation multiplicities (i.e. number of copies of a SNV within the same cancer cells) or mutation losses (i.e. SNVs that are deleted as a result of deletions affecting the mutant allele). Moreover, the simulations were generated based on the parameters of the TRACERx sequencing cohort, thus representing a simulated cohort grounded in biological reality. Overall, the TRACERx simulation framework is composed of four steps, briefly summarised below, and is publicly available in GitHub at https://github.com/zaccaria-lab/TRACERx_simulation_tool.

4.1 The TRACERx simulation framework of tumour evolution

The first step is the simulation of the topology of a tumour phylogenetic tree $T = (V, E)$ with a fixed number $n = |V|$ of tumour clones, which has been sampled from $\{8, \ldots, 16\}$, $\{12, \ldots, 24\}$, and $\{22, \ldots, 30\}$ for the low (2-3 samples), medium (4-7 samples) and high (> 7 samples) sample groups respectively. Specifically, a random tree topology is simulated by randomly and iteratively pruning a full rooted binary tree with n leaves until it only contains n nodes. In such a resulting tree, the root represents the most recent common ancestor (MRCA) of the tumour and an additional node is added and linked to the root through an edge defined as the trunk to represent the normal diploid germline ancestor.

The second step is the simulation of SNVs and SCNAs to label the evolutionary branches of the simulated tree T, whose numbers have been sampled from the corresponding distributions within the TRACERx421 sequencing cohort of 421 primary NSCLCs. According to these numbers, SNVs are thus assigned to the edges of T by preserving the chosen proportions of truncal vs subclonal mutations. Similarly, SCNAs are also assigned to the edges of T and randomly assigned to one random allele. Based on previous models of SCNA evolution[18], we model SCNAs such that each copy-number gain increases by one copy of one or both allele(s) of the corresponding locus, whereas copy-number deletions decrease the copies by one and, when reaching zero copies, they result in an irreversible state of LOH. Moreover, we model each WGD as an event that doubles the copy number of every allele present at one or more copies. Overall, we model the most frequent copy number states observed in previous pan-cancer sequencing studies[8], including allele-specific copy numbers $\{2, 1\}$, $\{3, 1\}$, $\{4, 1\}$, $\{3, 2\}$, $\{4, 2\}$ for gains and $\{1, 0\}$, $\{2, 0\}$ for deletions. Note that each copy number event can affect the multiplicity of SNVs when the event is assigned to the same allele harbouring the SNV. Moreover, SNVs and SCNAs are assigned in order to respect two of the most common evolutionary assumptions used in existing methods: (1) the infinite sites assumption, so that every SNV occurs only once in tumour evolution, and (2) the constant mutation multiplicity assumption, so that every SNV has the same mutation multiplicity across different tumour clones. Lastly, the order in which the events are applied is randomly chosen to simulate SNVs occurring both before and after SCNAs and WGDs.

The third step is the simulation of bulk tumour samples by mixing different subsets of the tumour clones among those generated in T. Specifically, we model each sample as a mixture of normal diploid cells and cancer cells belonging to $\hat{n} \in \{3, ..., 8\}$ randomly selected tumour clones. We model the mixture of normal and cancer cells in each generated sample by defining the tumour purity $\mu \in [0, 1]$ as the fraction of tumour cells within the sample and we represent the clone proportion $u_i \in [0, 1]$ of every clone i as the fraction of cancer cells that belong to tumour clone i from T. As such, we simulate a bulk tumour sample that is composed of $1 - \mu$ normal diploid cells and μ tumour cells obtained from $\hat{n}$ tumour clones chosen uniformly at random such that $\Sigma_{i \in \{1, ..., \hat{n}\}} u_i = 1$. While the number k of tumour samples and the tumour purity μ of each sample are sampled from the TRACERx distributions, we sample the clone proportions $u_1, ..., u_{\hat{n}}$ as values drawn from a Dirichlet distribution with uniform concentration parameters, i.e., $u_1, ..., u_{\hat{n}} \sim Dirichlet(\vec{1})$, as done in previous studies[18,19].

