



Università degli Studi di Milano Bicocca  
Dipartimento di Informatica, Sistemistica e Comunicazione  
PhD Program in Computer Science  
Cycle XXXIV

# Practical algorithms for Computational Phylogenetics

**Tutor:** Prof. Leonardo Mariani

**Supervisor:** Prof. Gianluca Della Vedova

**Co-Supervisor:** Dr. Yuri Pirola

**PhD Thesis:**

Simone Ciccolella

*Registration number 762234*

**Academic Year 2020-2021**

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	Evolution of cancer . . . . .	5
1.2	Cancer progression inference . . . . .	6
1.3	Clustering of genomic sequencing datasets dataset . . . . .	7
1.4	Comparison of phylogenies . . . . .	8
1.5	Pipeline for single-cell sequencing data analysis . . . . .	9
<b>2</b>	<b>Background and preliminaries</b>	<b>11</b>
2.1	Genomic Sequencing . . . . .	11
2.1.1	Bulk Sequencing . . . . .	11
2.1.2	Single Cell Sequencing . . . . .	12
2.2	Phylogeny models . . . . .	13
2.3	Perfect Phylogeny . . . . .	14
2.3.1	Computation of the Perfect Phylogeny . . . . .	14
2.4	The Dollo Parsimony Model . . . . .	16
2.4.1	Computation of the Dollo- $k$ Phylogeny . . . . .	16
2.5	Cancer Phylogenies . . . . .	17
2.6	Optimization techniques . . . . .	18
2.7	Clustering of biological data . . . . .	19
<b>3</b>	<b>Inference of cancer progression allowing loss of mutations</b>	<b>21</b>
3.1	Introduction . . . . .	21
3.2	Methods . . . . .	24
3.2.1	Formulation of the tree reconstruction problem . . . . .	24
3.2.2	Introduction of the Dollo- $k$ model . . . . .	25
3.2.3	Simulated Annealing . . . . .	27
3.2.4	Visualization . . . . .	29
3.3	Results . . . . .	30
3.4	Conclusion . . . . .	41
3.5	Additional formulations for phylogenetic inference . . . . .	47
3.5.1	Combinatorial ILP formulations . . . . .	48
3.5.2	Combinatorial CSP formulations . . . . .	54

<b>4</b>	<b>Clustering of SCS cancer data</b>	<b>59</b>
4.1	Introduction . . . . .	59
4.2	Methods . . . . .	61
4.2.1	Celluloid . . . . .	61
4.2.2	$k$ -modes . . . . .	64
4.2.3	$k$ -means . . . . .	64
4.2.4	Affinity propagation . . . . .	64
4.2.5	Hierarchical agglomerative clustering . . . . .	65
4.2.6	BIRCH clustering . . . . .	65
4.2.7	Spectral clustering . . . . .	66
4.2.8	Generation of simulated data . . . . .	66
4.3	Results . . . . .	67
4.3.1	Evaluating a clustering . . . . .	67
4.3.2	Assessing the impact of a clustering . . . . .	71
4.3.3	Application on real data . . . . .	77
4.4	Discussion . . . . .	78
<b>5</b>	<b>Comparing tumor phylogenies</b>	<b>80</b>
5.1	Introduction . . . . .	80
5.2	Methods . . . . .	81
5.2.1	Extension to fully labeled trees and multi-labeled trees	82
5.2.2	Extension to poly-occurring labels . . . . .	84
5.2.3	Similarity measure between trees . . . . .	84
5.3	Results . . . . .	87
5.3.1	Simulated Data . . . . .	87
5.3.2	Measures comparison . . . . .	87
5.3.3	Application to clustering of trees . . . . .	90
5.3.4	Application to real dataset . . . . .	92
5.3.5	Application to clustering of patients . . . . .	94
5.4	Discussion . . . . .	94
<b>6</b>	<b>Tumor phylogenetic pipeline</b>	<b>96</b>
6.1	Introduction . . . . .	96
6.2	Methods . . . . .	99
6.2.1	Dimensionality Reduction . . . . .	100
6.3	Results . . . . .	101
6.3.1	Real data . . . . .	101
6.3.2	Simulated data . . . . .	103
6.4	Discussion . . . . .	109
<b>7</b>	<b>Future work</b>	<b>114</b>
7.1	List of publications . . . . .	115

# Acknowledgments

I want to thank my family that had always supported and inspired me in every step and in every decision of my life; many thanks to my *mom*, *dad* and my fantastic sisters *Chiara* and *Francesca*. A very special thank you is needed for my love and partner *Erica*, who is the foundation of my life.

A heartfelt thanks goes to my colleagues *Luca*, *Marco*, *Giulia*, *Murray*, *Stefano* and *Mauricio* who helped, encouraged and inspired during the time we spent together over the years.

Lastly I'd like to thank my Supervisors *Gianluca*, *Paola*, *Yuri* and *Iman* for everything they did and they are continuing to do for my career; for their guidance and for the expertise that they passed to me, without whom none of this research could have been accomplished.

# Chapter 1

## Introduction

The predominant view of evolution of all the species is that all the organisms derive from a common ancestor and that new species arise by splitting one in two or more sub-populations that do not breed with each other. This separation is usually caused by an evolutionary pressure that leads to genomic mutations. These alterations are caused by random chance and are usually driven by external causes such as changes in habitat. Examples of these phenomena are seen in all animal species, where the same animal living in different locations on the globe show different characteristics, such as different colors, fur types and so on. These visible changes of the species (called *phenotypes*) are caused by changes in their DNA (*genotypes*) where, over the course of centuries and generations, some random genetic mutations resulted to be more efficient than others.

Using this definition of evolution, the history of the entirety of life on the Earth can be organized and displayed as a rooted tree, referred to as *the tree of life*, whose latest iteration can be seen in Figure 1.1. All the known extant and extinct species are represented as leaves of the tree, while each internal node represents a point in the history where a set of species diverged.

Computing the tree of life is an extremely computational intensive task that requires very efficient algorithms to be completed. The branches can be caused by an acquisition or a loss of a particular character; depending on how the tree is calculated such a character can be a phenotype character, such as the acquisition of wings or legs – resulting in a construction of a *species tree* – or a genotype character, *e.g.* the rearrangement of genes that causes different pathways of different protein expression – resulting in a *genes tree*. While not part of this manuscript, the reconciliation of species and genes tree is an interesting computational problem [40] that requires efficient algorithms and techniques since most of its formulations are NP-hard.

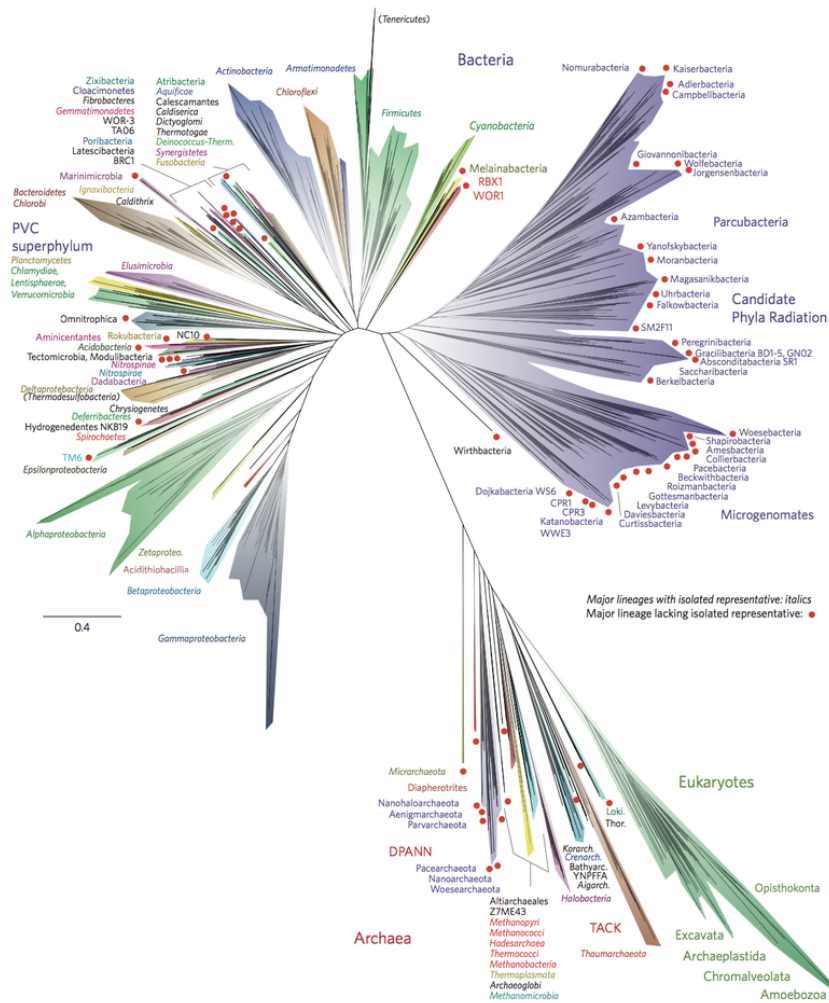


Figure 1.1: The tree of life, picture from [71]. Each branch represents the birth of a new distinct specie and the leaves are the species that are known at the time of computation.

## 1.1 Evolution of cancer

Cancer is a collection of diseases characterized by the abnormal growth of particular cells. The entire process begins with a random mutation in a single cell, called a driver mutation that, due to this alteration, gains an evolutionary advantage over the healthy cells. This advantage allows the cancer cells to live longer and to reproduce more frequently than normal ones [88]. In such conditions, mutated cells grow in number and form a mass – called neoplasm – that invades the body, causing a malfunction of the organism that can lead to the death of the person.

As the tumor evolves from a single mutated cell to a neoplasm, vari-

ous amount of genetic alterations are acquired. Indeed it is very common that a mutation leads to another one, especially if a *oncosuppressor gene* is affected in a way that makes it lose its main function – inhibiting cancer related phenotypes – and thus allowing various genetic modifications to occur. When a cell acquires a new mutation all of its offspring will share the same set of acquired mutations until another one will eventually gain a different alteration. A population of cells that express the same set of mutations is called a clonal expansion of the tumor, thus a cancer is considered as an accumulation of mutations over time.

In recent years, tumors are increasingly being cured with targeted therapies specifically designed for each type of cancer and for each patient — for example, the drug *Lynparza* has recently been approved for treatment of breast and ovarian cancers expressing a specific mutation of the *BRCA* gene. This form of treatment – called *Precision Medicine* – relies on the identification of the mutations present in the patient and most importantly on the evolutionary history of the alterations, making fundamental the discovery and understanding of the order in which the mutations are gained in each patient through different genomic sequencing techniques.

Unfortunately the presence of a tumor is usually detected only when it is large enough to cause symptoms. Hence the sequencing is usually performed at a later stage, when the tumor has mutated several times, resulting in mixture of different clonal population. Furthermore most of the times this sequencing is the only time-point data available, thus it is crucial to develop algorithms that are able to reconstruct the progression of the cancer starting from its final state.

It is commonly assumed [103] that the entire disease starts from one or a small set of – so called – *driver* mutations; therefore such drivers are the most crucial genomic alterations to identify since knowing which they are can be extremely useful for the treatment as well as facilitate an early diagnosis.

However it is not trivial to identify drivers among all the various passenger mutations that follow them. Being able to identify the evolutionary history of the tumor – and mainly the progression driver mutations – is a central problem that needs to be solved for developing targeted therapies and to understand the biology of cancer.

## 1.2 Cancer progression inference

Genomic sequencing technologies produce data relative to the entire genome divided into thousands small sections, called reads, which are then aligned to the reference human genome to reconstruct the entire sequence and later the mutations affecting the genome are called using different techniques; finally the pre-processed data is the input of cancer progression inference tools.

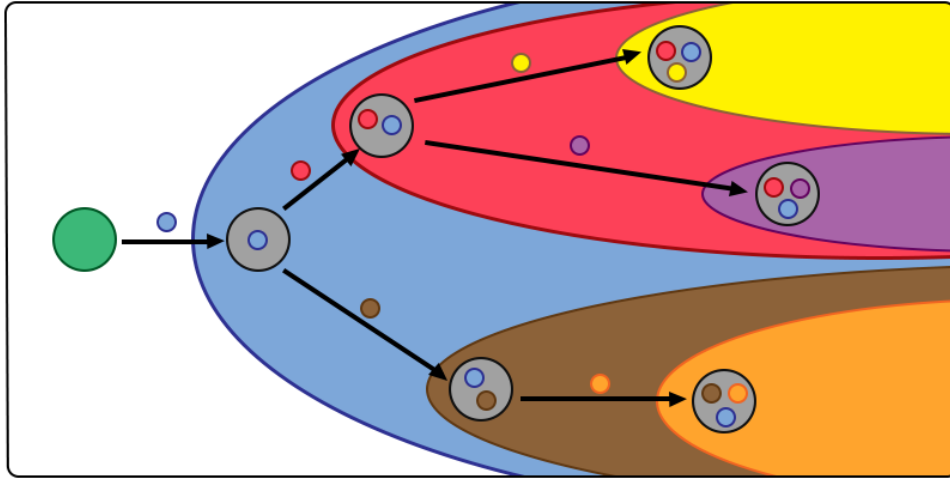


Figure 1.2: Birth and progression of cancer as time passes from left to right: one random mutation occurs in the healthy cell (green cell) and such mutation is carried to its harboring as well as other mutations that occurred later in the process.

The need for efficient algorithms is given by the scale of the input of, especially, *single cell sequencing* (SCS) data. Indeed it is very frequent that the number of mutations scale up to thousands as well as the number of cells that can be as high as various hundreds. One of the biggest SCS dataset studied during the realization of this work had 1842 mutations and 926 cells. Datasets as large as this require methods that are highly efficient and most of the available algorithms fail to run on them, since the solution space scales exponentially on the size of the mutations; therefore basic or naive approaches are not feasible.

Since both the composition of the samples and the evolutionary history are unknown in this model, most of the approaches relies on some simplifying assumption, such as the *Infinite Sites Assumption* (ISA) which postulates that each mutation is acquired exactly once, and never lost, in the entire tree. While this assumption has only limited biological validity [87, 20], it reduces greatly the space of all possible solutions, making feasible several approaches [44, 62], which are at least partially inspired by a classical linear time algorithm [55] for reconstructing the phylogeny from noise-free character data. This latter computational problem is called the *Perfect Phylogeny* (PP) problem.

### 1.3 Clustering of genomic sequencing datasets dataset

Nowadays, we are witnessing a decrease in genomic sequencing costs, coupled with improvements in the quality of the data produced, stimulating the



research on tools for tumor evolution inference data [75, 113, 30, 28, 138, 43]. We believe that this line of research is going to become even more important in the next few years, since currently available data is associated with a very large solution space. Moreover, the high missing value and false negative rates allow a huge number of possible phylogenies with near optimal values of the objective function. This fact makes difficult to determine which methods actually produce better solutions, and shows that the objective function is not able to fully capture the biological soundness of the phylogeny.

These advances in costs and quality of the data produced will result in larger, but more constrained, instances — the net effect being a considerable reduction in the number of likely solutions. Since most of the currently available methods do not scale well to large instances (usually their running time is quadratic with respect to the number of mutations), one solution could be to reduce the size of an instance by clustering: for example, SPhyR [43] uses  $k$ -means [97, 9] to such purpose. However,  $k$ -means is designed for continuous data — where means are usually based on a Euclidean distance — while sequencing data, specifying the presence (1) or absence (0) of a mutation in a cell, or the fact that it is missing (2), can be thought of as categorical.

Clustering categorical data is an active field of research in data mining [83], where massive databases of categorical data are handled, *e.g.*, finding groups of members of a particular insurance policy who also travel overseas on a regular basis. While the goal of clustering here is to allow faster downstream phylogeny inference, it could also be used for error correction in genomic data [102, 98] — a closely related topic.

## 1.4 Comparison of phylogenies

Recent methods to accurately infer the clonal evolution and progression of cancer have made it possible to develop targeted therapies for treating the disease. As discussed in several studies [100, 131], understanding the history of accumulation and the prevalence of somatic mutations during cancer progression is a fundamental step to devise these treatment strategies.

Given the importance of the task, a multitude of methods for cancer phylogeny reconstruction have been developed over the years. The increasing number of tools created has been encouraged by the diversity of data available; for instance, we are witnessing a shift from bulk sequencing data [60, 62, 16, 17, 136] towards single-cell data [75, 28, 138, 43] and hybrid approaches [92, 94].

Having many different tools accomplishing the same task requires solid methods to compare their results. In contrast with classical phylogenetic trees, whose leaves, and only leaves, are labeled (with the species they represent), the trees that model tumor phylogenies are *fully-labeled*, *i.e.*, they also have labels (corresponding to the mutations) on the internal nodes.

A variety of measure to compare classic leaf-labeled phylogenies is available in the literature, while specific methods for tumor are appearing in the latest years [38, 79, 54, 10, 11]. In a detailed analysis of different notions of measures [38] two measures had been introduced to complement the ones used in various tumor inference methods [30, 28]. Such new distances are more specific to capture specific aspects regarding inheritance of mutations, while remaining computationally efficient; however a common characteristic of all the distances available is their reliance on the analysis of *pairs* of nodes.

On the contrary the most widely used measures for classical phylogenies rely on rooted triples [19, 39, 3] (for rooted phylogenies) or quartets [41] of leaves-only labeled trees. While these metrics have limitations for the goal of cancer phylogenetic analysis — they do not apply directly to fully-labeled trees — they also show many desirable properties that are missing in this field. In detail they can efficiently capture the differences in the topologies; our goal is thus to extend such triplet-based measures to provide more insights into the different evolutionary progression, if applied to tumoral phylogenies.