The last step is the simulation of DNA sequencing data for the generated bulk samples. For every genomic locus p from every bulk tumour sample s, we aim to simulate the total number of sequencing reads $t_{p,s} \in N$ and the number of variant reads $v_{p,s} \in N$. First, as in previous cancer sequencing studies[19], we model $t_{p,s}$ as a Poisson distribution based on the expected total number of reads, which corresponds to $\frac{f_{p,s}}{\rho_s} \gamma_s$ where $f_{p,s}$ is the fractional copy number (the average total copy number of p across all cells present in s, i.e., $f_{p,s} = \sum_{i \in \{1, \dots, \hat{n}\}} u_i (x_{i,p} + y_{i,p})$ for the allele-specific copy numbers $x_{i,p}$, $y_{i,p}$), where ploidy $\rho_s = \frac{1}{m} \sum_{p \in \{1, \dots, m\}} f_{p,s}$ is the tumour sample ploidy (i.e., the average fractional copy number across all cells in the sample), and where γ_s is the sequencing coverage (sampled from TRACERx cohort distributions). As such, $t_{p,s}$ is drawn from a Poisson distribution with the mean equal to the expected total number of reads, i.e. $t_{p,s} \sim Poisson(\frac{f_{p,s}}{\rho_s} \gamma_s)$. Second, $v_{p,s}$ is modelled as a Binomial distribution based on previous studies[3,19,20]. We defined the expected value of the observed VAF, $\psi_{p,s}$, i.e., $\frac{v_{p,s}}{t_{p,s}}$, as $\psi_{p,s} = \frac{\sum_{i \in \{1, \dots, \hat{n}\}} u_{i,s} z_{i,p}}{\sum_{i \in \{1, \dots, \hat{n}\}} u_{i,s} (x_{i,p} + y_{i,p})}$, where $x_{i,p}$, $y_{i,p} \in N$ represent the allele-specific copy numbers of the genomic locus p in tumour clone i and $z_{i,p} \in N$ represents the mutation multiplicity, or number of copies of the locus harbouring an SNV. Therefore, $v_{p,s}$ is drawn from a Binomial distribution with number of trials equal to $t_{p,s}$ and with probability of success equal to $\psi_{p,s}$, i.e., $v_{p,s} \sim Binomial(t_{p,s}, \psi_{p,s})$. Additionally, noisy and artifactual mutations are simulated by generating groups of SNVs with $\psi_{p,s}$ computed using randomly chosen values of clone proportions $u_1, \dots, u_{\hat{n}}$ in each simulated tumour sample.

Additional details are reported in the TRACERx study[1].

4.2 Simulated datasets generated to benchmark CONIPHER

For the purposes of this manuscript, three collections of the ground truth simulations described were generated for benchmarking analysis, the first 2 datasets comprised 150 simulated tumours: the first collection (simulated dataset 1) includes simulated sequencing error and mutation loss[1] as described above; the second collection (simulated dataset 2) is generated in the same way, except is error-free and enforces no mutation loss. The final dataset (simulated dataset 3) comprised 36 simulations (9 simulations each with coverage values of 50x, 100x, 200x, 400x), generated with noise and mutation loss as in simulated dataset 1. The vast majority of the benchmarking analysis has been carried out using simulated dataset 1. For simulated dataset 2, we ran the CONIPHER tree building stage on the ground truth clusters, in order to benchmark the performance of the tree reconstruction only. simulated dataset 3 was used to validate the accuracy of CONIPHER using tumours simulated with a lower sequencing coverage or purity.

Assessing the identification of mutation clusters. For each considered method, we assessed the performance of correctly identifying mutation clusters (Figure 2a). We used the standard adjusted rand index (ARI) to evaluate the accuracy of the inferred mutation clusters compared to the simulated ground truth (the minimum value is 0, which is expected in the case of random cluster assignments, and the maximum value is 1). Note that ARI values capture both erroneous mutations that are in the same ground truth cluster but have been separated in different inferred clusters, as well as mutations in different ground truth clusters that have been identified in the same inferred cluster.

Assessing the reconstruction of tumour phylogenetic trees. For each method, we also assessed the performance of reconstructing the correct evolutionary history of all identified somatic mutations (Figure 2b). To evaluate the identification of the correct ancestral relationships between every pair of mutations, we computed the mutation descendant accuracy similar to previous studies[11,21].

4.3 Validating performance of CONIPHER with varying sequencing coverage and purity

We used simulated dataset 3 to assess the performance of CONIPHER with varying sequencing coverage. We compared the tumour-effective sequencing coverage with the mutation clustering ARI using simulated dataset 3, whereby the tumour-effective sequencing coverage was computed as the product of sequencing coverage and tumour purity. Overall we found that the performance of CONIPHER was only affected when considering relatively low sequencing coverage and tumour purity (Supplementary Figure 2a). Specifically, we observe that when the effective sequencing coverage is <50x the clustering is less reliable. We note that since CONIPHER uses the PyClone algorithm in its mutation clustering step, the effect of sequencing coverage and tumour purity will be highly dependent on the probabilistic method of PyClone; further details on appropriate sequencing coverage and purity thresholds required to produce successful mutation clustering can be found in the Pyclone manuscript[6,22]. Also, we observed that the mutation descendant accuracy did not decrease with lower tumour-effective sequencing coverage on the same simulated dataset. We concluded that CONIPHER's performance is robust when analysing datasets with >50x effective coverage.

4. 4 Benchmarking alternative tree ranking using simulations

We used the ground truth simulations to assess whether the tree ranking implemented in CONIPHER allows the identification of the ground truth tree as a high-rank solution. simulated datasets 1 and 2 were used in this analysis, in which N=138 and N=150 simulated tumour cases had multiple potential phylogenetic trees, for simulated datasets 1 and 2, respectively.

We first calculated the mutation descendant accuracy of all potential alternative tree structures (simulated dataset 1) inferred by CONIPHER. We found that the reconstructed alternative trees with the highest mutation descendant accuracy had lower SCE scores compared to less accurate alternative phylogenetic trees (Supplementary Figure 3a).