## 1.5 Pipeline for single-cell sequencing data analysis

The field of cancer progression inference is crystallized into different steps that are combined into a complete method that starting from single-cell sequencing datasets computes one or more phylogenies. However the combination of such steps is usually ad-hoc and needs to be implemented manually for each method - or combination of methods - making it time-consuming and error-prone to develop a pipeline.

With the goal of rendering the cancer analysis more streamlined, we developed `plastic` (PipeLine Amalgamating Single-cell Tree Inference Components), an all-in-one and easy-to-use package that includes clustering, inference and comparison of single-cell sequencing tumor data. The method is developed with portability and reproducibility in mind, such that it can be used to create custom scripts or used directly into interactive notebooks.

The module currently incorporates the publicly available methods described in this manuscript *celluloid* [29], `SASC` [30], and `MP3` [26] — respectively for the clustering, inference, and distance — but the interface is publicly available and open-source, making it easily extendable to incorporate any other tool; for this reason `plastic` provides common data structures shared across multiple tools and the file-system.

Furthermore we devised and implemented machine-learning techniques focused on the reduction of the number of cells. In particular we utilized ridge regression [65, 96] and autoencoder [133], that have been proven to

be successful in similar tasks. All the submodules available in `plastic` can either be used in conjunction or independently to create complex interaction, thanks to the specific data structures devised for the scope.

## Chapter 2

# Background and preliminaries

### 2.1 Genomic Sequencing

Genomic sequencing (either DNA or RNA) is the main step that leads to the investigation of the evolutionary history of cancer from a computational point of view. Sequencing is a process that produces short fragments of the original genomic sequence, called *reads*. From the reads, different types of data can be produced that require extremely efficient and accurate methods to reconstruct tumor cell evolution.

The first process is usually the alignment of reads to the reference genome - that in case of humans has a length of over 3 billions bases [32, 31] - using powerful and highly computational intensive tools [89, 7]. After the reads are aligned to the genome, it is possible to process them using a *mutation caller* algorithm that detects mutations in the sequenced DNA, i.e. bases or entire DNA sub-sequences that differ from their healthy counterpart. After identifying the set of relevant mutations the data can be processed to obtain inputs as described in the following sections.

Nowadays there are two different sequencing methods corresponding to two main different kinds of data that are available and that are the input given to algorithms. Figure 2.1 shows the different type of data, while in the following we will show how data is produced and the advantages and disadvantages of each method.

#### 2.1.1 Bulk Sequencing

In a Bulk Sequencing process a sample is extracted from the tumor, either a tissue in the case of solid tumors or a blood sample in the case of Leukemia, and the entire biomass is sequenced. This method is relatively cheap and produces reliable data. Each sample contains several cells, tumoral and

Gene	Mutant	Reference
BBS4	393	1607
CAMSAP1	337	1663
DOCK3	382	1618
EPHA10	412	1588
EYA4	201	1799
HIPK4	654	1346
HIST1H2AG	380	1620

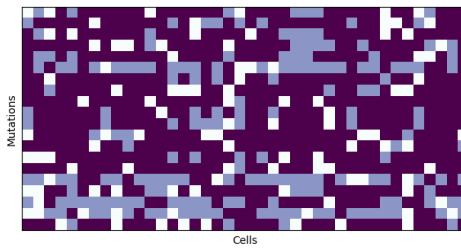


Figure 2.1: Example of Bulk Sequencing (left) and Single Cell Sequencing (right) data. Bulk data produces Variant Allele Frequency (VAF), calculated as  $VAF = \text{Mutant} / (\text{Mutant} + \text{Reference})$ . The heatmap shows the SCS data, where columns represent cells and rows are the mutation called; furthermore entries in purple indicate the absence of a mutation in a given cell, light-blue entries indicate the presence of a mutation in a cell, finally the white color represents a missing entry.

healthy, hence this kind of data produces an estimate of the fraction of cells in the sample with a given mutation. Usually, the data is then represented as a matrix  $VAF(s, m)$  that corresponds to the fraction of cells of the sample  $s$  that express mutation  $m$ . It is common that the number of mutations found in the samples rises up to several thousand, especially in solid tumors.

### 2.1.2 Single Cell Sequencing

During a Single Cell Sequencing (SCS) process the sampled is divided into wells where each cell is extracted from the tumor and individually sequenced, thus producing data for each single cell. This method produces much more granular data, but given its novelty is still very expensive, can result in up to 20-40% false negative rate, a 1-5% false positive rate and a 10-30% missing data rate.

In this case, the data are represented as a matrix  $M_{cm}$  with values 0,1,? corresponding respectively to the fact that the cell  $c$  does not have mutation  $m$ , has mutation  $m$ , or there is no information. This technology is becoming cheaper and more accurate, therefore it is more and more common to sequence millions of cells and to detect thousand of mutations. The size of the data in this case requires efficient algorithmic solutions to the computational problems that arise.

Single-cell Sequencing (SCS) technologies promise to deliver the best resolution for understanding the underlying causes of cancer progression. However, it is still difficult and expensive to perform SCS experiments with a high degree of confidence or robustness. The techniques available nowadays are producing datasets which contain a high amount of noise in the form of false negatives from allelic dropout, and missing values due to low coverage.

Although this sequencing technology is rapidly improving, and some issues are slowly fading away, it is important to develop methods that are able to infer cancer progression despite the lack of accuracy in the data produced by current SCS techniques.

## 2.2 Phylogeny models

Here we will present and discuss two types of *character-based* phylogenies, *i.e.* the input of the problem is a set of attributes called *characters* that a set of species may possess. The goal of computational phylogenetic problems is to compute an evolutionary history of the species, based on the expression of the characters, that optimizes a likelihood function (*maximum-likelihood*) or minimize the number of operation necessary to obtain the solution (*parsimony-based*).

**Definition 1.** Let  $M$  be an  $n \times m$  binary matrix representing  $n$  species in terms of  $m$  characters that describe the species. Each character takes on one of two possible states, 0 or 1, and cell  $(s, i)$  of  $M$  has a value of 1 if and only if species  $s$  has character  $i$ .

**Definition 2.** Given an  $n \times m$  binary-character matrix  $M$  for  $n$  species, a *phylogenetic tree* for  $M$  is a rooted tree  $T$  with exactly  $n$  leaves that obeys the following properties:

1. Each of the  $n$  species labels exactly one leaf of  $T$ .
2. Each of the  $m$  characters labels exactly one edge of  $T$ .
3. For any species  $s$ , the characters that label the edges along the unique path from the root to leaf  $s$  specify all characters of  $s$  whose state is one.

The interpretation of a phylogenetic tree  $T$  for  $M$  is that it produces and evolutionary history of the species in terms of branching pattern, based of the following biological assumptions:

1. The root of the tree represents an ancestral object that has none of the  $m$  characters, *i.e.* the state of each characters is zero in the "ancestral root specie".
2. Each of the characters change from the 0 state to the 1 state exactly once. This condition is called the *Infinite Sites Assumption (ISA)*.

While the first condition is generally considered true for any type of phylogenetic model, the second one can be violated. In the following we will describe different models of evolution starting from the one that adheres to ISA.

## 2.3 Perfect Phylogeny

The *Perfect Phylogeny (PP)* is the simplest evolutionary model – based on the ISA – where each character  $c$  mutates exactly once, from 0 to 1. A Perfect Phylogeny can be represented, as discussed previously, using a binary matrix  $\mathbf{M}$  where the columns indicates the character and the rows represent the species. Every entry of the matrix is either a 0 or a 1 where at the position  $M_{ij}$  a 1 represents that the specie  $i$  has the character  $j$  and a 0 indicates that the character is not present.

**Theorem 3.** (Three gametes rule). *If a binary matrix  $M$  contains at least one pair of columns  $p, q$  in which are presents all the pairs  $(0, 1)$ ,  $(1, 0)$ ,  $(1, 1)$  then  $M$  **does not** admit a Perfect Phylogeny.*

Figure 2.2 shows an example of matrix (*left*) admitting a PP, and on the (*right*) the computed evolutionary history  $T$  of the matrix, where all the species  $s_1, \dots, s_4$  are the leaves of  $T$  and the inner nodes are not used. A PP, like all other phylogenies, can be extended such that even the inner nodes can be used to express species.

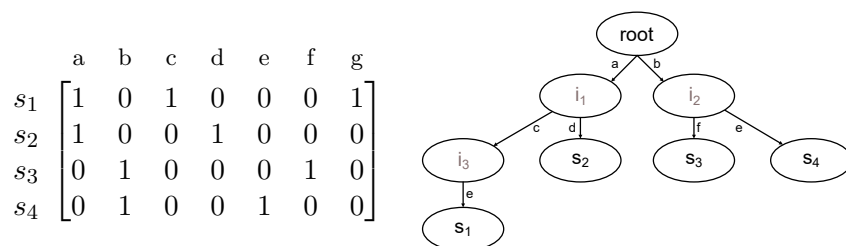


Figure 2.2: (left) Example of binary matrix that adheres to the ISA and (right) the tree representing its perfect phylogeny.

When the three gametes rule is violated, it is not possible to compute a PP, for example in the matrix in Figure 2.3 the columns  $a$  and  $c$  have in the first row  $(1, 1)$ , in the third  $(1, 0)$  and in the fourth  $(0, 1)$  thus it does not adhere to the ISA. These three pairs are also called the *forbidden gametes*.

### 2.3.1 Computation of the Perfect Phylogeny

There exists a polynomial-time algorithm [56, 57] for computing a binary perfect phylogeny, if it exists that consist of:

1. Consider each column of  $M$  as a binary number. Using radix sort, sort these numbers in decreasing order, placing the largest number in column 1. Call the new matrix  $\overline{M}$  and name each character by its column position in  $\overline{M}$ .

$$\begin{array}{cccc}
& a & b & c & d \\
s_1 & \left[ \begin{array}{cccc} 1 & 0 & 1 & 0 \end{array} \right. \\
s_2 & \left[ \begin{array}{cccc} 0 & 0 & 0 & 1 \end{array} \right. \\
s_3 & \left[ \begin{array}{cccc} 1 & 1 & 0 & 1 \end{array} \right. \\
s_4 & \left[ \begin{array}{cccc} 0 & 0 & 1 & 0 \end{array} \right.
\end{array}$$

Figure 2.3: Example of a matrix not admitting a PP, since columns  $a$  and  $c$  violates the *three games rule*.

2. For each row  $p$  of  $\overline{M}$ , construct the string consisting of the characters, in sorted (increasing) order, that  $p$  posses.
3. Build the keyword tree  $T$  for the  $n$  strings constructed in step 2.
4. Test whether  $T$  is a perfect phylogeny for  $M$ .

However it is common that the input matrix  $M$  contains some unknown entry, complicating the problem; in particular the character-based phylogeny reconstruction problems we study in this manuscript are constrained versions of the general *Incomplete Directed Perfect Phylogeny (IDP)* [105]. The IDP problem asks for completing missing data in a binary matrix, where missing data are represented by the symbol  $?$ , in such a way that the completed matrix is explained by a perfect phylogeny. More precisely, the input data is an  $n \times m$  matrix  $M_?$ , where  $M_?(i, j) \in \{0, 1, ?\}$  represents the absence, presence or uncertainty of a character  $j$  in the species  $i$  respectively. If a solution exists, then it consists of changing each  $?$  into 0 or 1 obtaining a new binary matrix  $M_s$  that has a directed perfect phylogeny.

Interestingly, the IDP problem has an efficient solution given by an  $O(mn \log^2(m + n))$ -time algorithm [105] when the phylogeny is directed, that is the root is known (and is the all 0s vector), otherwise, the problem of deciding whether there exists an unrooted solution of the incomplete input matrix is NP-complete [124]. There exists an ILP formulation for variants of the IDP problem, where the main question is to complete missing data in an input matrix on  $\{0, 1, ?\}$  with the goal of minimizing the conflicting pairs [59]. Since finding a perfect phylogeny is easy, the main difficulty in solving the IDP problem consists of replacing each  $?$  with a 0 or a 1 to minimize the number of conflicting pairs of columns.



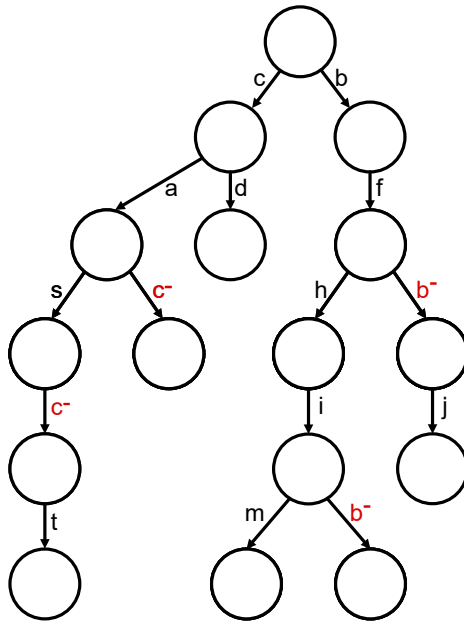


Figure 2.4: Example of a Dollo phylogeny model, where mutations  $b$  and  $c$  are lost twice in the evolutionary history.

## 2.4 The Dollo Parsimony Model

There are some biological phenomena, such as *homoplasy* or *cancer*, that violate the fundamental assumptions of Perfect Phylogeny, thus a more general model is needed. The *Dollo model* allows any character to be lost multiple (infinite) times. Most implementation of the model restrict the number of losses to a predefined upper bound  $k$ , such models are called Dollo- $k$  models. With the restriction of the ISA the optimization problems associated with the Dollo model are NP-Hard.

This phylogeny is used for evolutionary characters that are gained, but that are lost with much higher probability — allowing the 1 to 0 mutation. The Dollo model is appropriate for reconstructing evolution of the gene repertoire of eukaryotic organisms because although multiple, independent losses of a gene in different lineages are common, multiple gains of the same gene are improbable. In Figure 2.4 we can see an example where all characters are gained exactly once, and both characters  $b$  and  $c$  are lost twice.

### 2.4.1 Computation of the Dollo- $k$ Phylogeny

All the different representation of the computation of a Dollo- $k$  phylogeny are NP-Hard, here we will show one possible solution that exploits the properties of the PP [17].

Let  $M$  be a binary (incomplete) matrix with  $n$  rows (species) and  $m$  characters. The extended matrix  $M_{D(k)}$  for the Dollo- $k$  model is defined as follows:

- $M_{D(k)}$  has  $n$  rows and  $m \times (k + 1)$  columns, where each character  $j$  of matrix  $M$  is associated to  $k + 1$  columns in  $M_{D(k)}$  denoted by  $j^+, j_1^-, \dots, j_k^-$ .
- If  $M(i, j) = 1$  then  $M_{D(k)}(i, j^+) = 1$  and  $M_{D(k)}(i, j_l^-) = 0, l \in [1, k]$ .
- If  $M(i, j) = 0$  or  $M(i, j) = ?$  then  $M_{D(k)}(i, j^+) = ?$  and  $M_{D(k)}(i, j_l^-) = ?$  for each  $l \in [0, k]$ .

For a character  $j$ , the column  $j^+$  represents the acquisition of character  $j$  while each of the  $j_1^- \dots j_k^-$  columns represents a possible loss of the gained character. If  $M(i, j) = 1$ , then it is not possible for species  $i$  to lose the character  $j$ , thus the only possible configuration is  $M_{D(k)}(i, j^+) = 1$  and  $M_{D(k)}(i, j_l^-) = 0, l \in [1, k]$ . Otherwise, if  $M(i, j) = 0$  then the character has either (1) never been acquired, or (2) been acquired, then lost along the path from the root to the species  $i$  of any solution. Therefore  $\sum_{1 \leq l \leq k} M_{D(k)}(i, j_l^-) = M_{D(k)}(i, j^+)$ .

Finally, if  $M(i, j) = ?$ , that is the entry of  $M$  is missing, we must allow both the constraints for the case  $M(i, j) = 0$  as well as  $M(i, j) = 1$ .

**Theorem 4.** *Let  $M$  be an incomplete binary matrix, and let  $MIDPP(M_{D(k)}, \mathcal{R}_{D(k)}(M))$  be the corresponding incomplete instance in the extended matrix  $M_{D(k)}$ . Then there exist a completion  $M_c$  of  $M$  satisfying the Dollo- $k$  model if and only if  $MIDPP(M_{D(k)}, \mathcal{R}_{D(k)}(M))$  admits a PP.*

## 2.5 Cancer Phylogenies

*Cancer phylogenies* are derived from the previous *classic* phylogenies and are conceptually similar, albeit showing few differences. In particular, while traditional phylogenies tend to focus on the leaves of the trees as internal nodes are often transitional or extinct species, cancer phylogenies emphasize the internal nodes of the tree, *i.e.* the mutations occurring in the tumor. On the other hand the leaves, that represent the sequenced samples, are often omitted from the representation of the tree. The root is usually called *germline*, *i.e.* it represents all the non-somatic mutations that are specific from the person and are not related to the progression of the cancer.

While most of the previous differences were mostly cosmetic, there are two differences that completely change the landscape of the computational problems: (1) while classical phylogenies tend to be (or can be transformed into) binary trees, this is not true for cancer progression in which the trees are not following any specific structure; (2) cancer tree nodes can have more

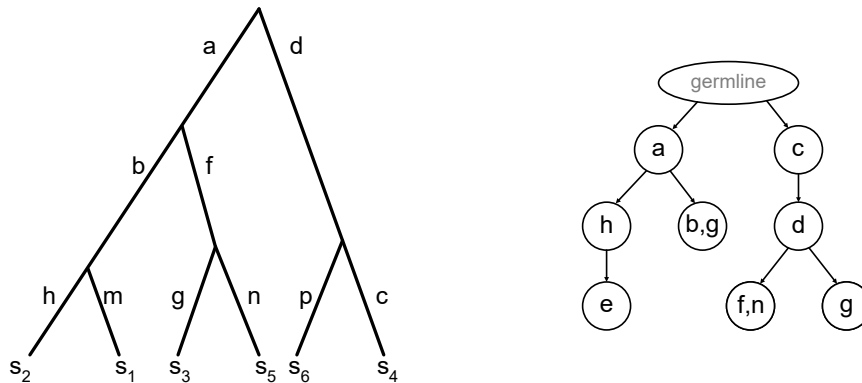


Figure 2.5: (*left*) Representation of a classical phylogeny model; a binary tree with leaves (species) as only labels nodes. (*right*) More traditional representation of a cancer clonal tree, where the nodes are labeled - possibly by multiple mutations appearing multiple times - and the species (samples) nodes are not portrayed.

than one label associated and this fact renders almost all definition defined for classical trees inapplicable to tumor progression.

Therefore it is easy to convert a classical phylogeny to a cancer one, but the opposite can be impossible if (1) or (2) are present in the tumor progression tree. An example of the differences can be seen in Figure 2.5.

## 2.6 Optimization techniques

Classical views of optimization in phylogenetics revolve around the use of a *Parsimony*-based approach, where – following the idea of the *Occam's razor* – the best solution is the simplest. This philosophical reasoning is translated into a computational one where given a set of operations the most optimal solution will be the one that utilizes the less amount of them. For example in the case of the PP or the Dollo- $k$  model, described in the previous sections, the idea is to change the minimum amount of entries in the input matrix and - for the Dollo model - the minimum amount of losses introduced. While the optimization method is based on a basic concept, it tends to perform very well, especially in bioinformatics problems since the same principle apply to the functioning of biology and to its evolution.

On the other hand in recent years more statistically-oriented approaches has been developed - called *maximum-likelihood* - where a given probability function is being optimized, usually based on Bayesian reasoning. The advantage of such method is the ability to define more granular functions that can capture different aspects of the same problem, as well as assigning different priorities to different aspects.

There exist many different optimization techniques to solve these prob-

lems, both from a *combinatorial* and *heuristic* approach. The main combinatorial methods use either *Integer Linear Programming (ILP)* [28, 94, 17, 18, 43] or *Constraint Satisfaction Problems (CSP)* [94]; while the most used heuristic are *Monte Carlo Markov Chain (MCMC)* [75, 92] or *Hidden Markov Models (HMM)* [139, 138] and *Simulated Annealing (SA)* [30].

ILP and CSP models define a mathematical representation of the problem respectively in form of a system of numerical constraints to satisfy or a system of Boolean (truth) constraints. Both approaches allow the user to define a specific optimization function in the same numerical or Boolean fashion. These methods are very elegant and powerful however they require the use of solvers to be computed. Since there is a lot of competition for the solvers they are usually proprietary and require expensive licenses.

Heuristic algorithms can be defined in many different ways and do not need to adhere to any specific paradigm; most of them are based on probability function that allow the algorithm to move to one solution to another in the search space. Some methods are inspired by nature, such as SA, Particle Swarm Optimization and some from more statistical principles such as MCMC and HMM. The main advantage is that there is a huge variety of algorithms to use and they tend to be fast to compute, but unlike ILP or CSP they cannot assure the optimality of the solution. While in theory this seems to be a huge downside, in practice they tend to find solutions very close to the optimum thus making them extremely useful for real case scenarios.

## 2.7 Clustering of biological data

*Clustering* in general is a very common technique in computer science where it is used to find similarity in the data either to group it or to reduce the dimensions by choosing a representative for each given cluster. Notable examples of clustering methods include *k-means*, *k-modes*, *DBSCAN* and many others that work on numerical values. Other algorithms use properties of graphs - or other data structures - to find communities, *i.e.* highly connected components of nodes, that represent specific clusters of elements.

Given their powerfulness and usefulness clustering algorithms are widely used in computer science, however since biological data is not numerical most of them require special re-formulation to be applied in bioinformatics. The work done in this field is usually to adapt genomic data to work with already existing methods, but recently more interest has been put to devise specific algorithms for sequencing data. In this manuscript we will explore a newly defined method specific for SCS data [29].

Clustering of biological data is extremely useful since the amount of genomic data is rapidly increasing each day and the pure number of sequences available is too big to be computed directly. The main goal in the field

would be to group similar data and select a single representative to use a surrogate for the entire class, as a sort of reduction of the input.

## Chapter 3

# Inference of cancer progression allowing loss of mutations

In recent years, the well-known Infinite Sites Assumption (ISA) has been a fundamental feature of computational methods devised for reconstructing tumor phylogenies and inferring cancer progressions. However, recent studies leveraging Single-Cell Sequencing (SCS) techniques have shown evidence of the widespread recurrence and, especially, loss of mutations in several tumor samples. While there exist established computational methods that infer phylogenies with mutation losses, there remain some advancements to be made. We present **SASC** (Simulated Annealing Single-Cell inference): a new and robust approach based on simulated annealing for the inference of cancer progression from SCS data sets. In particular, we introduce an extension of the model of evolution where mutations are only accumulated, by allowing also a limited amount of mutation loss in the evolutionary history of the tumor: the Dollo- $k$  model. We demonstrate that **SASC** achieves high levels of accuracy when tested on both simulated and real data sets and in comparison with some other available methods. The Simulated Annealing Single-Cell inference (**SASC**) tool is open source and available at <https://github.com/sciccolella/sasc>.

### 3.1 Introduction

Recent developments in targeted therapies for cancer treatment rely on the accurate inference of the clonal evolution and progression of the disease. As discussed in several recent studies [100, 131], understanding the order of accumulation and the prevalence of somatic mutations during cancer progression can help better devise these treatment strategies.

Most of the available techniques for inferring cancer progression rely

on data from next-generation bulk sequencing experiments, where only a proportion of observable mutations from a large amount of cells is obtained, without the distinction of the cells that carry them. In recent years, many computational approaches have been developed for the analysis of bulk sequencing data with the purpose of inferring tumoral subclonal decomposition and reconstructing tumor phylogenies (evolutionary trees) [125, 77, 62, 136, 107, 93, 45, 95, 116, 18, 127]. The main drawback of this technique is that a bulk sequencing sample contains a mixture of both healthy and cancerous cells — and this clonal evolution can only be estimated by the proportions of observable mutations.

Single-Cell Sequencing (SCS) technologies promise to deliver the best resolution for understanding the underlying causes of cancer progression. However, it is still difficult and expensive to perform SCS experiments with a high degree of confidence or robustness. The techniques currently available are producing datasets which contain a sizable amount of noise in the form of false negatives from allelic dropout, and missing values due to low coverage. Another issue that these technologies suffer from is the presence of doublet cell captures. However, such issues are slowly fading away and the state-of-the-art in preprocessing steps for removing such artifacts is quite advanced [37]. Hence, we believe that more immediate issues, such as the lack of accuracy reflected in the high dropout and false negative rates inherent to the technology, call for methods that are able to infer cancer progression from this data produced by current SCS techniques.

Various methods have been recently developed for this purpose [75, 113, 138, 137], some of them introducing a hybrid approach of combining both SCS and VAF (bulk sequencing) data [110, 92, 115]. Most of these methods, however, rely on the Infinite Sites Assumption (ISA), which essentially states that each mutation is acquired at most once in the phylogeny and is never lost. One reason being that such a simplifying assumption leads to a computationally tractable model of evolution, namely, the problem of finding a perfect phylogeny [55]. This model is safe to use in settings such as the evolution of natural populations, and tends to be the norm more than the exception in this setting [81]. Cancer progression, however, is a fairly extreme situation, where the evolution is very fast, under attack from the immune system, and with a high mutation rate. As a result, studies of SCS data are beginning to reveal phenomena that cannot always be explained with a perfect phylogeny [87, 20]. Some papers [87] reveal widespread recurrence and loss of mutations, while large deletions on several branches of a tree can span a shared locus [20], thus a given mutation may be deleted independently multiple times.

In this work we propose a novel and more general model to explain the above phenomena, which is not unnecessarily held back by strict adherence to the ISA. Some recent methods are beginning to appear, which have the same objective in mind, such as TRaIT [110], SiFit [138] and SPhyR [43]:

in detail, TRaIT accounts for violations of the ISA by accommodating for convergent evolution; SiFit accounts for both mutation recurrence and loss without specifying a particular model of evolution; and, on the other hand, SPhyR (independently from our paper) utilizes the same phylogeny model used in this work, thus allowing deletions of mutations.

In our approach, we use the Dollo model [47, 111], one of the models that is more general than the perfect phylogeny model, to allow the loss of point mutations. In particular, while the Dollo model still constrains that a mutation can only be *acquired* at most once, it allows any number of independent losses of the mutation. Once we depart from an ideal, error-free, perfect phylogeny model [55], we lose its convenient computational tractability: in fact allowing errors or missing data results in an NP-hard problem. Adopting the more general Dollo model is only going to exacerbate the problem. However, if we restrict the number of losses of any mutation to 1 or 2 (rather than strictly 0), the resulting solution space is still small enough to explore a sizable portion of it in a reasonable amount of time, in practice. Moreover, from a biological point of view, one would not expect a mutation to be lost more than a few times, since it is not likely that mutations are widely lost [87, 20]. Furthermore, all the currently available methods assume that the false negative rate is the same for all mutations. While this is suitable for samples coming from DNA (*i.e.*, scDNA-seq) data, the false negative rate of the mutations in samples coming from RNA (*i.e.*, scRNA-seq) data can vary a because of differing levels of gene expression. Since our approach is suitable for both types of data, that is, a suitable parameter setting can be found for modeling the progression of cancer from samples coming from either DNA or RNA data, to accommodate the latter, our approach also allows a different false negative rate for each mutation: it is one of the first methods with this feature. In fact, to the best of our knowledge, the only other paper with a similar feature has appeared very recently [135]; in that paper, different false negative and false positive rates are allowed for each mutation and for each cell. At the same time, mutation losses are not allowed. SciΦ [123] also allows different rates, but it is essentially a phylogeny-aware mutation caller, not a tool designed to infer tumor phylogenies.

Here we introduce the Simulated Annealing Single-Cell inference (SASC) tool, a maximum likelihood phylogeny search framework that allows deletion of mutations, by incorporating the Dollo parsimony model [47, 111]. We show that our approach is competitive with the state-of-the-art tools for inferring cancer progression from SCS data, while being the only tool to correctly identify important driver mutations in some real datasets, as verified by the manually curated progression scenarios for these data.



## 3.2 Methods

### 3.2.1 Formulation of the tree reconstruction problem

As mentioned before, cancer progression reconstruction can be modeled as the construction of a character-based incomplete phylogeny on a set of (cancer) cells, where each character represents a mutation.

In this framework we consider the input as an  $n \times m$  ternary matrix  $I_{ij}$ , where an entry  $I_{ij} = 0$  indicates that the sequence of cell  $i$  does not have mutation  $j$ ,  $I_{ij} = 1$  indicates the presence of mutation  $j$  in the sequence of cell  $i$ , and a  $?$  indicates that there is not enough information on the presence/absence of mutation  $j$  in cell  $i$ . This uncertainty about the presence of a mutation in a cell is a consequence of insufficient coverage in the sequencing, a common scenario in Single-Cell Sequencing experiments.

However, the uncertainty of some entries is not the only issue that results from the sequencing process. In fact, entries of the input matrix  $I$  can also contain false positives and false negatives — while the false positive rate is usually very low, the false negative rate can be high and can also vary depending on different factors. In particular, for scRNA-seq data, the varying expression levels of different genes can easily lead to different false negative rates for each mutation, since a highly expressed gene will have significantly higher coverage than an under-expressed gene, resulting in a more accurate SNV call for that particular gene. On the other hand, a gene which is less expressed is likely to have a lower coverage, leading to a less accurate presence/absence estimation in the cells. We assume that these errors occur independently across all the (known) entries of  $I$ . Namely, if  $E_{ij}$  denotes the final  $n \times m$  output matrix, *i.e.*, the binary matrix without errors and noise estimated by the algorithm, then  $\alpha_j$  denotes the false negative rate of mutation  $j$ , and  $\beta$  denotes the false positive rate, similarly to [75, 113, 138, 43]. Hence, for each entry of  $E_{ij}$  the following holds:

$$\begin{aligned} P(I_{ij} = 0 | E_{ij} = 0) &= 1 - \beta & P(I_{ij} = 0 | E_{ij} = 1) &= \alpha_j \\ P(I_{ij} = 1 | E_{ij} = 0) &= \beta & P(I_{ij} = 1 | E_{ij} = 1) &= 1 - \alpha_j. \end{aligned}$$

We aim to find a matrix which maximizes the likelihood of the observed matrix  $I$  [75] under the probabilities of false positives/negative and missing entries. Differently from previous works, our model also accounts for losses of mutations, thus we define the prior probability  $P(L(j)) = \gamma_j$  — independent from the previous ones — of losing mutation  $j$  and the set of variables  $c_j$  for  $j = 1, \dots, m$  that denotes the total number of losses for mutation  $j$  in the evolutionary history. In practice, we expect that a researcher might be able to determine that some mutations  $j$  are very unlikely to be lost, therefore setting  $\gamma_j = 0$ .

However, we are interested in the reconstruction of the evolutionary history of the input cells, thus the resulting matrix  $E$  should contain clones

(groups of cells with the same mutations) that can be explained by an evolutionary process of the mutations. This restriction motivates the introduction of the concept of phylogenetic tree, or simply phylogeny.

A (cancer) phylogeny  $T$  on a set  $C$  of  $m$  mutations and  $n$  cells (affected by these mutations) is defined as a rooted tree whose internal nodes are labeled by the mutations of  $C$ , while the leaves are labeled by the cells (see Fig. 3.1A). Notice that the labeling must satisfy some restrictions depending on the evolutionary model that we consider. For example, in a perfect phylogeny, no two nodes have the same label. This is an alternative, but essentially equivalent, definition of classical character-based phylogeny, where the tree  $T$  is defined on a set of characters and where leaves have no label and represent different species.

The *state* of a node  $x$  is defined as the set of mutations that have been acquired but not lost in the path from the root to  $x$ . The state of each leaf  $l$  of  $T$  is naturally represented by a binary vector of length  $m$ , called genotype profile, that we denote  $D(T, l)$ , where  $D(T, l)_j = 1$  if and only if the leaf  $l$  has the mutation  $j$  and 0 otherwise (see Fig. 3.1B).

We say that the tree  $T$  *encodes* a matrix  $E$  if there exists a mapping  $\sigma$  of the rows (cells) of  $E$  to the leaves of  $T$  such that  $E_i = D(T, \sigma_i)$  for each row  $i$  of  $E$ , where  $\sigma_i$  denotes the image of row  $i$  through the mapping  $\sigma$ . Informally,  $\sigma_i$  is the node in the phylogenetic tree corresponding to the node where the cell  $i$  is attached. Notice that the matrix  $E$  is fully characterized by the pair  $(T, \sigma)$  (see Fig. 3.1C). Thus, our problem can be expressed as finding the tree  $T$  that maximizes the following objective function:

$$\max \sum_j^m \left[ -c_j \log(1 - P(L(j))) + \sum_i^n \log(P(I_{ij} | D(T, \sigma_i)_j)) \right]$$

We point out that the values assigned to the unknown entries of the input matrix do not factor into the objective function, that is  $P(I_{ij} = ? | E_{ij} = 1) = P(I_{ij} = ? | E_{ij} = 0)$ . To simplify the computation of the likelihood, we slightly abuse notation in supposing that  $P(I_{ij} = ? | E_{ij} = 1) = P(I_{ij} = ? | E_{ij} = 0) = 1$ . Furthermore,  $\sigma$  can be computed directly from  $T$ ; for each tree we can obtain the best assignment using an exact mapping; leaving  $T$  as the only variable to optimize.