Then, we ran the CONIPHER tree building stage on a dataset with no mutation loss or injected erroneous mutations (simulated dataset 2). For each tumour case, for each unique parent-child edge inferred across any of the alternative trees, we evaluated the fraction of alternative trees in which this edge was predicted to occur, and took this as the empirical probability of this tree edge. We observed that the tree edges that were present in the ground truth tree had a higher edge probability across alternative trees inferred by CONIPHER, than incorrectly inferred edges (Supplementary Figure 3b).

To determine whether the alternative phylogenetic trees with the best metric scores (i.e. the trees obtaining the lowest SCE and highest edge probability scores) had a higher accuracy than alternative phylogenies, we performed the following test. For each simulated tumour case (simulated dataset 1), we computed the rank of each alternative tree based on its ground truth mutation descendant accuracy, as a percentage of all alternative trees. We then compared the mean tree accuracy ranking of the set of trees that obtained the best metric scores, with the mean tree accuracy ranking of the set of alternative trees using a paired, two-sided Wilcoxon test (Supplementary Figure 3c). We observed that the default tree, the trees with highest edge probability and the trees with lowest SCE overall had a higher tree accuracy ranking than the average of all alternatives.

5. Benchmarking CONIPHER on sequencing data

We compared the mutation clustering output (Supplementary Figure 4a) and reconstructed tumour phylogenetic trees (Supplementary Figure 4b) of CONIPHER with CITUP[23], LICHeE[15] and Pairtree[24] on CRUK0063, a NSCLC dataset from the TRACERx421 cohort comprising five samples from the primary tumour, one metastatic lymph node sample and one sample taken at disease recurrence[2].

From this case, we found that CONIPHER provided the most realistic reconstruction of tumour evolutionary history. CITUP and LICHeE produced the most similar clustering to CONIPHER, however Pairtree obtained a very different mutation cluster set (Supplementary Figure 4a). In particular, Pairtree assigned many clusters to the truncal cluster, which were consistently assigned to be subclonal in CONIPHER, CITUP and LICHeE. Note that, while Pairtree results might be partially due to the default clustering algorithm used by Pairtree and the use of different clustering methods might vary the results[25], the performance of all methods were assessed using default parameters. Accordingly, the Pairtree phylogenetic tree was inferred as a linear phylogeny (Supplementary Figure 4b). LICHeE did not assign any mutation cluster to the trunk of the phylogenetic tree and inferred multiple subclonal clusters to descend from a germline node (Supplementary Figure 4b). CITUP inferred a phylogenetic tree structure most similar to CONIPHER (Supplementary Figure 4b), however CITUP was only able to identify a limited number of mutation clusters, in contrast to what was suggested from the mutation frequencies (Supplementary Figure 4c). For example, the mutations assigned to cluster 6 in CITUP (left side of panel), were separated into clusters 6, 12 and 23 by CONIPHER. This can be explained due to the presence of cluster 12 in sample CRUK0063_SU_T1.R5 (versus absence of clusters 6 and 23), and the different PhyloCCF values of clusters 6 and 23 in sample CRUK0063_SU_T1.R4. Also, some mutations assigned to cluster 1 in CITUP were separated to a distinct cluster 18 by CONIPHER, which was estimated to have mean PhyloCCF=0.44 in the lymph node metastasis sample CRUK0063_SU_FLN1, compared to cluster 1 (mean PhyloCCF=1.03). Two mutations assigned to cluster 1 by CITUP were assigned to cluster 2 by CONIPHER, as these were separated from cluster 1 due to presence in sample CRUK0063_SU_T1.R3.

Supplementary Note

1. Considering patients input to CONIPHER with multiple genomically distinct tumours

The CONIPHER clustering and tree building algorithm assumes that all input mutation data is from one genomically related tumour. If it is suspected that the input data contains mutations from multiple genomically independent tumours from a single patient, then it is recommended to run the processing steps detailed in our companion manuscript[1]. In brief, if there exist ≥ 10 mutations that have a VAF $> 1\%$ in all tumour samples, the samples are deemed genomically related. If < 10 mutations have a VAF $> 1\%$ in all tumour samples, samples are clustered into two groups based on the mutation VAFs, and this procedure is repeated within sample groups to identify the final sets of samples that are genomically related.

2. Exploring mutations removed during the CONIPHER PROCEDURE

The CONIPHER tree building step returns information on removed mutations (see PROCEDURE) that includes mutations removed during CONIPHER tree building as part of copy number driven clusters or tree removed clusters. These data can be used to easily query the phylogenetic tree produced from the set of mutations input to CONIPHER. In our companion TRACERx study[1], we have found it to be rare that driver mutations are removed during this stage. However, in the rare case driver mutations are removed, we advise manual review of VAFs and inspection to determine whether these could conceivably reflect parallel evolution.

3. Additional output data

During the protocol, CONIPHER creates additional files (some specifically for downstream analysis in R) that can be read in by the user, if desired. These are detailed below. Other output not mentioned below includes intermediary files produced that are not required for downstream phylogenetic analysis and can be ignored.

! CRITICAL. The following output files described are non-essential and only supplemental to the main output described in the PROCEDURE section of the manuscript.