### 3.2.2 Introduction of the Dollo-k model

The Dollo parsimony rule assumes that, in a phylogeny, any single mutation is uniquely introduced in the evolutionary history, but deletions of the mutation can occur any number of times. A restricted version of the Dollo model can be obtained by bounding the number of deletions for each mutation. We denote as Dollo- $k$  the evolutionary model in which each mutation can be acquired exactly once and can be lost at most  $k$  times. The

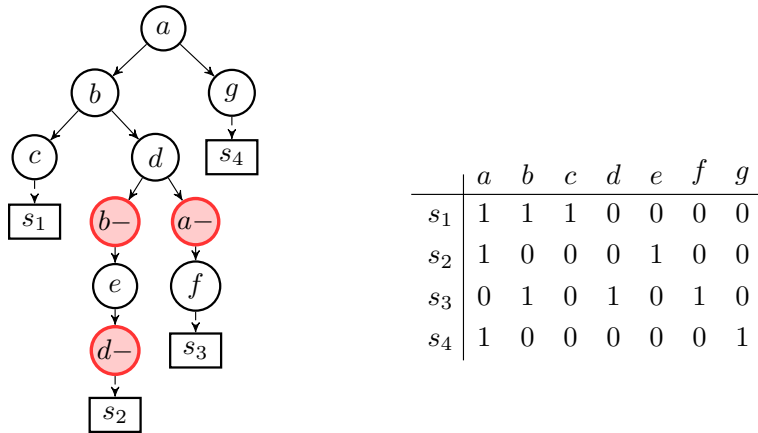


Figure 3.1: Example of a binary matrix  $E$  (right) representing a sample of the ( $n=4$ ) cells  $\{s_1 \dots s_4\}$  affected by the set  $C = \{a \dots g\}$  of mutations. The tree (left) is a cancer phylogeny  $T$  explaining this matrix. Note that the state of the internal node in the tree (left) labeled with (mutation)  $f$  has state  $\{b, d, f\}$  (mutation  $a$  appears in the root, but was lost in the path to this node), hence the genotype profile  $D(T, s_3)$  of leaf  $s_3$  in the tree is 0101010. Note that  $E_{s_i} = D(T, \sigma_{s_i})$  holds for the (trivial) mapping  $\sigma_{s_i} = s_i$ , hence  $T$  (left) encodes  $E$  (right). Informally, leaf  $s_1$  was “attached” to the internal node labeled by  $f$  because genotype profile  $D(T, s_3)$  of leaf  $s_3$  in  $T$  matches the row for  $s_3$  in  $E$ , for example. Observe that the matrix (right) does not allow a perfect phylogeny, and that the tree (left) is a Dollo-1 phylogeny.

special cases, Dollo-0 and Dollo-1, correspond to the perfect [55] and persistent [15, 128, 14] phylogeny models, respectively. The phylogeny reconstruction problem under a Dollo- $k$  model is NP-complete [52] for any  $k > 1$ .

Since the Dollo evolutionary model allows back mutations, we introduce a new type of node label in the phylogenetic tree, to express mutational losses. For each mutation  $p$  we create  $k$  new mutations  $p_l^-$  for  $l \in \{1, \dots, k\}$ , representing the possible losses of mutation  $p$ . As in the perfect case, we require that no two different nodes have the same label. Additionally, we impose that all nodes labeled by a mutation loss  $p^-$  are descendants of the node labeled by the gain of mutation  $p$ . Consequently, the vector  $D(T, \sigma_i)$  which expresses the genotype profile of a row  $i$  will have a 1 in mutations acquired but never lost in the path from the root to the parent  $\sigma_i$  of the leaf  $i$ . Note that the tree of (Fig. 3.1D) is a Dollo-1 phylogeny. We stress that, unlike the case of the perfect phylogeny, when deletions are introduced, we might have more than one tree that is a solution. For example, switching the labels of nodes  $b^-$  and  $d^-$  in Fig. 3.1 produces a different tree which is still a solution of the proposed input matrix when the Dollo model is

considered. Moreover, the set of ancestral relationships between those two mutations is opposite in both representations. An increase of the number of cells and mutations, coupled with the noise caused by false calls and missing entries, expands the solution space of this problem, increasing the number of different cancer progression phylogenies which equally explain the same input.

**Our model.** The model we employ in this work is the Dollo- $k$  model, with the added restriction that there are at most  $d$  total mutation losses in the entire progression. In addition to  $k$ , this  $d$  is a user supplied parameter. Note that, with a maximum  $d$  of total losses in the progression, it means that the variable  $c$  is subject to (1)  $c_j \leq k \forall j$  and (2)  $\sum_j^m c_j \leq d$ . Only a small number of mutation losses in each tumor have been reported [87], therefore we expect small values of  $k$  and  $d$  to be used in practice. Most precisely, we believe that  $k \leq 2$  and  $d \leq 5$  in almost all cases. If the number of mutation is not too small, setting  $d \leq 5$  essentially implies  $k \leq 1$ , hence making the parameter  $k$  mostly irrelevant. Still, we have decided to keep it because it guarantees that some degenerate trees are never computed. We recall that our model also has the  $\gamma_j$  parameters, that is, the prior probability of losing mutation  $j$ .

### 3.2.3 Simulated Annealing

As mentioned before, the fact that (1) we can flip entries and that (2) we want to find the maximum likelihood tree, makes the phylogeny reconstruction problem under the Dollo- $k$  model computationally hard for any  $k > 0$ . For this reason, in this manuscript we consider the Simulated Annealing [82] (SA) approach in order to find a tree which maximizes the likelihood of an incomplete input matrix and that satisfies the Dollo- $k$  phylogeny model, where  $k$  is given as input.

SA is a random search technique which explores the region of feasible solutions, searching for an optimal one. As all other meta-heuristic strategies, it is not guaranteed that SA finds the optimal value of the objective function in a finite number of steps; nevertheless, unlike other deterministic search methods which can be trapped into local optima, SA has been designed to overcome this drawback and converge to a global optimum. The basic idea of the algorithm is to perform a random search which accepts, with some probability, changes that do not necessarily improve the objective function. At each step, the probability of moving to some state with a smaller value changes according to a parameter called the *temperature*, which continuously decreases as the exploration evolves. In the first iterations of the algorithm execution, the temperature is very high, and it is possible (with a fairly high probability) to accept a move into a state with a lower objective value, but as temperature decreases, the probability of moving also decreases. At the

end, when the temperature is sufficiently low, the algorithm becomes a local search method, hence unable to escape a local optimum.

### Neighborhood topology

An essential element of a SA approach that we must provide how the algorithm search process can move from a given state to another. In our particular framework, we attempt to find a tree, thus we must define the neighborhood of a phylogenetic tree in the feasible region, and the algorithm moves from a tree to one of its neighbors. The choice of neighborhood is crucial in the algorithm definition since it determines how feasible solutions are explored, hence ultimately determining whether or not the algorithm converges.

In our approach, the notion of neighborhood is operational, that is, two phylogenetic trees are neighbors if one can be obtained from the other via some operation from a set we will define shortly. For the sake of clarity, we introduce some notation: given a phylogenetic tree  $T$  and a node (labeled as)  $i$ ,  $\rho(i)$  denotes the parent of  $i$  in  $T$ .

- **Subtree Prune and Reattach:** given a tree  $T$  and two internal nodes  $u, v \in T$  such that neither is an ancestor of the other, we prune the subtree rooted in  $u$  by removing the edge  $(u, \rho(u))$  and we reattach it as a new child of  $v$  by adding the edge  $(v, \rho(v))$
- **Add a deletion:** given two nodes  $u, v \in T$  such that  $v$  is an ancestor of  $u$ , we insert a node  $v^-$  that represents a loss of mutation  $v$ . The new node is made the parent of  $u$ . We remark that this operation takes place only if the resulting tree satisfies the desired phylogeny model. More precisely, for the Dollo- $k$  we must check that the mutation  $v$  has been previously lost in the tree at most  $k - 1$  times, and never lost in any ancestor or descendant of  $v^-$ .
- **Remove a deletion:** given a node  $u \in T$ , labeled as a loss, we simply remove it from the tree  $T$ : all children of  $u$  are added as children of  $\rho(u)$  and the node  $u$  is then deleted.
- **Swap node labels:** given two internal nodes  $u, v \in T$ , the labels of  $u$  and  $v$  are swapped. If a previously added loss becomes invalid due to this operation — because a mutation  $c$  is lost in a node  $c^-$ , but the node where the mutation  $c$  is acquired is not an ancestor of  $c^-$  anymore — then we remove the deletion  $c^-$ .

### The algorithm

The goal of the algorithm is to find a maximum likelihood Dollo- $k$  phylogeny tree; a SA process is performed using the previously defined set of

valid operations according to the same temperature decay process — in each iteration, one of these operations is performed, chosen uniformly among all possible candidate operations. Finally, after a new neighbor is generated, cells are optimally attached to the tree, maximizing its likelihood, resulting in the score of the new solution.

Moreover, in the SA search processes, we have that, given a tree and a valid tree operation, the probability of accepting the new solution is  $\min\{e^{\Delta v/T}, 1\}$ , where  $\Delta v$  is the possible change in the likelihood function after performing the operation, and  $T$  is the current temperature. The **cooling process** follows a geometric decay with a factor (cooling rate)  $cr$  of  $10^{-2}$ , *i.e.*, the temperature at the  $i$ -th iteration is equal to  $T_i = (1 - 10^{-2})T_{i-1}$  and  $T_0 = 10^4$ . The SA process stops when the temperature drops below a lower bound set at  $10^{-3}$ .

Since mutation losses are not as frequent as mutation gains, our approach allows to set an upper bound on  $d$ : the total number of deletions of the resulting tree. For example, in a Dollo- $k$  model we can consider only trees where each mutation is lost at most  $k$  times, but there are at most  $d$  nodes associated to mutation losses.

### 3.2.4 Visualization

Alongside the main tool, we produced a post-processing tool, called **SASC-viz**, that can be used to perform processing and filtering operations after the computation of the main tool. Notice that the following operations do not change the actual evolutionary history computed, but only how it is displayed.

- **Collapsing simple paths:** when this option is activated, all simple non-branching paths are collapsed, *i.e.* if a node has only one child, then such node is merged with its child;
- **Collapsing low-support paths:** when this option is activated, if the support of a node  $i$  is lower than a specified value, then  $i$  is merged with its parent  $\rho(i)$ , where the support  $s_i$  of a mutation  $i$  is computed on the output tree as follows: let  $st(i)$  be the set of nodes in the subtree rooted at  $i$ , and let  $C(i)$  be the number of cells assigned to the node  $i$ . Then the mutation support  $s_i$  is:

$$s_i = \frac{\sum_{u \in st(i)} C(u)}{\sum_{v \in st(\pi(i))} C(v) - C(\rho(i))}.$$

We used **SASC-viz** to produce the pictures of the real dataset analyzed. In particular, Fig. 3.12 and 3.13 are obtained by activating the collapsing simple paths option. Fig. 3.11 was produced with more enabled options: by collapsing simple paths and collapsing low-support paths with threshold 5%.

### 3.3 Results

#### Results on simulated data

We have test our method on simulated data, where the ground truth phylogeny is known. We recall that it is possible, however, that a completely different tree achieves a better likelihood on the input data than the one obtained via simulation. This problem is essentially unavoidable, since generating a progression that is the unique solution for the corresponding SCS input matrix would require the contrived addition of artifacts to both the desired tree and the input matrix. These artifacts would likely be so artificial that the resulting instance would not satisfy even the basic assumptions on cancer progression.

#### Generating simulated datasets

To test the methods, we run three different experiments with increasingly sophisticated models, according to the parameter settings of Table 3.1. In the first experiment, we explore a model with the possibility of mutation losses, a phenomenon which has been evidenced by [87]; the second experiment is produced with a model using only different (mutation-specific) false negative rates, as seen in the real data we use. For the third experiment, we combine the previous two to extend the simulation to the most general model in which mutations have the possibility to be lost, and each have a specific false negative rate.

For each of the three experiments, we produced dataset consisting of 50 randomly-generated clonal trees, according to the following procedure for (randomly) generating a tree. Given the number  $S$  of subclones, we generate a random tree on  $S$  nodes by adding a new node as a child of a random pre-existing one. Each of the  $M$  mutations  $q_1, \dots, q_M$  is then, uniformly at random, assigned to one of the  $s_i$  subclones. We allow at most a fixed number  $d$  of deletions in each clonal tree, *i.e.*, according to the prior probability  $\gamma_j = P(L(j))$  of losing mutation  $q_j$ , at most  $d$  mutations are randomly selected to be lost. For each such loss, a new node  $q^-$  is created, and then inserted in the tree at a random valid position, *i.e.*, in the subtree where mutation  $q$  is gained but not in a path where it has already been lost.

To obtain the genotype profile of the  $N$  cells, we uniformly assign, at random, each cell to a node and derive its profile from the clonal tree. Finally to simulate noise in the data, we change a 0 entry to 1 with probability  $\beta$  to simulate false positives and a 1 entry to 0 with the corresponding probability  $\alpha_j$  to simulate false negatives. Moreover, each entry has a probability  $\mu$  to be a missing entry. All flips and missing values are independently distributed, without repetitions.

All the tools were run with the correct false positive rates, where the values of false negatives rates are heterogeneous, the average value of the  $\alpha_j$

is given as input to the other tools, since **SASC** is the only method allowing for different false negative rates.

The false negative error rate distribution of the real datasets are obtained by analyzing the raw data of MGH36 and MGH64 from [126] and comparing the mutation frequencies in the scRNA-seq data to the matching Whole Exome Sequencing (WES) from bulk RNA sequencing, to deduce the drop out frequency. We have analyzed more than 2000 mutations and observed that the distribution of the real data are consistent with a Beta distribution, which we have used as a base for the generation of our simulated data

The values of different false negative rates are randomly chosen from a *Beta distribution*  $\mathcal{B}(\alpha, \beta)$  with parameters  $\alpha, \beta < 1$ , to better simulate the values found in real datasets; the different values of the prior probabilities of mutation losses are produced using a *Triangular distribution* with lower limit  $a$ , upper limit  $b$ , and mean  $c$ , indicated as  $\mathcal{T}(a, c, b)$ . Such distribution is usually used when only the mode, upper and lower bounds are known in a population as proxy for a fair estimation of real-case scenarios.

#	$S$	$M$	$N$	$d$	$\alpha$	$\beta$	$\gamma$	$\mu$
1	12	50	200	3	$\mathcal{B}(0.1, 0.1)$	$3.4 \times 10^{-5}$	$\mathcal{T}(10^{-4}, 0.2, 1)$	0.15
S1	9	50	250	2	0.15	0.1	$\mathcal{T}(10^{-4}, 0.2, 1)$	0.2
S2	7	80	150	0	$\mathcal{B}(0.2, 0.2)$	$3 \times 10^{-5}$	—	0.1

Table 3.1: **Parameters used to simulate the input matrices.** Here,  $S$  is the number of subclones,  $M$  is the number of mutations,  $N$  is the number of cells,  $d$  is the maximum number of allowed mutation deletions,  $\alpha$  is the false negative rate,  $\beta$  the false positive rate,  $\gamma$  is the prior probability of a mutation loss and  $\mu$  is the missing data rate.

### Evaluating the simulated datasets

For each of the three experiments, we measure the accuracy of **SASC** with two scores based on standard cancer progression measures used in various studies [92, 75], *i.e.* *Ancestor-Descendant* and *Different-Lineage* accuracies; a novel parsimony-based score based on the difference between the number of flips, *i.e.*, changes from 0 to 1 and from 1 to 0, estimated by some tool to correct the input; and the actual number of flips introduced by the simulation process to induce the noise. Lastly, we evaluate the trees using the edit distance measure of [104].

**Ancestor-Descendant accuracy:** This measure considers all pairs of mutations  $(x, y)$  that are in an Ancestor-Descendant relationship in the ground truth tree  $T$ . For each such pair we check whether the ancestor-descendant relationship is conserved in the inferred tree  $I$ , in fact we calculate the number of mutations in an Ancestor-Descendant correctly inferred



(true positives); the number of mutations that are incorrectly inferred to have an Ancestor-Descendant relationship (false positives); the number of mutations correctly inferred to not be Ancestor-Descendant (true negatives); finally the number of mutations incorrectly inferred to not have an Ancestor-Descendant relationship (false negatives). The score is defined by the  $F_1$  score, that is the geometric mean of the precision and recall.

**Different-Lineage accuracy:** Just as the previous measure, we consider all pairs of mutations  $(x, y)$  that are not in an ancestor-descendant relationship, *i.e.* are in different branches of  $T$ . The score is defined, similarly to the previous measure, as the resulting  $F_1$  score.

**Parsimony Score:** This is a parsimony-based measure. We measure the difference between the number of flips, *i.e.*, changes from 0 to 1 and from 1 to 0, estimated by some tool to correct the input, and the actual number of flips introduced by the simulation process to induce the noise. The rationale is that a good solution should be smaller, *i.e.*, closer to the correct amount of changes introduced by the simulation process. Formally, the Parsimony Score is defined as  $|\mathcal{H}(S) - \mathcal{H}(E)|$  where  $\mathcal{H}(S)$  is the total number of flips induced by the simulation, and  $\mathcal{H}(E)$  is the number of flips estimated by the tool. While this measure does not consider the overall accuracy of a solution, it is a good estimation if used in conjunction with the previous ones.

**MLTED [104]:** Similar to the Parsimony score, we measure the *distance* between the tree inferred by some tool, and that of the ground truth — according to the recently presented *multi-labeled tree edit distance* (MLTED), which aims to define a distance tailored to cancer progression trees. Again the idea is that a good solution should be smaller to the ground truth, in terms of this distance. In [104], the MLTED is defined as the minimum number of label deletions, leaf deletions and vertex expansions to convert a pair of trees to the maximal common tree. The authors claim that such measure has been recently presented aiming to define a distance tailored to cancer progression trees, since most of the classic tree edit distances do not adapt well when applied to cancer phylogenies. Here we use the implementation of MLTED available at <https://github.com/khaled-rahman/MLTED> to compute the MLTED results reported below.

Note that none of the above mentioned metrics explicitly measures the ability of tools to correctly infer ISA violations.

Additionally to the aforementioned measures, we provide two accuracy measures for the estimation of false negatives: (a) an accuracy of the estimation of the average false negative rate in the simulations and (b) the value of the average, over the 50 trees, of the Mean Squared Error (MSE) over the set of estimations, for each mutation, of the mutation-specific false negative rates. Note that (b) gives an indication also of the *variance* of the estimation of false negative rates, which is important when these rates are heterogeneous, and far from being normally distributed — something we see

in real data that we use here and that is due to varying gene amplification and expression levels.

### Results of the simulation experiments

We decided to compare **SASC** against SCITE [75], SiFit [138] and SPhyR [43]. While B-SCITE [92] is a clear improvement over SCITE, it combines single-cell data with bulk sequencing data — since we do not manage the latter kind of data, a fair comparison is not feasible. For the same reason, we do not compare against TRaIT [110] and PhISCS [94]. OncoNEM [113] was excluded because it is not able to complete the execution on datasets as large as the ones used in the simulations. Each of the tools is properly run with millions of iterations and multiple restarts

The first experiment consists of the cases where only mutational losses occur, thus representing our simplest model, based on scDNA-seq error model. From Figures 3.2 , 3.3, 3.4 we see that SPhyR outperforms all other tools, while **SASC** and SCITE score almost identically. SiFit on the other hand shows poor results in all accounted measures. This is an expected result, since this are the experimental settings for which SPhyR was designed and, given its ILP nature, it is able to achieve a near-optimal solution in most cases.

In the second experiment we want to focus at the cases where a heterogeneous set of false negatives is present in the data while no deletion is allowed, thus simulating errors from scRNA-seq data without any loss of mutation. From Fig. 3.5 we see a clear improvement of **SASC** over SCITE, while SPhyR shows an excellent accuracy. **SASC** and SPhyR perform very close in the Ancestor-Descendant, Different Lineages and Parsimony score while SCITE and SiFit show lower accuracy values. According to the MLTED distance (Fig. 3.6 **SASC** outperforms all the other methods, with a slight advantage over SPhyR. **SASC** also better infers the false negatives rate in both terms of average estimation and MSE, as seen in Fig. 3.7.

Lastly the third experiment shows the results when the datasets contain both heterogeneous false negatives and deletions based on scRNA-seq error model, thus complementing the other experiments. **SASC** outperforms any other tool in every considered measure (Fig. 3.8 and 3.9) and it also shows the best estimation of the false negative rates in terms of average and MSE (Fig. 3.10). It is particularly interesting to notice the drop in performance of SPhyR when it is forced to employ the Dollo model, since this is the only experiment with mutation losses involved. It is also very clear that **SASC** outperforms all the available methods when it deals with heterogeneous false negative rates and mutation losses. It also interesting to notice that **SASC** shows a much higher accuracy than the other two tools that allow mutational losses — SiFit and SPhyR — when such losses are present in the dataset.

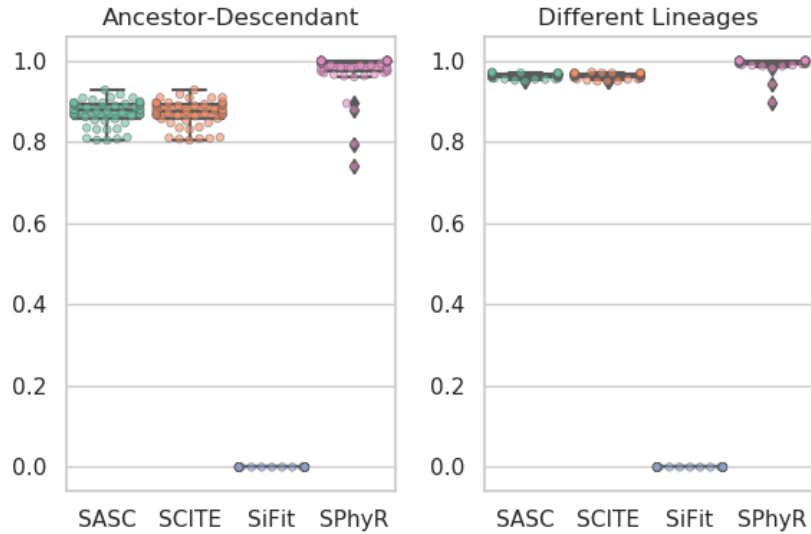


Figure 3.2: Accuracy results for Experiment 1, described in the section “Evaluating the simulated datasets”. SPhyR outperforms all the other tools, since the experiment matches the settings it was designed for, and given its ILP nature, it is able to achieve a near-optimal solution in most cases. SASC and SCITE score almost exactly, both with high accuracy, while SiFit shows lower results.

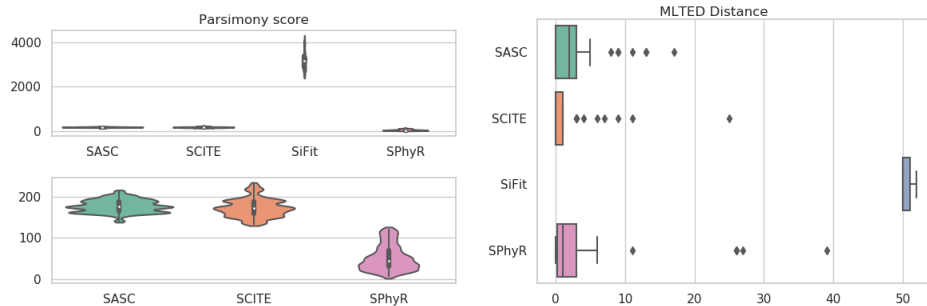
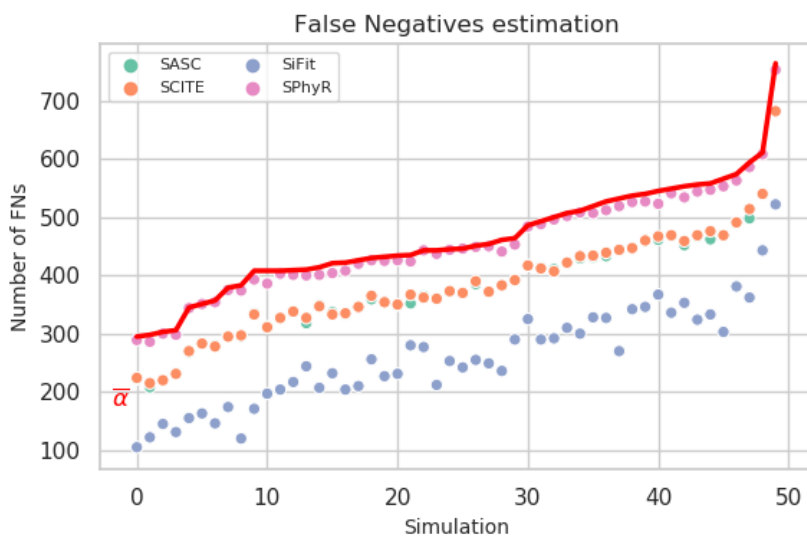


Figure 3.3: Parsimony scores and MLTED results for Experiment 1, described in the section “Evaluating the simulated datasets”. SPhyR outperforms all the other tools, since the experiment matches the settings it was designed for, and given its ILP nature, it is able to achieve a near-optimal solution in most cases. SASC and SCITE score almost exactly, both with high accuracy, while SiFit shows lower results.

### Results on real cancer data

We test and compare SASC on four different datasets, comprising both scDNA-seq and scRNA-seq sequencing data. Since SiFit exhibited poor



Mean Squared Error (average)

	SASC	SCITE	SiFit	SPhyR
MSE	7.62	7.64	5.20	9.07

Figure 3.4: False negative rates estimation for Experiment 1, described in the section “Evaluating the simulated datasets”. SPhyR outperforms all the other tools, since the experiment matches the settings it was designed for, and given its ILP nature, it is able to achieve a near-optimal solution in most cases. SASC and SCITE score almost exactly, both with high accuracy, while SiFit shows lower results.

performances on the simulated datasets, it is excluded in the comparison on real datasets.

### Oligodendroglioma IDH-mutated tumor

We test SASC on an oligodendroglioma IDH-mutated tumor; in particular, on cancer MGH36 [126], consisting of 77 SNVs, distinguished from PCR false positives using matched WES, over 579 cells. Fig. 3.11 shows the tree computed by SASC and the distribution of the false negative rates (shown in the bottom-right corner plot). The distribution stresses the necessity of a method that considers heterogeneous false negative rates, since there are two spikes of rates (at roughly 0.1 and 0.9), *i.e.*, it is highly bimodal, and using the average of the rates would not be an accurate representation. In this particular tumor, no deletion was expected: this is confirmed by the inferred tree.

For the dataset MGH36 from [126], there is no manually curated tree to compare the results of the tools, thus we report the number of false negatives

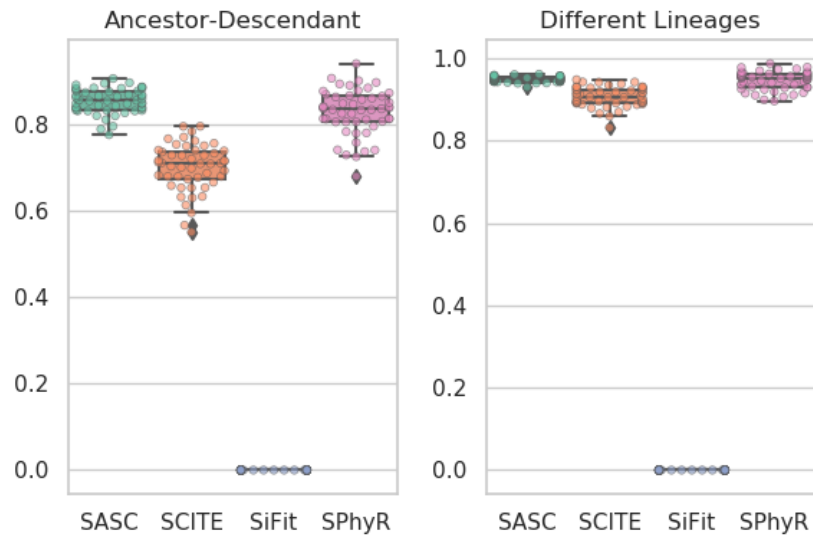


Figure 3.5: Accuracy results for Experiment 2, described in the section “Evaluating the simulated datasets”. **SASC** slightly outperforms **SCITE** in both measures, while **SiFit** is shown to be the poorest scoring method. **SPhyR** scores slightly better than **SASC** on the Ancestor-Descendant accuracy and it outperforms all other tools on the Different Lineages measure. Notice that larger values of both measures are better.

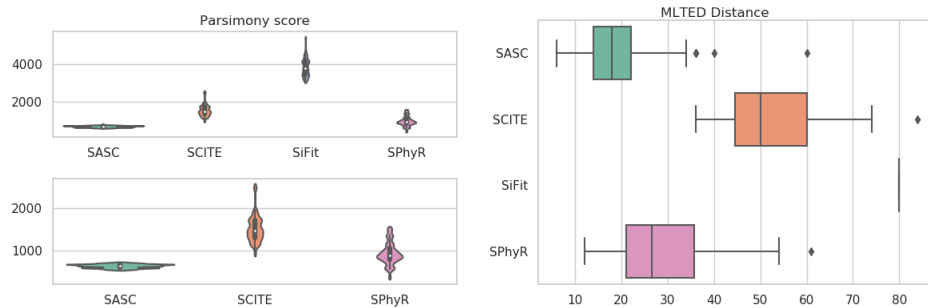
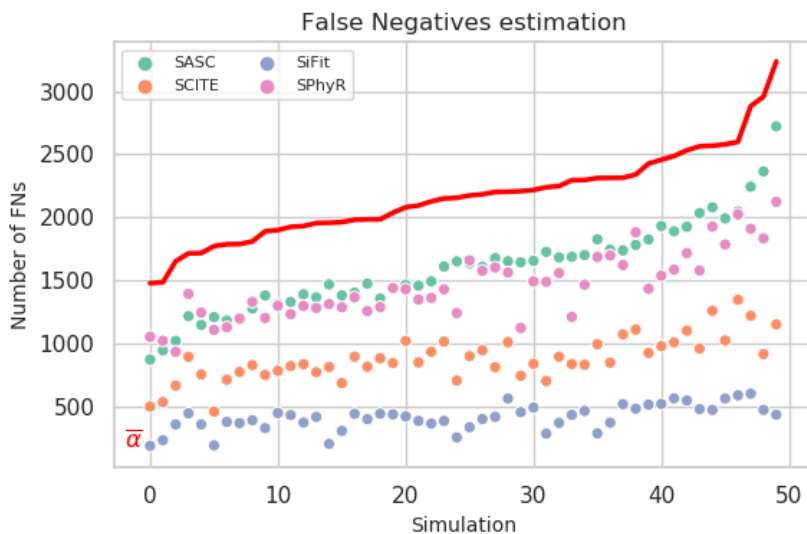


Figure 3.6: Parsimony scores and MLTED results for Experiment 2, described in the section “Evaluating the simulated datasets”. **SASC** obtains better results than **SCITE** in both measure, while **SPhyR**’s performance on the Parsimony score is very similar (albeit with a larger variance) to that of **SASC**. According to MLTED distance **SASC** scores better than any other tool, even though its performance is only slightly better than **SPhyR**. We represent the results of the parsimony score with and without **SiFit**, since its results are much different from the other ones. Notice that smaller values of both measures are better.



Mean Squared Error (average)				
	SASC	SCITE	SiFit	SPhyR
MSE	7.60	7.64	5.20	9.07

Figure 3.7: False negative rates estimation for Experiment 2, described in the section “Evaluating the simulated datasets”. **SASC** shows a more accurate estimation of the false negative rates than the other tools both in terms of average estimation as well as MSE of the single rates for each mutation. The thick red line is the average of the individual false negative rates of the mutations in the ground truth.

and false positives inferred by the methods, this is the number of flips from 0 to 1 and from 1 to 0 respectively from the input to the output. The rationale for this score is to report a parsimony score of the algorithms; a comparison of the likelihood values will not be fair, since **SASC** uses a different formula than the other tools. Such score is shown in Table 3.2; **SASC** introduces the lowest number of false negatives to obtain the solution, albeit being very close to **SCITE**, while **SPhyR** infers the highest number.

### Childhood Acute Lymphoblastic Leukemia

Furthermore, we test **SASC** on Childhood Acute Lymphoblastic Leukemia data from [50]. In particular, we focus on Patient 4 and Patient 5 of this study, given their large amount of both cells and mutations, as well as their complexity. Data on Patient 4 consists of 78 somatic Single Nucleotide Variants (SNVs) over 143 cells, while Patient 5 is affected by 104 somatic SNVs over 96 cells. The original study estimated an allelic drop-out rate of less than 30%. Since the trees in [50], determined using expectation

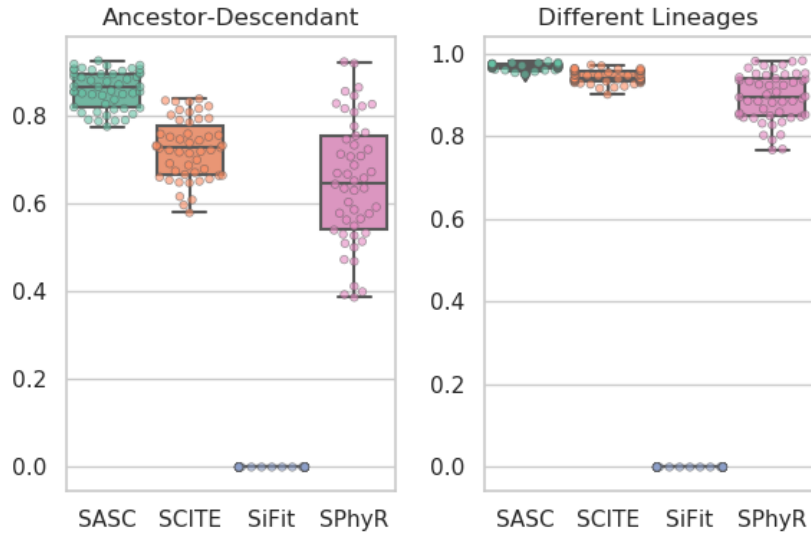


Figure 3.8: Accuracy results for the simulated experiment. In this experiment, **SASC** scores better than any other tool in these measures. Once again **SiFit** is the poorest scoring method. The accuracy of **SPhyR** lowers when mutation losses are included into the dataset and it is forced to employ a Dollo model. To the contrary, **SASC** performs the best when it utilizes the full extent of its capabilities, *i.e.*, the handling of heterogeneous false negative rates and mutation losses. Notice that larger values in both measures are better.

maximization on a multivariate Bernoulli distribution model, are manually curated and of high quality, we select them as the ground truth.

To ensure the absence of doublets, *i.e.* noise produced by error due to the fact that two cells are sequenced instead of a unique cell, we preprocess the input using the *Single-Cell Genotyper* (**SCG**) tool [114]. **SCG** is a statistical model which removes all cells of the datasets that are likely to be doublets.

Fig. 3.12 shows the tree inferred by **SASC** for Patient 4; **SASC** correctly infers the tree structure obtained in the study, as well as the size of the subclonal population. The driver mutations are correctly detected, and mutations **COL5A2**, **SDPR** and **TRHR** are inferred as deletions. Furthermore, boldfaced and colored mutations indicate the correctly placed specific driver mutations for the subclone of the same color. It is interesting to notice that, in the original study, the violet subclone does not have mutations **COL5A2** and **TRHR**: these particular mutations are in fact deleted in the clone. This solution was found assuming a Dollo-1 phylogeny model with no restriction on the total number of deletions in the cancer progression.