Clustering step

- `<CASE_ID>.all.SNV.cpn.xls`. This is a table giving information about the mutation copy number of each mutation of the tumour. Each row is a new mutation. Tumour sample-specific information is found in columns that begin with `<SAMPLE>.*`.
- `<CASE_ID>.SCoutput.DIRTY.tsv`. This is a mutation table in the same format as the files `<CASE_ID>.SCoutput.FULL.tsv` and `<CASE_ID>.SCoutput.CLEAN.tsv` described in PROCEDURE, except restricting to mutations removed, and hence deemed ‘dirty’, during CONIPHER clustering.

Tree building step

- `tree.RDS`. This is an R list object combining all tree building output for this tumour. The most important list elements are described below. These objects can be used for phylogenetic analysis in R.

- `tree$ccf_table_pyclone_clean` is a mutation table with one row per mutation, and columns describing the PhyloCCF of each mutation in each tumour sample. Column

`PycloneCluster` indicates the cluster this mutation was assigned to, after mutation clustering and tree building. If this cluster was merged with another cluster, the final cluster name is indicated in this table.

- `tree$merged_clusters` is a list of pairs of clusters that were merged during tree building.
- `tree$cpn_removed_clusters` and `tree$tree_removed_clusters` are vectors listing the clusters removed during tree building due to subclonal copy number alterations and mutational errors, respectively.
- `tree$graph_pyclone` is itself a list of objects related to the inferred phylogenetic tree structure. Specifically, `tree$graph_pyclone$Corrected_tree` is a matrix of the tree structure of the default tree, and `tree$graph_pyclone$alt_trees` is a list of all potential tree structures, as shown in the worked example in the manuscript, patient CRUK0063[2]. Alternative tree 1 is always the default tree.
- `tree$nested_pyclone` is itself a list describing output from the cluster nesting stages of the tree building step (Figure 1) listing the clusters removed during tree building due to subclonal copy number alterations and mutational errors, respectively.
    - `tree$nested_pyclone$ccf_cluster_table` is a matrix containing the mean PhyloCCF of each cluster (rows) within each tumour sample (columns). The row names indicate the cluster ID.
    - `tree$nested_pyclone$ccf_ci_lower` and `tree$nested_pyclone$ccf_ci_upper` are matrices in the same format as `tree$nested_pyclone$ccf_cluster_table` with values of the upper and lower confidence intervals for the PhyloCCF values of each cluster in each tumour sample.

- `tree$clonality_table_corrected` is a matrix in the same format as `tree$nested_pyclone$ccf_cluster_table` containing one of the following values: `absent`, if the mutation cluster (row) is absent in that tumour sample (column); `subclonal`, if that mutation cluster is significantly different from the truncal cluster in that tumour sample; or `clonal`, if that mutation cluster is not significantly different from the truncal cluster in that tumour sample.
- `tree$graph_pyclone` is itself a list describing the phylogenetic tree structure inferred from the tree building step.
    - `tree$graph_pyclone$Corrected_tree` describes the default tree output; `tree$graph_pyclone$highest_log_edge_probability_tree` is a list of the alternative trees with highest edge probability; `tree$graph_pyclone$lowest_sce` is a list of the alternative trees with the lowest SCE; and `tree$graph_pyclone$alt_trees` is a list of all alternative phylogenetic trees found by the tree building algorithm (Overview of the CONIPHER method and Supplementary Methods 2.4). The first column is the parental node; the second column is the child node. Example CRUK0063 is indicated below.

```
tree$graph_pyclone$alt_trees
[[1]]
      i
 [1,] "2"  "1"
 [2,] "8"  "3"
 [3,] "21" "4"
…

[[2]]
   i
6  "2"  "8"
```

```
14 "8"  "21"
…
```

- `tree$clone_proportion_out` is itself a list describing estimated clone proportions of each subclone in each tumour sample.
  - `tree$clone_proportion_out$clone_proportion_table` is a matrix in the same format as `tree$nested_pyclone$ccf_cluster_table` with dimensions (number of clusters x number of tumour samples). Entries of the matrix are estimated clone proportions of the subclone relating to that mutation cluster in that tumour sample. This is computed from the default tree.
  - `tree$clone_proportion_out$clone_proportions_min_sce_trees` is a named list of clone proportion tables computed from the tree(s) with lowest SCE. The names of the list elements indicate the names of the alternative trees with lowest SCE.
- `tree$subclonal_expansion_score_out` is itself a list describing estimated subclonal expansion score in each tumour sample.
  - `tree$subclone_expansion_score_out$subclonal_exp_score` is a data frame describing the subclonal expansion score for each tumour sample, computed from the default tree.
  - `tree$subclone_expansion_score_out$subclonal_exp_score_min_sce_trees` is a named list of subclonal expansion score dataframes computed from the tree(s) with lowest SCE. The names of the list elements indicate the names of the alternative trees with lowest SCE.