In Fig. 3.13, the inferred solution for Patient 5 of the same study is

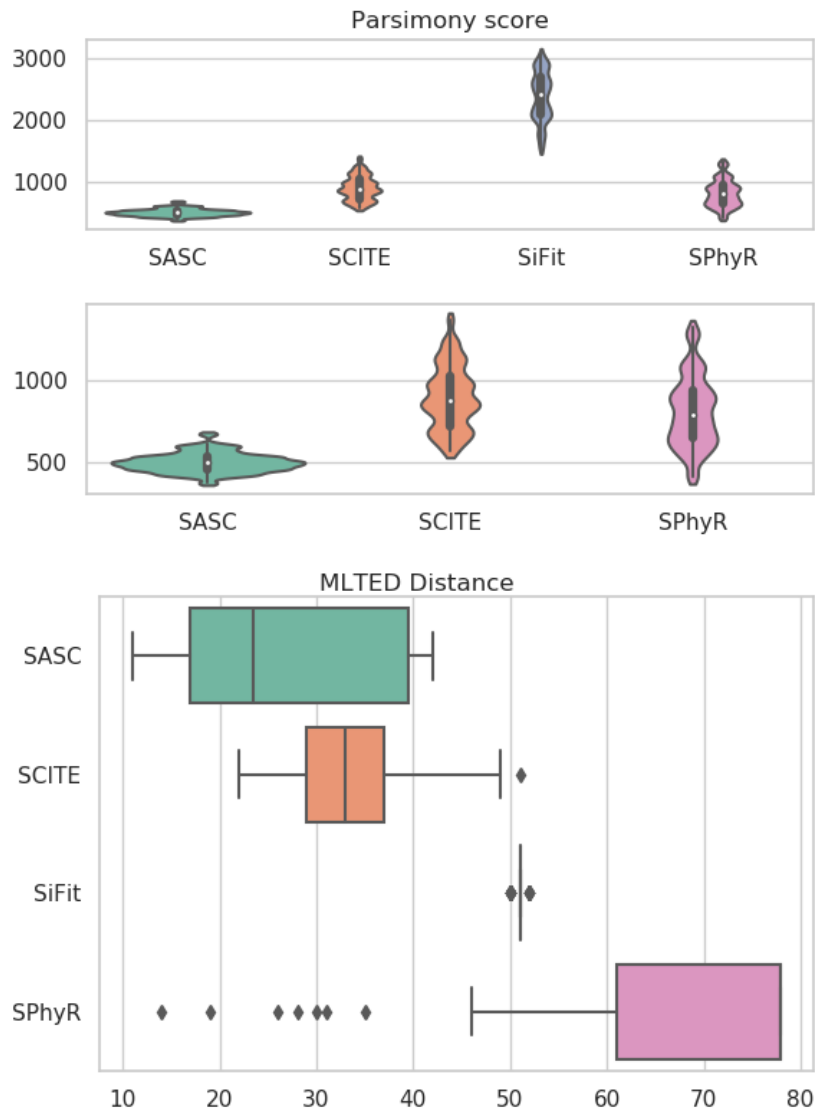
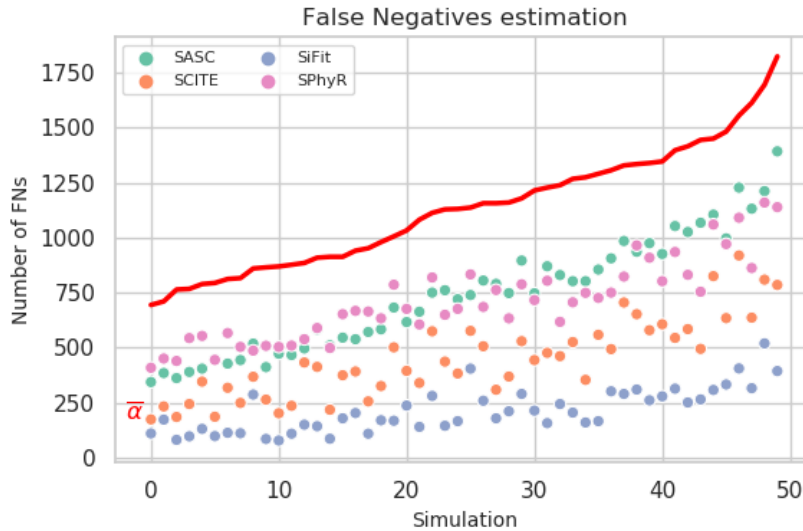


Figure 3.9: Accuracy results for the simulated experiment. According to these two measures, **SASC** scores better than any other tool. A clear performance drop is noticed when **SPhyR** is forced to employ a Dollo model. We represent the results of the parsimony score with and without **SiFit**, since its results are much different from the other ones. Notice that smaller values of both measures are better.

shown. As in the previous dataset, our inferred tree perfectly supports the hypotheses proposed in the original sequencing study: in fact, it correctly infers the topology of the tree, as well as the placement of driver mutations.





Mean Squared Error (average)				
	SASC	SCITE	SiFit	SPhyR
MSE	14.76	8.96	4.26	14.17

Figure 3.10: False negative rates estimation for the simulated experiment. **SASC** estimates the false negative rates better than the other tools, both in terms of average estimation, as well as MSE of the single rates for each mutation. Especially in the latter measure, we can notice a vast discrepancy in the accuracy of the estimation of false negative rates. The thick red line is the average of the individual false negative rates of the mutations in the ground truth.

Boldfaced mutations are the driver mutations for the tree or the subclone with the same color. This solution was found assuming a Dollo-1 phylogeny model with a restriction of 10 deletions in the cancer progression. As described in the section “Simulated Annealing”, such values for  $k$  and  $d$  were empirically found to give the best likelihood.

Since the original study [50], provides manually curated trees we can compare **SASC**, **SCITE** and **SPhyR** to them.

**SCITE** is run using the same setting used for **SASC**, *i.e.* the proposed values of false positive and false negative rates. The tree inferred by **SCITE** for Patient 4 shows a similar structure to the one proposed in the manuscript but it presents more clones. The tree inferred by **SCITE** for Patient 5 shows the correct topology, but a few driver mutations were not correctly detected.

**SPhyR** is run using the same setting used for **SASC**, *i.e.* the proposed values of false positive and false negative rates and assuming a Dollo-1 model. For Patient 4 the tree structure is similar to the one proposed in

the manuscript. The drivers and distinct subclones are also correctly placed. The tree topology inferred by SPhyR for Patient 5 is correctly inferred, however it infers a large number of mutation losses, which is very unlikely and it is probably due to the fact that deletions are used, in this case, to correct false negatives at no cost in terms of likelihood function.

### Medulloblastoma

Lastly we test the methods on Medulloblastoma patient BCH1031 from [66] consisting of 96 mutations over 330 cells. **SASC** and SPhyR computed the solution using a Dollo-2 phylogeny model.

Fig. 3.14 shows the tree inferred by **SASC**, which reported a total of 2 mutation losses. Both trees inferred by **SASC** and SCITE express, as expected, various mutations correlated to the TUBB gene. On the other hand SPhyR inferred a total of 56 mutations over the 96 present in the sample. Furthermore, similarly to the previous experiment, SPhyR inferred a total of 24 mutational losses, which is very unlikely for so many losses to be present in a single sample, since evidence from [87] suggests that this phenomenon is extremely rare. It is more likely that, also in this case, deletions are used to correct false negatives at no cost in terms of likelihood. Lastly, while **SASC** and SCITE each finished its computation in less than 2 hours, SPhyR took more than 24 hours.

	<b>SASC</b>	SCITE	SPhyR
FN	115	121	430
FP	7	6	10

Table 3.2: Number of false positive and false negative introduced on the MGH36 instance.

## 3.4 Conclusion

We have presented **SASC** and we have shown that it is an accurate tool for inferring intra-tumor progression and subclonal composition from both scDNA-seq and scRNA-seq data. **SASC** manages cases with mutation losses and is robust to various sources of noise in all data.

We have tested **SASC** on three simulated datasets, and we have shown that **SASC** is able to outperform all tools when there are mutation losses, while being competitive with SCITE and SPhyR when there are no mutation losses.

We have tested **SASC** on three real datasets. **SASC** has inferred a likely phylogeny tree structure, correctly identifying the driver mutations and the

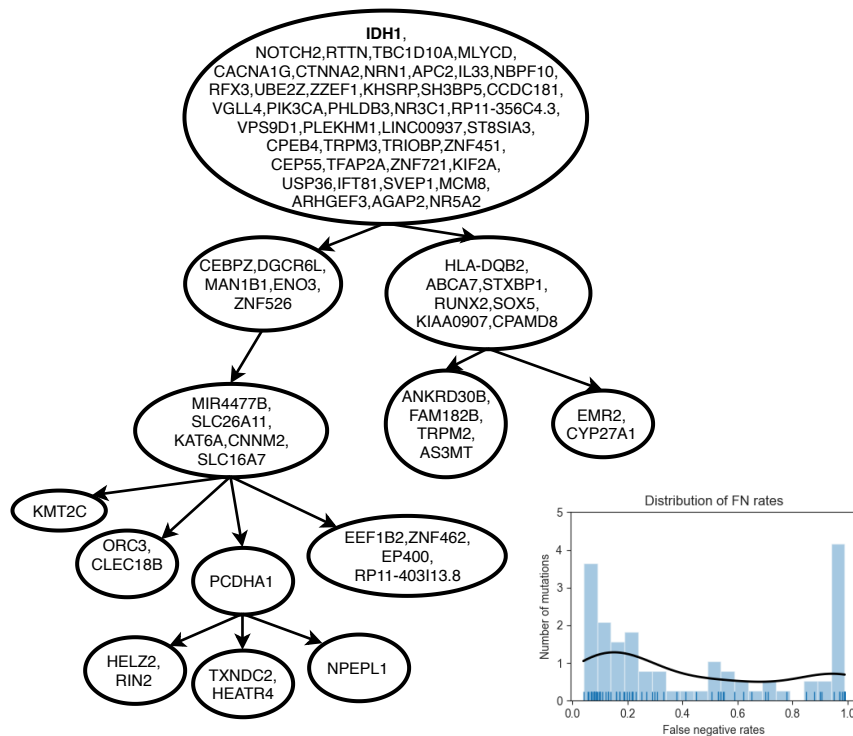


Figure 3.11: Tree inferred by **SASC** for the oligodendroglioma IDH-mutated MGH36 from [126]. The tree was computed using as input different false negative rates for each mutation, whose distribution can be seen in the bottom-right corner plot. The picture was drawn using the **SASC-viz** post-processing tool.

decomposition of the clones. Furthermore, it has solved those large datasets in adequate runtime.

A particularly interesting example is given by the inferred tree in Fig. 3.13. The corresponding input dataset in this case contains more than 5000 conflicts between mutations — each conflict is a pair of mutations witnessing a violation of the Infinite Sites Assumption (ISA). With only a slight relaxation of the ISA — the Dollo-1 model — **SASC** is able to infer an accurate solution with a total of only 8 deletions, while perfect phylogeny methods would require a large number of changes to the entries in the input just to produce a feasible solution.

A future extension could be the inclusion of coverage information from the reads, as in Monovar [139] and SciΦ [123], since it will also have an impact on the false negative rates. Another direction is towards even more general models, for example, allowing each mutation to appear more than once in the tree. Also in this case, special attention must be paid to keeping the model sufficiently restricted so that computation time does not explode,

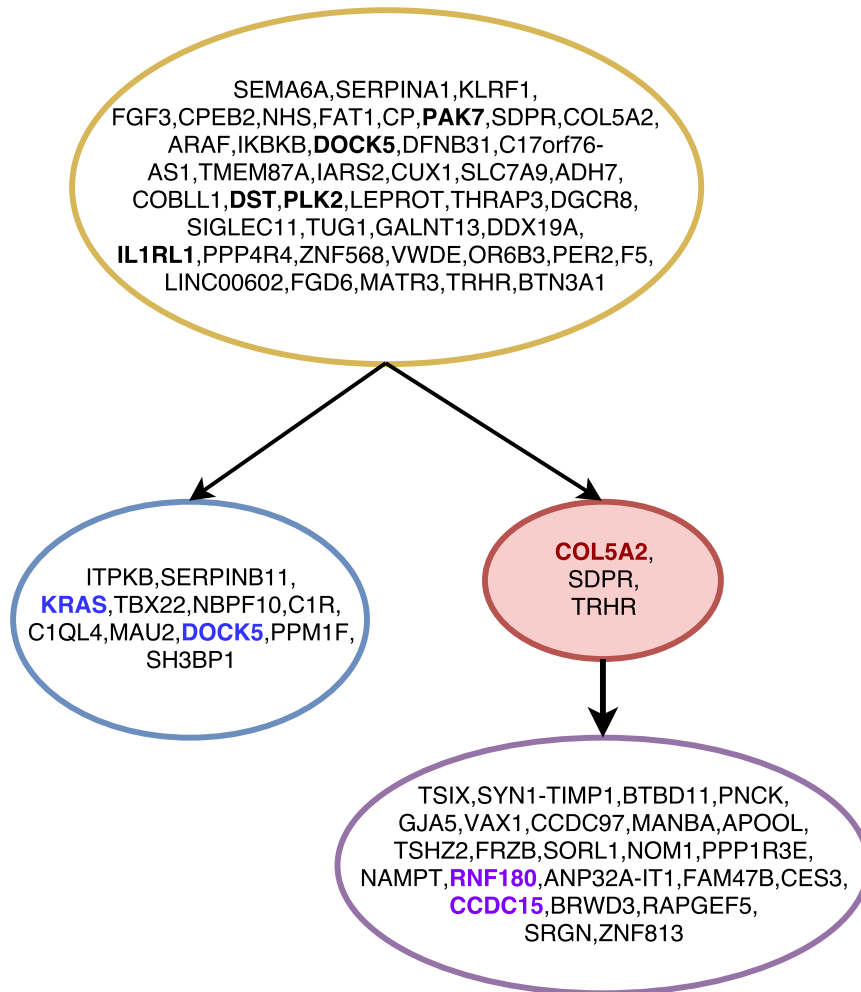


Figure 3.12: The tree inferred by *SASC* for Patient 4 of the Childhood Lymphoblastic Leukemia data from [50]. Different clones are indicated with different colors. Red nodes indicate deletions of mutations, while boldfaced mutations are the mutations indicated as driver in the original sequencing study. Mutations in bold and colored are driver mutations for the clone with the same color. Mutations are clustered by collapsing simple linear paths. The picture was drawn using the *SASC-viz* post-processing tool.

and inferred trees are still relevant from a biological point of view.

The need for a model that allows mutation losses has been established in [87], but no clear consensus on the model that is most suited to represent the true evolution of tumors has been reached so far, to the best of our knowledge. In our manuscript, we introduce and follow a restricted version of the Dollo- $k$  model, where the number of mutations in each site and the number of overall mutations is limited — even though our method can

be used also in a more relaxed setting. Determining which of the possible models is going to be the basis for effective and efficient tumor phylogeny inference is something that needs to be explored in the future, but it will likely need the development of different methods, and a deeper understanding of the models.

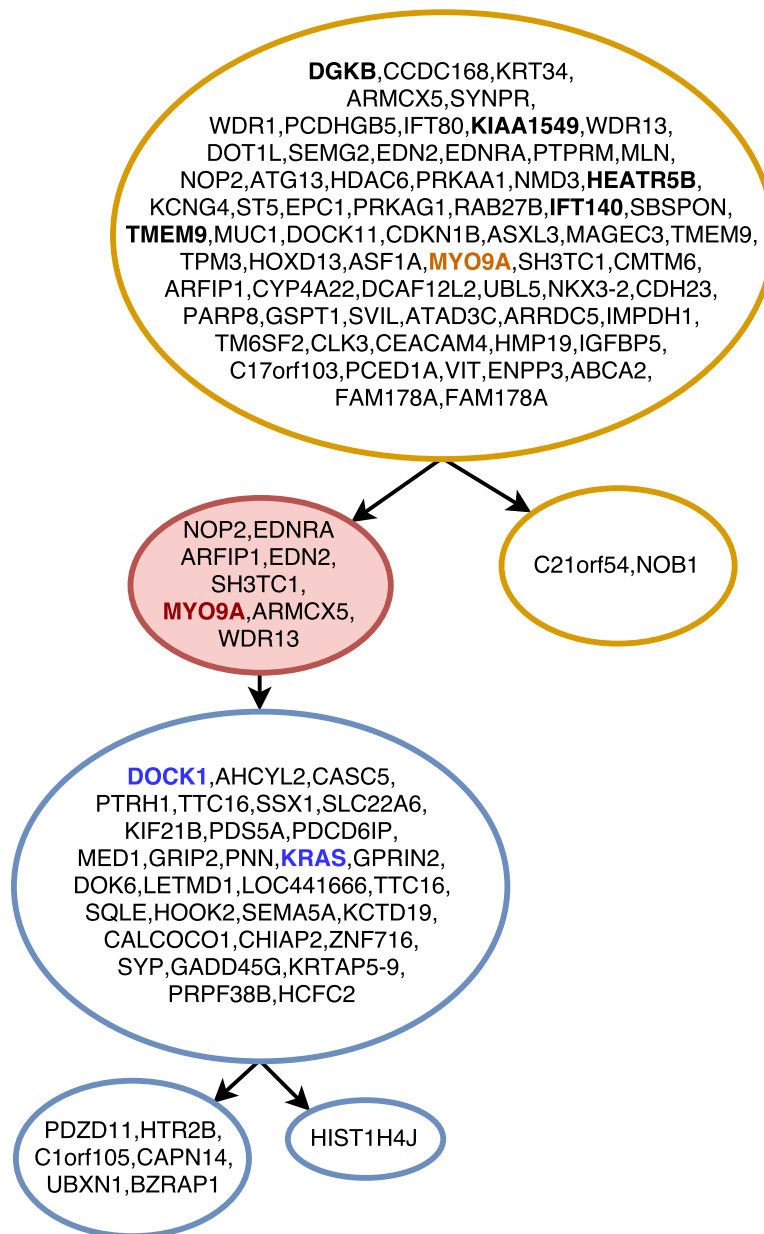


Figure 3.13: Tree inferred by SASC for Patient 5 of the Childhood Lymphoblastic Leukemia data from [50]. Different clones are indicated with different colors, while the red-colored nodes indicate deletions of mutations, and mutations highlighted in bold are the mutations indicated as driver in the original sequencing study. Mutations boldfaced and colored are driver mutations for the same colored clone. Mutations are clustered by collapsing simple linear paths. The picture was drawn using the SASC-viz post-processing tool.



### 3.5 Additional formulations for phylogenetic inference

Even though most of the methods for SCS data analysis are probabilistic, many of the related methods for bulk sequencing data analysis are combinatorial in nature [125, 61, 107, 44, 93, 46]. Combinatorial, in particular integer linear programming (ILP) formulations for phylogeny inference have been available in the literature for a while. One example is the *Haplotype Inference Problem* a.k.a. HIP [59], where given a binary incomplete matrix  $M$  of  $n$  rows (corresponding to *species*) and  $m$  columns (corresponding to *sites*), the goal is to decompose each row to two binary vectors (haplotypes) so that the haplotypes can fit in a *Perfect Phylogeny*, i.e. a phylogeny satisfying ISA. HIP can be formulated and efficiently solved as an instance of ILP. Later, a similar formulation was proposed in [58] to solve the *Persistent Phylogeny Problem* [52, 13]. A persistent phylogeny is one in which each mutation is allowed to be “lost” at most once. Recently, an extension of formulation from [58] was proposed in [16], where more general phylogeny models are used and the goal is to infer entire cancer phylogenies by the use of bulk sequencing data.

Finally, the notion of *flip distance* was introduced in [108] and later explored in [22], to compare a matrix  $M$  (see above) that does not admit a perfect phylogeny with  $M'$ , a matrix admitting a perfect phylogeny that differs from  $M$  as little as possible. As will be seen, our method builds on this notion of distance.

ILP formulations for HIP and related problems are routinely solved through commercial tools such as Gurobi or IBM CPLEX - which have been developed over many years and provide reliable and fast solutions for relatively small sized optimization problems. These solvers aim to optimize a typically linear objective while satisfying a number of linear constraints. As such, ILP is related to another fundamental problem, the *Boolean Constraint Satisfaction Problem* (CSP) that can be used as an alternative for modeling many ILP problems encountered in practice.

Perhaps the best-known variant of CSP is the satisfiability problem (SAT) which asks to find a Boolean assignment to a set of input variables to satisfy (the conjunction of) a number of Boolean constraints. Another variant is Max-SAT, which asks to find a Boolean assignment to variables such that not necessarily all but the maximum number of input constraints are satisfied. Finally, the weighted version of Max-SAT, which can be abbreviated as wMax-SAT, asks for the assignment that maximizes the sum of (user defined) weights of the constraints satisfied. The generality of wMax-SAT has prompted the development of many tools to solve them with the goal of obtaining solutions to practical instances of NP-complete problems. These tools compete in the annual SAT conference on



several benchmarking datasets generated by a wide variety of applications (see <http://www.satcompetition.org>). Recently developed wMax-SAT solvers such as MAXINO [8] and MaxHS [34, 35, 36], are very fast; in addition MaxHS is open source. A number of studies had already demonstrated the utility of CSP solvers for the haplotype inference problem and its variants - before the advent of high throughput sequencing [91, 101, 64].

We here describe a combinatorial formulation to solve this problem which ensures that several lineage constraints imposed by the use of variant allele frequencies (VAFs, derived from bulk sequence data) are satisfied. We express our formulation both in the form of an integer linear program (ILP) and - as a first in tumor phylogeny reconstruction - a Boolean constraint satisfaction problem (CSP) and solve them by leveraging state-of-the-art ILP/CSP solvers. The resulting method, which we name PhISCS, is the first to integrate SCS and bulk sequencing data while accounting for ISA violating mutations. In contrast to the alternative methods, typically based on probabilistic approaches, PhISCS provides a guarantee of optimality in reported solutions.

This chapter only focuses on the formal definitions of the computational tools and does not go into the details of the results and experimental analysis, as they tend to be similar in execution to the ones described previously; therefore we chose to omit them in this section and describe only the more interesting algorithmic alternative approaches to the problem of the cancer phylogeny inference.

### 3.5.1 Combinatorial ILP formulations

We start describing the algorithmic underpinnings of PhISCS by formulating integrative tumor phylogeny reconstruction as a combinatorial optimization problem. We first focus two special cases of the problem for the instance in which only single-cell sequencing data is available: (i) a special case where the ISA cannot be violated, (ii) the case where ISA can be violated. We then describe the general integrative problem where both bulk and SCS data are available. We present solutions for this problem in the form of a novel Integer Linear Program (ILP) as well as a Constraint Satisfaction Program (CSP).

#### PhISCS-I for Tumor Phylogeny Inference via SCS Data with no ISA Violations Allowed

The input to as a ternary matrix  $I$  with  $n$  rows and  $m$  columns, where columns represent mutations and rows represent genotypes of single cells observed in a single-cell sequencing experiment. For a given entry,  $I(i, j) = 0$  indicates the absence,  $I(i, j) = 1$  indicates the presence and  $I(i, j) = ?$  indicates the lack of knowledge about absence or presence (i.e. missing

entry) of a mutation  $j$  in a cell  $i$ .

We ask to find the minimum weighted number of *bit flips* (typically from 0 to 1 and rarely from 1 to 0) and *bit assignments* (assigning a 0 or 1 to an entry with value ?), where *bit assignments* are not a part of the objective - such that the resulting matrix provides a Perfect Phylogeny (PP). We recall that a binary matrix is a PP if the *three-gametes rule* holds, i.e. for any given pair of columns (mutations) there are no three rows (cells) with configuration (1, 0), (0, 1) and (1, 1). *Bit flipping* in the input matrix  $I$  is essential to building a PP as some mutation inferences in  $I$  are false positives and some mutations are not indicated in  $I$  (false negatives) as they do not have sufficient read support in sequenced single cells. We name any pair of mutations and triplet of cells violating three-gametes rule as a *conflict* and refer to PP matrix also as a *conflict-free matrix*.

To allow correction of noisy genotypes in  $I$  (i.e. bit flips and bit assignments), for each cell  $i$  and mutation  $j$ , we introduce a binary variable  $Y(i, j)$  which denotes the (unknown) true status (i.e. absence or presence) of the mutation  $j$  in the cell  $i$ . If  $\alpha$  and  $\beta$  respectively denote false positive and false negative error rates of single-cell data (as per other methods [92, 75, 113, 138, 137], we assume that  $\alpha$  and  $\beta$  are fixed parameters), we have:

$$\begin{aligned} P(I(i, j) = 0 \mid Y(i, j) = 0) &= (1 - \alpha) & P(I(i, j) = 0 \mid Y(i, j) = 1) &= \beta \\ P(I(i, j) = 1 \mid Y(i, j) = 0) &= \alpha & P(I(i, j) = 1 \mid Y(i, j) = 1) &= (1 - \beta). \end{aligned} \tag{3.1}$$

Assuming that the mutated loci are independent and that the missing entries in  $I$  are non-informative (i.e. bit assignments are not part of the objective), we define the likelihood of an arbitrary conflict-free matrix  $Y$  as:

$$P(I \mid Y) = \prod_{(i, j) \in \mathcal{S}} P(I(i, j) \mid Y(i, j)) \tag{3.2}$$

where  $\mathcal{S}$  is set of all pairs of integers  $(i, j)$  such that  $1 \leq i \leq n$ ,  $1 \leq j \leq m$  and  $I(i, j) \in \{0, 1\}$ .

Here, our goal is to find a conflict-free matrix  $Y$  such that the likelihood  $P(I \mid Y)$  defined in equation (3.2) is maximized.

Now, observe that (3.1) can be rewritten as:

$$\begin{aligned} P(I(i, j) = 0 \mid Y(i, j)) &= (1 - \alpha)^{1 - Y(i, j)} \cdot \beta^{Y(i, j)} = (1 - \alpha) \cdot \left( \frac{\beta}{1 - \alpha} \right)^{Y(i, j)}, \\ P(I(i, j) = 1 \mid Y(i, j)) &= \alpha^{1 - Y(i, j)} \cdot (1 - \beta)^{Y(i, j)} = \alpha \cdot \left( \frac{1 - \beta}{\alpha} \right)^{Y(i, j)}. \end{aligned} \tag{3.3}$$

and our objective is equivalent to maximizing the logarithm of  $P(I | Y)$  which can be expressed as:

$$\begin{aligned} & \sum_{(i,j):I(i,j)=0} \left[ \log(1 - \alpha) + \log \frac{\beta}{1 - \alpha} Y(i, j) \right] + \\ & \sum_{(i,j):I(i,j)=1} \left[ \log(\alpha) + \log \frac{1 - \beta}{\alpha} Y(i, j) \right]. \end{aligned} \quad (3.4)$$

In order to enforce that matrix  $Y$  satisfies the three-gametes rule, for each pair of mutations  $(p, q)$ , we first introduce variables  $B(p, q, a, b)$ , for each  $(a, b) \in \{(0, 1), (1, 0), (1, 1)\}$ . The variable  $B(p, q, a, b)$  is set to 1 if there exists row  $r$  such that  $Y(r, p) = a$  and  $Y(r, q) = b$ . This property of matrix  $B$  is guaranteed by adding the following constraints for all  $1 \leq i \leq n$  and  $1 \leq p, q \leq m$ :

$$Y(i, p) + Y(i, q) - B(p, q, 1, 1) \leq 1 \quad (3.5)$$

$$-Y(i, p) + Y(i, q) - B(p, q, 0, 1) \leq 0 \quad (3.6)$$

$$Y(i, p) - Y(i, q) - B(p, q, 1, 0) \leq 0. \quad (3.7)$$

$$(3.8)$$

Now, adding constraints

$$B(p, q, 0, 1) + B(p, q, 1, 0) + B(p, q, 1, 1) \leq 2 \quad (3.9)$$

for all  $1 \leq p, q \leq m$  suffices to ensure that three-gametes rule holds for matrix  $Y$ .

The problem defined above represents an instance of ILP and can be solved using any of the standard ILP solvers.

### Allowing ISA violations in PhISCS-I

As we have already discussed in the Introduction, recent evidence suggests that the ISA might be violated for a subset of mutations in the input data. To account for this, we introduce a more general version of what we discussed in the previous section where we allow *elimination* (i.e. deletion from the input matrix) of a given (maximum) number of mutations which do not satisfy ISA; the remaining mutations, after genotype corrections, are expected to satisfy PP. In order to achieve this, for each mutation  $q$  we introduce binary variable  $K(q)$  which is set to 1 if and only if mutation  $q$  is among eliminated mutations. To exclude eliminated mutations from three-gametes rule, we do not require mutational pairs  $(p, q)$ , where at least one of  $p$  and  $q$  is among eliminated mutations, to fulfill this rule. Therefore we modify constraint (3.9) from the integer linear program described above by replacing it with:

$$B(p, q, 0, 1) + B(p, q, 1, 0) + B(p, q, 1, 1) \leq 2 + K(p) + K(q). \quad (3.10)$$

The objective defined in (3.4) is also modified so that the eliminated mutations do not contribute to the objective score. This leads to the following objective to handle the case allowing ISA violations:

$$\begin{aligned} & \sum_{(i,j):I(i,j)=0} (1 - K(j)) \cdot \left[ \log(1 - \alpha) + \log \frac{\beta}{1 - \alpha} Y(i, j) \right] + \\ & \sum_{(i,j):I(i,j)=1} (1 - K(j)) \left[ \log(\alpha) + \log \frac{1 - \beta}{\alpha} Y(i, j) \right]. \end{aligned} \quad (3.11)$$

All other constraints used previously in the limited version of the problem are preserved. Note that the above objective contains quadratic terms (of the form  $K(j)Y(i, j)$ ) which can be transformed to linear variables using standard linearization techniques. One can observe that mutation elimination never decreases data likelihood hence the global optimum in the above maximization problem is achieved when all variables  $K$  are set to 1. However, in real applications we expect only a limited number of ISA violating mutations and therefore set the upper bound  $k_{max}$  on the number of eliminated mutations which is implemented by the addition of the following constraint

$$\sum_{q=1}^m K(q) \leq k_{max}, \quad (3.12)$$

where  $k_{max}$  is an empirically estimated constant. It is also possible to computationally estimate  $k_{max}$ .

### Additional ILP Constraints to Integrate VAFs Derived from Bulk Sequencing Data into PhISCS-I

Now we show how to integrate SCS data with bulk sequencing data - specifically the VAF of each mutation we consider - through additional linear constraints. These constraints will only apply to the set of single nucleotide variants from the regions not affected by copy number aberrations. Suppose that a particular SNV, denoted  $M$ , satisfies the above requirement; let  $v$  and  $r$  respectively denote the number of reads supporting the variant and the reference allele at the genomic locus of  $M$ . The VAF of  $M$  is typically defined as  $\frac{v}{v+r}$ . Since we are interested in *cellular prevalence* rather than the VAF below, we define  $\text{vaf}(M) = \frac{2v}{v+r}$ . (Cellular prevalence represents the expected fraction of cells in the sample that harbor  $M$ .)

Before defining constraints related to VAFs, we first define the *root node* via a new row, indexed by 0, that represents genotype of a healthy cell. We also add a new column, indexed by 0, and associated *null mutation*  $M_0$  which represents mutation specific to the normal cell or, in other words, germline SNP present in all cells. We set  $I(t, 0) = 1$  for  $t = 0, 1, \dots, n$  and  $I(0, p) = 0$  for  $p = 1, 2, \dots, m$ . We also set  $\text{vaf}(M_0) = 1$  and do not allow elimination

of  $M_0$ . Matrices  $B$  and  $Y$  are also expanded in an obvious way by allowing mutational indices to be equal to 0. The remainder of the tree topology is imposed through additional constraints that specify ancestor-descendent relationships in a consistent manner across all nodes.

1. We must satisfy the following constraints: (i)  $K(0) = 0$ , (ii)  $Y(t, 0) = 1$  for  $t = 0, 1, \dots, n$ , and (iii)  $Y(0, p) = 0$  for  $p = 1, 2, \dots, m$ .
2. If a mutation  $p$  is an ancestor of a mutation  $q$  and ISA holds for both  $p$  and  $q$  then the true cellular prevalence of  $p$  is expected to be greater than or equal to true cellular prevalence of  $q$ . Since  $\text{vaf}(p)$  and  $\text{vaf}(q)$  reflect cellular prevalences as discussed above, we expect that in the implied evolutionary tree  $\text{vaf}(p)(1 + \delta) \geq \text{vaf}(q)$ , where  $\delta$  is some positive constant which allows for the noise typically present in the observed VAFs. In order to incorporate VAFs in our model, we introduce binary function  $a$ , such that  $a(p, q) = 1$  only if  $p$  is an ‘‘ancestor’’ of  $q$ . By definition we set  $a(p, p) = 0$  for all  $p \in \{0, 1, \dots, m\}$ . The constraints that we need to introduce are thus as follows.

- (a) For any pair of distinct mutations  $p, q$  we must satisfy the following two constraints to ensure that (i) only one of them could be the ancestor of the other, and (ii) if there is a cell in which they appear together, then one must be the ancestor of the other (this also eliminates the possibility of nodes with multiple mutation assignments):

$$\begin{aligned} a(p, q) + a(q, p) &\leq \min\{1 - K(p), 1 - K(q)\} \\ a(p, q) + a(q, p) &\geq B(p, q, 1, 1) - K(p) - K(q) \end{aligned} \quad (3.13)$$

(we remind the reader that for any mutation  $r$ ,  $K(r) = 1$  indicates that the column  $r$  in input matrix  $I$  has been eliminated).

- (b) Each non-eliminated mutation  $q$  different from null mutation must have at least one ancestor. This is ensured by adding the following constraint:

$$\sum_{p=0}^m a(p, q) \geq 1 - K(q). \quad (3.14)$$

On the other hand, null mutation has no ancestors so we set  $a(p, 0) = 0$  for all  $p \in \{0, 1, \dots, m\}$ .