4. Expected run time

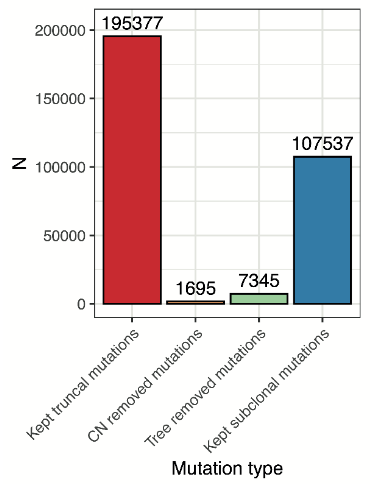
Supplementary Table 1 shows average measures of run time for CONIPHER clustering and tree building run on the simulated dataset 1 (Supplementary Methods 4).

		Run Time					
		Minutes			**Hours**		
		mean	**min**	**max**	**mean**	**min**	**max**
Group	**Method**						
Low	Clustering	45.7	9.2	251.6	0.76	0.15	4.20
	Tree building	0.7	0.2	4.6	0.01	0.004	0.08
Medium	Clustering	80.6	16.0	356.9	1.34	0.27	5.95
	Tree building	2.4	0.4	14.4	0.04	0.007	0.24
High	Clustering	121.9	27.9	336.5	2.03	0.46	5.61
	Tree building	5.7	1.0	29.9	0.10	0.02	0.50

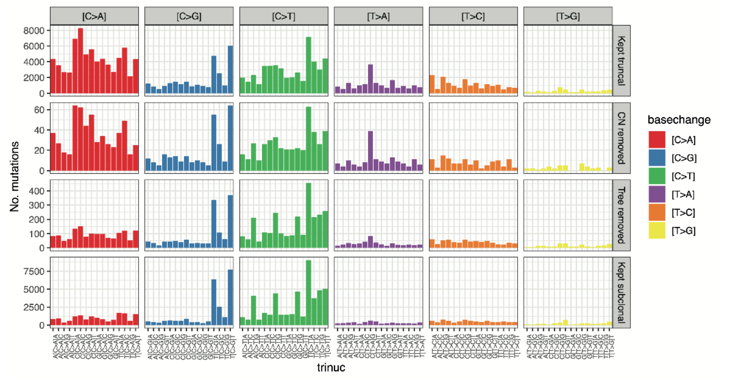
Supplementary Table 1. Mean, minimum and maximum run time of CONIPHER clustering and tree building steps run on the simulated dataset 1 used to benchmark the performance of our method.

Supplementary Figures

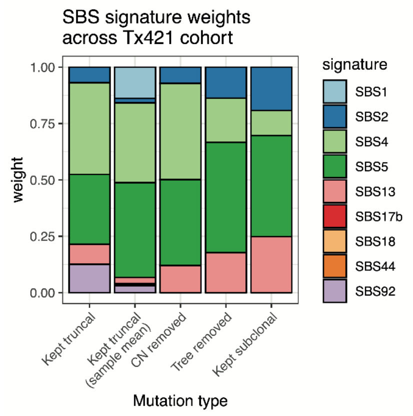
a.



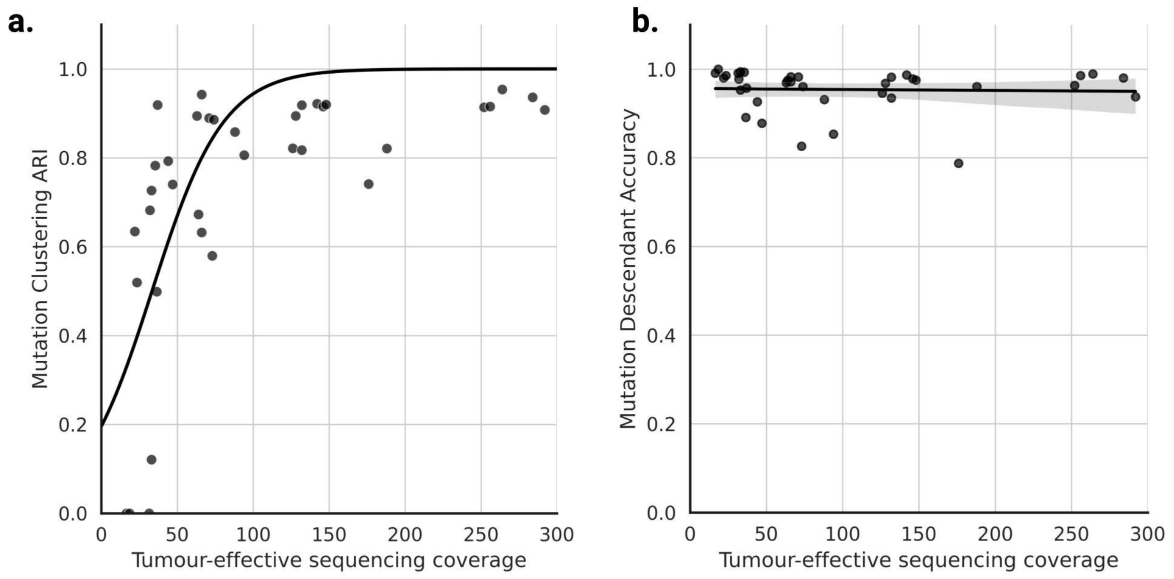
b.



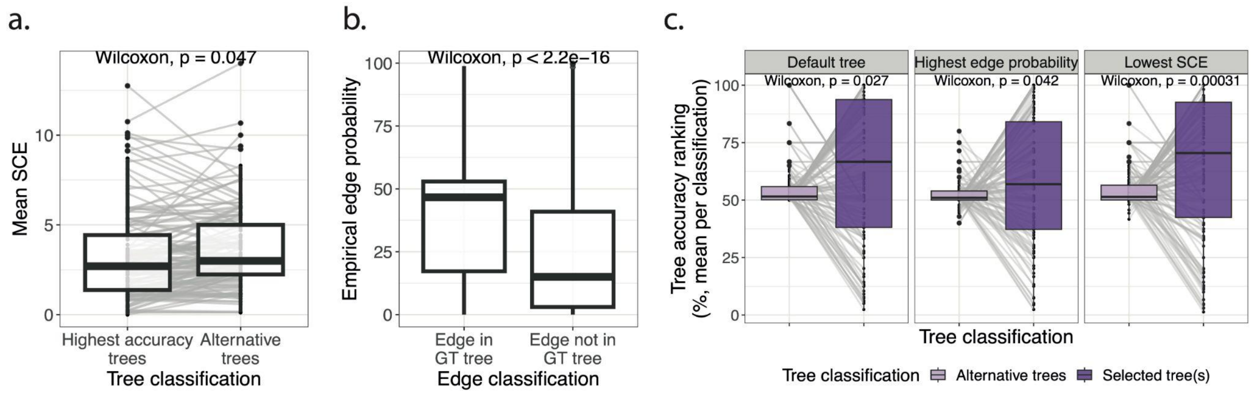
c.



Supplementary Figure 1. **Removed mutations during CONIPHER tree building in the TRACERx421 primary NSCLC cohort. a.** Total number of mutations in the TRACERx421 NSCLC primary cohort that were either kept or excluded from phylogenetic tree building. Mutations are separated into the following mutation groups: kept truncal mutations, mutations removed in copy number-driven clusters, mutations in clusters removed from the tree, and kept subclonal mutations. **b.** Single-base substitution (SBS) mutational profiles, as reported using COSMIC (v.3.2)[26] of the aforementioned groups. Columns show the trinucleotide contexts, rows show the mutation groups. **c.** Signature weights for each mutational signature, as output from deconstructSigs[16], for each of the mutation groups in **a.** and the mean signature weights of truncal mutations across all tumours in the TRACERx421 cohort.

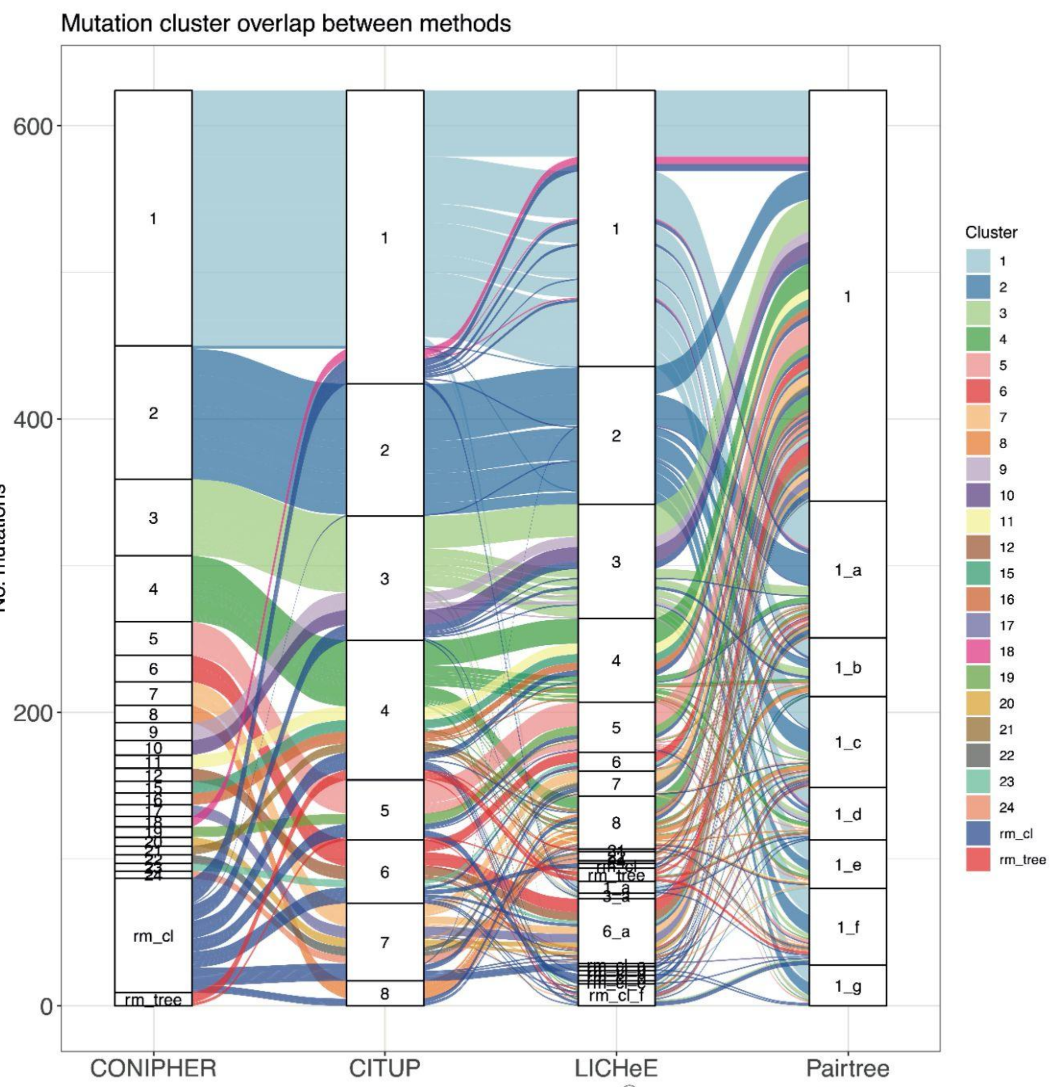


Supplementary Figure 2. CONIPHER performance with varying sequencing coverage. The performance of CONIPHER was measured with varying tumour purity and with varying sequencing coverage (simulated dataset 3). The tumour-effective sequencing coverage is computed as the product of the sequencing coverage and tumour purity. The tumour-effective sequencing coverage is compared with **a.** Mutation Clustering ARI, which measures the accuracy in the identification of mutation clusters for CONIPHER, and **b.** Mutation Descendant Accuracy, which measures the accuracy of CONIPHER in identifying the correct ancestor-descendant relationship between pairs of mutations.

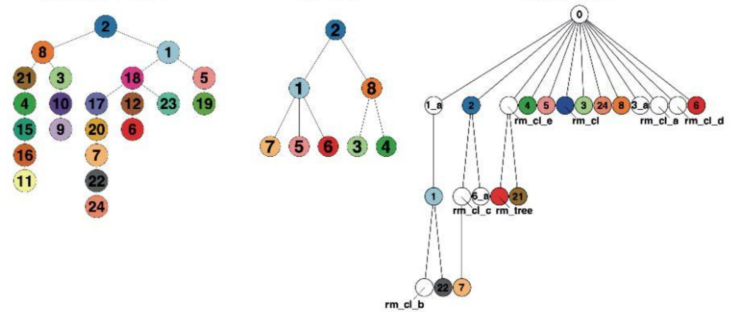


Supplementary Figure 3. **Validating the CONIPHER method of ranking alternative phylogenies. a.** Comparing the mean SCE between trees with highest mutation descendant accuracy with the mean SCE for alternative trees (simulated dataset 1). **b.** Comparing empirical edge probabilities between CONIPHER-inferred tree edges that were present in the ground truth tree with edges not present in the ground truth tree (simulated dataset 2). **c.** Comparing mean tree accuracy ranking between trees selected with metrics: default tree, tree with lowest sum condition error and tree with highest edge probability; and alternative trees (simulated dataset 1).

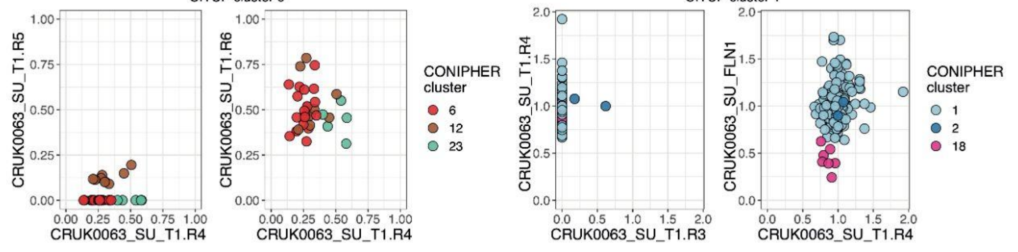
a.


Mutation cluster overlap between methods
No. mutations
0
200
400
600
CONIPHER
CITUP
LICHeE
Pairtree
rm_cl
rm_tree
1_a
1_b
1_c
1_d
1_e
1_f
1_g
6_a
Cluster
1
2
3
4
5
6
7
8
9
10
11
12
15
16
17
18
19
20
21
22
23
24
rm_cl
rm_tree

b.


rm_cl_e
rm_cl
rm_cl_a
rm_cl_d
rm_cl_c
rm_tree
rm_cl_b

c.


CITUP cluster 6
CRUK0063_SU_T1.R5
CRUK0063_SU_T1.R4
CRUK0063_SU_T1.R6
CRUK0063_SU_T1.R4
CONIPHER cluster
6
12
23
CITUP cluster 1
CRUK0063_SU_T1.R4
CRUK0063_SU_T1.R3
CRUK0063_SU_FLN1
CRUK0063_SU_T1.R4
CONIPHER cluster
1
2
18

Supplementary Figure 4. Clustering and tree building method comparison on TRACERx case CRUK0063. a. Alluvial plot of inferred mutation clusters and **b.** Inferred phylogenetic trees of TRACERx421 case CRUK0063[1,2] from CONIPHER, CITUP[23], LICHeE[15], and Pairtree[24]. Each mutation cluster is labelled and coloured according to the corresponding CONIPHER cluster with the largest number of mutations overlapping that cluster. The final two clusters are those removed during the CONIPHER clustering step (rm_cl) and tree building step (rm_tree). **c.** Examples of the PhyloCCF of mutations assigned to CITUP cluster 6 (left two panels) and CITUP cluster 1 (right two panels). Each point represents one mutation and is coloured by the CONIPHER cluster assignment for that mutation.

References

1. Frankell, A. M. *et al.* The evolution of lung cancer and impact of subclonal selection in TRACERx. *Nature* **616**, 525–533 (2023).
2. Al Bakir, M. *et al.* The evolution of non-small cell lung cancer metastases in TRACERx. *Nature* **616**, 534–542 (2023).
3. Jamal-Hanjani, M. *et al.* Tracking the Evolution of Non–Small-Cell Lung Cancer. *N. Engl. J. Med.* **376**, 2109–2121 (2017).
4. Koboldt, D. C. *et al.* VarScan 2: somatic mutation and copy number alteration discovery in cancer by exome sequencing. *Genome Res.* **22**, 568–576 (2012).
5. Cibulskis, K. *et al.* Sensitive detection of somatic point mutations in impure and heterogeneous cancer samples. *Nat. Biotechnol.* **31**, 213–219 (2013).
6. Roth, A. *et al.* PyClone: statistical inference of clonal population structure in cancer. *Nat. Methods* **11**, 396–398 (2014).
7. McGranahan, N. *et al.* Clonal status of actionable driver events and the timing of mutational processes in cancer evolution. *Sci. Transl. Med.* **7**, 283ra54 (2015).
8. Dentro, S. C. *et al.* Characterizing genetic intra-tumor heterogeneity across 2,658 human cancer genomes. *Cell* **184**, 2239–2254.e39 (2021).
9. Gerstung, M. *et al.* The evolutionary history of 2,658 cancers. *Nature* **578**, 122–128 (2020).
10. Hothorn, T., Hornik, K., van de Wiel, M. A. & Zeileis, A. Implementing a Class of Permutation Tests: The coin Package. *J. Stat. Softw.* **28**, 1–23 (2008).
11. Satas, G. & Raphael, B. J. Tumor phylogeny inference using tree-constrained importance sampling. *Bioinformatics* **33**, i152–i160 (2017).
12. Deshwar, A. G. *et al.* PhyloWGS: reconstructing subclonal composition and evolution from whole-genome sequencing of tumors. *Genome Biol.* **16**, 35 (2015).
13. Dentro, S. C., Wedge, D. C. & Van Loo, P. Principles of reconstructing the subclonal

architecture of cancers. Cold Spring Harb Perspect Med. 2017; 7: a026625.

14. Dentro, S. C., Wedge, D. C. & Van Loo, P. Principles of Reconstructing the Subclonal Architecture of Cancers. *Cold Spring Harb. Perspect. Med.* **7**, (2017).
15. Popic, V. *et al.* Fast and scalable inference of multi-sample cancer lineages. *Genome Biol.* **16**, 91 (2015).
16. Rosenthal, R., McGranahan, N., Herrero, J., Taylor, B. S. & Swanton, C. DeconstructSigs: delineating mutational processes in single tumors distinguishes DNA repair deficiencies and patterns of carcinoma evolution. *Genome Biol.* **17**, 31 (2016).
17. Watkins, T. B. K. *et al.* Pervasive chromosomal instability and karyotype order in tumour evolution. *Nature* **587**, 126–132 (2020).
18. Satas, G., Zaccaria, S., El-Kebir, M. & Raphael, B. J. DeCiFering the Elusive Cancer Cell Fraction in Tumor Heterogeneity and Evolution. *bioRxiv* 2021.02.27.429196 (2021).
19. Zaccaria, S. & Raphael, B. J. Accurate quantification of copy-number aberrations and whole-genome duplications in multi-sample tumor sequencing data. *Nat. Commun.* **11**, 4301 (2020).
20. El-Kebir, M. *et al.* Complexity and algorithms for copy-number evolution problems. *Algorithms Mol. Biol.* **12**, 13 (2017).
21. El-Kebir, M., Oesper, L., Acheson-Field, H. & Raphael, B. J. Reconstruction of clonal trees and tumor composition from multi-sample sequencing data. *Bioinformatics* **31**, i62–70 (2015).
22. Gillis, S. & Roth, A. PyClone-VI: scalable inference of clonal population structures using whole genome data. *BMC Bioinformatics* **21**, 571 (2020).
23. Malikic, S., McPherson, A. W., Donmez, N. & Sahinalp, C. S. Clonality inference in multiple tumor samples using phylogeny. *Bioinformatics* **31**, 1349–1356 (2015).
24. Wintersinger, J. A. *et al.* Reconstructing Complex Cancer Evolutionary Histories from Multiple Bulk DNA Samples Using Pairtree. *Blood Cancer Discov* **3**, 208–219 (2022).

25. Wintersinger, J., ethanumn & Moravec, J. *morrislab/pairtree: v1.0.0*. (2022). doi:10.5281/zenodo.7007566.

26. Alexandrov, L. B. *et al.* The repertoire of mutational signatures in human cancer. *Nature* **578**, 94–101 (2020).