- (c) Consider two non-eliminated mutations  $p$  and  $q$ . If  $a(p, q) = 1$  then in genotype corrected output matrix  $Y$  the column  $p$  should dominate the column  $q$  - i.e. for each cell/row  $r$  if the entry for  $p$  is 0 then the entry for  $q$  should also be 0. In other words, there should not exist row  $r$  such that  $Y(r, p) = 0$  and  $Y(r, q) = 1$ ,

which is equivalent to  $B(p, q, 0, 1) = 0$ . To ensure this, for all pairs of mutation  $(p, q)$ , we add the following constraint:

$$a(p, q) \leq 1 - B(p, q, 0, 1). \quad (3.15)$$

- (d) If, for two non-eliminated mutations  $p$  and  $q$ , matrix  $Y$  contains a cell in which  $p$  is present and  $q$  is absent (i.e. there exists  $i$  such that  $Y(i, p) = 1$  and  $Y(i, q) = 0$ , which is equivalent to  $B(p, q, 1, 0) = 1$ ), as well as a cell where both  $p$  and  $q$  are present (i.e. there exists  $j$  such that  $Y(j, p) = 1$  and  $Y(j, q) = 1$ , which is equivalent to  $B(p, q, 1, 1) = 1$ ), then  $p$  must be ancestor of  $q$  (i.e.  $a(p, q) = 1$ ). In order to ensure this, for all pairs of mutations  $(p, q)$  we add the following constraints:

$$a(p, q) \geq B(p, q, 1, 0) + B(p, q, 1, 1) - 1 - K(p) - K(q). \quad (3.16)$$

- (e) For some small user defined error tolerance value  $\delta > 0$  that accounts for variation in sequence coverage, if  $\text{vaf}(q) > \text{vaf}(p)(1 + \delta)$  then  $a(p, q) = 0$ ; in other words for every pair of mutations  $p$  and  $q$  we must satisfy:

$$a(p, q) \cdot \text{vaf}(p) \cdot (1 + \delta) \geq a(p, q) \cdot \text{vaf}(q). \quad (3.17)$$

If more than one sample from the same tumor with (independent) bulk sequencing data are available, we will have to satisfy the VAF constraints imposed by all of them. Let  $\text{vaf}_\ell(p)$  denote  $\text{vaf}(p)$  in sample  $\ell$ . Then for each pair of mutations  $p$  and  $q$  such that  $\text{vaf}_\ell(q) > \text{vaf}_\ell(p)(1 + \delta)$  we must satisfy:  $a(p, q) = 0$ ; i.e. for each sample  $\ell$ :

$$a(p, q) \cdot \text{vaf}_\ell(p) \cdot (1 + \delta) \geq a(p, q) \cdot \text{vaf}_\ell(q) \quad (3.18)$$

- (f) For all triplet of mutations  $p, q, r$ , we must ensure that if  $a(p, q) = 1$  and  $a(q, r) = 1$  then  $a(p, r) = 1$ :

$$\forall p, q, r : a(p, r) \geq a(p, q) + a(q, r) - 1. \quad (3.19)$$

3. Now we can describe our constraint for every triplet of distinct mutations  $p, q$  and  $r$ , such that  $p$  is an ancestor of  $q$  and  $r$  but  $q$  and  $r$  do not have an ancestor descendant relationship (i.e.  $a(p, q) = a(p, r) = 1$  and  $a(q, r) = a(r, q) = 0$ ).

$$\begin{aligned} \text{vaf}(p) \cdot (1 + \delta) &\geq \text{vaf}(q) \cdot [a(p, q) - a(r, q) - a(q, r)] + \\ &\quad \text{vaf}(r) \cdot [a(p, r) - a(r, q) - a(q, r)]. \end{aligned} \quad (3.20)$$

If again, multiple samples with (independent) bulk sequencing data are available, we have to satisfy the triple-VAF constraint for each sample  $\ell$ , i.e. for each triplet of mutations  $p, q, r$ :

$$\begin{aligned} \text{vaf}_\ell(p) \cdot (1 + \delta) \geq & \text{vaf}_\ell(q) \cdot [a(p, q) - a(r, q) - a(q, r)] + \\ & \text{vaf}_\ell(r) \cdot [a(p, r) - a(r, q) - a(q, r)]. \end{aligned} \quad (3.21)$$

Note that the above *triple-VAF* constraint does not fully utilize the information provided by VAFs, e.g. in case a *parent* mutation has three distinct *children* whose total VAF should, in principle, not exceed that of the parent. It is possible to generalize the triple-VAF constraint to any number of children. Nevertheless we still recommend the use of the triple-VAF constraint instead of this general-VAF constraint (even though this choice may, in principle, produce trees that violate the general-VAF constraint) since the two sets of constraints do not seem to produce different trees in practice. Furthermore the general-VAF constraint is quadratic and thus slows down PhISCS.

### 3.5.2 Combinatorial CSP formulations

#### PhISCS-B for Tumor Phylogeny Inference via SCS Data

In this section we first show how to reduce the ILP formulation of PhISCS where only single-cell data is used as the input and no mutation elimination allowed to a wMax-SAT problem. For each input entry  $I(i, j)$ ,  $1 \leq i \leq n$ ,  $1 \leq j \leq m$ , we introduce a Boolean variable  $Y(i, j)$  which represents the true state of mutation  $j$  in cell  $i$ . Our goal is to find the assignment of values to variables  $Y(i, j)$  such that the resulting matrix  $Y$  is conflict-free and the objective defined below is maximized. In order to enforce that  $Y$  is conflict-free matrix, we use a set of additional Boolean variables  $B(p, q, a, b)$  (analogous to binary variables used in earlier sections) that need to satisfy the following *hard* constraints (the constraints that *need* to be satisfied):

$$\begin{aligned} & \neg(Y(i, p) \wedge Y(i, q) \wedge \neg B(p, q, 1, 1)) \\ & \neg(\neg Y(i, p) \wedge Y(i, q) \wedge \neg B(p, q, 0, 1)) \\ & \neg(Y(i, p) \wedge \neg Y(i, q) \wedge \neg B(p, q, 1, 0)) \\ & \neg(B(p, q, 0, 1) \wedge B(p, q, 1, 0) \wedge B(p, q, 1, 1)). \end{aligned} \quad (3.22)$$

We can now define our objective as satisfying all the hard constraints with alterations on the input matrix  $I$  with maximum probability, where each alteration (indicating a false positive or false negative) is independent. This objective corresponds to the minimizing the (weighted) number of flipped entries in the solution matrix  $Y$  in comparison to  $I$ , or, for the purpose of formulating the problem as an instance of wMax-SAT, maximizing the

weighted sum of the following “soft” constraints (for all  $i, j$ , s.t.  $I(i, j) \neq ?$  originally):

$$\begin{aligned} \text{if } I(i, j) = 0 \text{ weight for } Y(i, j) \text{ is: } & \log \frac{\beta}{1 - \alpha} \\ \text{if } I(i, j) = 1 \text{ weight for } Y(i, j) \text{ is: } & \log \frac{1 - \beta}{\alpha}. \end{aligned} \quad (3.23)$$

Note that in order to get exactly the same objective value as in objective defined in (3.4) we need to add constant terms from (3.4) to the objective defined in (3.23). Alternatively, after solving for matrix  $E$ , we can compute  $P(D | E)$  by the use of formula given in (3.2).

We now show how to account for ISA violations: for each column  $j \in \{1, 2, \dots, m\}$  we introduce a Boolean variable  $K(j)$  that is set to 1 if and only if column  $j$  is eliminated (i.e. mutation corresponding to column  $j$  is not considered as a part of the output).

Analogously as in the ILP formulation, we allow at most  $k_{\max}$  columns to be eliminated, where  $k_{\max}$  is a user-defined constant. In order to ensure that no more than  $k_{\max}$  of variables  $K(1), K(2), \dots, K(m)$  are set to 1, for each possible  $(k_{\max} + 1)$ -tuple  $(i_1, i_2, \dots, i_{k_{\max}+1})$  of integers such that  $1 \leq i_1 < i_2 < \dots < i_{k_{\max}+1} \leq m$  we add the the following hard clause

$$\neg \left( K(i_1) \wedge K(i_2) \wedge \dots \wedge K(i_{k_{\max}+1}) \right) \quad (3.24)$$

to our model.

Now, for any eliminated column  $p$  we do not have to check whether it is in conflict with any other column  $q$  or vice versa. Therefore, for each pair  $(p, q)$  of columns we replace the constraint (3.22) above with the following.

$$\neg(\neg K(p) \wedge \neg K(q) \wedge B(p, q, 0, 1) \wedge B(p, q, 1, 0) \wedge B(p, q, 1, 1)). \quad (3.25)$$

To get the objective equivalent to (3.11) for each pair of cell  $i$  and mutation  $j$  we introduce a binary variable  $X(i, j)$  and add the following hard constraint:

$$\left( \neg Y(i, j) \vee \neg K(j) \vee X(i, j) \right) \wedge \left( Y(i, j) \vee \neg X(i, j) \right) \wedge \left( K(j) \vee \neg X(i, j) \right) \quad (3.26)$$

and transform (3.11) to an instance of wMax-SAT where the goal is to maximize weighted sum of the following “soft” constraints (for all  $i, j$ , s.t.  $I(i, j) \neq ?$ ):



$$\begin{aligned}
&\text{if } I(i, j) = 0 \text{ weight for } \neg K(j) \text{ is: } \log(1 - \alpha) \\
&\text{if } I(i, j) = 1 \text{ weight for } \neg K(j) \text{ is: } \log(\alpha) \\
&\text{if } I(i, j) = 0 \text{ weight for } X(i, j) \text{ is: } -\log \frac{\beta}{1 - \alpha} \\
&\text{if } I(i, j) = 1 \text{ weight for } X(i, j) \text{ is: } -\log \frac{1 - \beta}{\alpha} \\
&\text{if } I(i, j) = 0 \text{ weight for } Y(i, j) \text{ is: } \log \frac{\beta}{1 - \alpha} \\
&\text{if } I(i, j) = 1 \text{ weight for } Y(i, j) \text{ is: } \log \frac{1 - \beta}{\alpha}. \tag{3.27}
\end{aligned}$$

### Additional Boolean Constraints to Integrate VAFs Derived from Bulk Sequencing Data into PhISCS-B

In order to integrate information derived from bulk sequencing data, represented in the form of VAFs of the given set of mutations, we explicitly impose a tree structure on the output matrix  $Y$  through the use of a number of Boolean constraints.

The Boolean constraints below start by defining the *root node* via a new row, indexed by 0, that represents genotype of a normal cell. We also add a new column, indexed by 0, and associated *null mutation*  $M_0$  which represents mutation specific to the normal cell or, in other words, germline SNP present in all cells. We set  $I(t, 0) = 1$  for  $t = 0, 1, \dots, n$  and  $I(0, p) = 0$  for  $p = 1, 2, \dots, m$ . We also set  $\text{vaf}(M_0) = 1$  and do not allow elimination of  $M_0$ . The remainder of the tree topology is imposed through additional constraints that specify ancestor-descendent relationships in a consistent manner across all nodes:

1. We must satisfy the following constraints which can easily be converted into Boolean expressions: (i)  $K(0) = 0$ , (ii)  $Y(t, 0) = 1$  for  $t = 0, 1, \dots, n$ , and (iii)  $Y(0, p) = 0$  for  $p = 1, 2, \dots, m$ .
2. If a mutation  $p$  is an ancestor of mutation  $q$  in the implied evolutionary tree, then  $\text{vaf}(p) \geq \text{vaf}(q)$  within some relatively small error tolerance. In order to employ VAFs using the above constraints between mutational pairs, we introduce Boolean function  $a$  such that  $a(p, q) = 1$  if and only if  $p$  is an ancestor of  $q$ . The hard constraints that need to be imposed on  $a$  are as follows.

- (a) For all pairs of distinct mutations  $p$  and  $q$ , where both  $p$  and  $q$  are different from null mutation, we must satisfy:

$$\begin{aligned}
a(p, q) \vee a(q, p) &\Rightarrow \neg K(p) \wedge \neg K(q) \\
&\quad \neg[(a(p, q) \wedge a(q, p))] \\
B(p, q, 1, 1) &\Rightarrow a(p, q) \vee a(q, p). \tag{3.28}
\end{aligned}$$

- (b) For each non-eliminated mutation  $q$  different from null mutation we must make sure that it has an ancestor mutation (which could be null mutation). This is achieved by imposing the following constraint:

$$\left( \bigvee_{\forall p \neq q} a(p, q) \right) \vee K(q). \quad (3.29)$$

- (c) Consider two non-eliminated mutations  $p$  and  $q$ . If  $a(p, q) = 1$  then in genotype corrected output matrix  $Y$  the column  $p$  should dominate the column  $q$  - i.e. for each cell/row  $r$  if the entry for  $p$  is 0 then the entry for  $q$  should also be 0. In other words, there should not exist row  $r$  such that  $Y(r, p) = 0$  and  $Y(r, q) = 1$ , which is equivalent to  $B(p, q, 0, 1) = 0$ . To ensure this, for all pairs of mutation  $(p, q)$ , we add the following constraint:

$$\neg a(p, q) \vee \neg B(p, q, 0, 1) \vee K(p) \vee K(q). \quad (3.30)$$

- (d) If, for two non-eliminated mutations  $p$  and  $q$ , matrix  $Y$  contains cell in which  $p$  is present and  $q$  is absent (i.e. there exists  $i$  such that  $Y(i, p) = 1$  and  $Y(i, q) = 0$ , which is equivalent to  $B(p, q, 1, 0) = 1$ ), as well as cell where both  $p$  and  $q$  are present (i.e. there exists  $j$  such that  $Y(j, p) = 1$  and  $Y(j, q) = 1$ , which is equivalent to  $B(p, q, 1, 1) = 1$ ), then  $p$  must be ancestor of  $q$  (i.e.  $a(p, q) = 1$ ). In order to ensure this, for all pairs of mutations  $(p, q)$  we add the following constraints:

$$(B(p, q, 1, 0) \wedge B(p, q, 1, 1)) \Rightarrow (a(p, q) \vee K(p) \vee K(q)). \quad (3.31)$$

- (e) For some small user defined error tolerance value  $\delta > 0$  that accounts for variation in bulk sequencing coverage, if  $\text{vaf}(q) > \text{vaf}(p) \cdot (1 + \delta)$  then  $a(p, q) = 0$ ; in other words for each pair of mutations  $p$  and  $q$  for which  $a(p, q) = 1$ , we must satisfy  $\text{vaf}(p) \cdot (1 + \delta) \geq \text{vaf}(q)$ . In order to express this as a Boolean constraint we introduce a new Boolean function  $\text{Pvaf}(p, q)$  defined for all pairs of mutations  $p$  and  $q$  (as a part of the input specification) as follows:

$$\begin{aligned} \text{Pvaf}(p, q) &= 1, \text{ if } \text{vaf}(p) \cdot (1 + \delta) \geq \text{vaf}(q) \\ &= 0, \text{ otherwise.} \end{aligned} \quad (3.32)$$

Then the constraint that must be satisfied for each pair of mutations  $p$  and  $q$  are:

$$a(p, q) \Rightarrow \text{Pvaf}(p, q). \quad (3.33)$$

If more than one sample from the same tumor with (independent) bulk sequencing data are available, we will have to satisfy the VAF constraints imposed by all of them. Specifically, let  $\text{vaf}_\ell(p)$  denote  $\text{vaf}(p)$  in sample  $\ell$ . Then for each pair of mutations  $p$  and  $q$ ,  $\text{Pvaf}(p, q) = 1$  only if  $\text{vaf}_\ell(p) \cdot (1 + \delta) \geq \text{vaf}_\ell(q)$  for all samples  $\ell$ , and  $\text{Pvaf}(p, q) = 0$ , otherwise.

- (f) For all triplet of mutations  $p, q, r$ , we must ensure that if  $a(p, q) = 1$  and  $a(q, r) = 1$  then  $a(p, r) = 1$ :

$$\forall p, q, r : a(p, q) \wedge a(q, r) \Rightarrow a(p, r). \quad (3.34)$$

3. For all triplet of distinct mutations  $p, q$  and  $r$  such that  $p$  is an ancestor of  $q$  and  $r$ , but  $q$  and  $r$  do not have an ancestor descendant relationship (i.e. they belong to different lineages in the tree), we must satisfy  $\text{vaf}(p) \cdot (1 + \delta) \geq \text{vaf}(q) + \text{vaf}(r)$ . In order to express this as a Boolean constraint we introduce yet another Boolean function  $\text{Tvaf}(p, q, r)$  defined for all triplet of mutations  $p, q, r$  (as a part of the input specification) as follows:

$$\begin{aligned} \text{Tvaf}(p, q, r) &= 1, \text{ if } \text{vaf}(p) \cdot (1 + \delta) \geq \text{vaf}(q) + \text{vaf}(r) \\ &= 0, \text{ otherwise.} \end{aligned} \quad (3.35)$$

Then the constraint that must be satisfied for all mutations  $p, q, r$  is:

$$[a(p, q) \wedge a(p, r) \wedge \neg a(q, r) \wedge \neg a(r, q)] \implies \text{Tvaf}(p, q, r). \quad (3.36)$$

If multiple samples from the same tumor with (independent) bulk sequencing data are available, we will have  $\text{Tvaf}(p, q, r) = 1$  if  $\text{vaf}_\ell(p) \cdot (1 + \delta) \geq \text{vaf}_\ell(q) + \text{vaf}_\ell(r)$  for all  $\ell$ .

## Chapter 4

# Clustering of SCS cancer data

Single cell sequencing (SCS) technologies provide a level of resolution that makes it indispensable for inferring from a sequenced tumor, evolutionary trees or phylogenies representing an accumulation of cancerous mutations. A drawback of SCS is elevated false negative and missing value rates, resulting in a large space of possible solutions, which in turn makes it difficult, sometimes infeasible using current approaches and tools. One possible solution is to reduce the size of an SCS instance — usually represented as a matrix of presence, absence, and uncertainty of the mutations found in the different sequenced cells — and to infer the tree from this reduced-size instance. In this work, we present a new clustering procedure aimed at clustering such *categorical* vector, or matrix data — here representing SCS instances, called *celluloid*. We show that celluloid clusters mutations with high precision: never pairing too many mutations that are unrelated in the ground truth, but also obtains accurate results in terms of the phylogeny inferred downstream from the reduced instance produced by this method. We demonstrate the usefulness of a clustering step by applying the entire pipeline (clustering + inference method) to a real dataset, showing a significant reduction in the runtime, raising considerably the upper bound on the size of SCS instances which can be solved in practice. Our approach, celluloid: *clustering single cell sequencing data around centroids* is available at <https://github.com/AlgoLab/celluloid/> under an MIT license, as well as on the *Python Package Index* (PyPI) at <https://pypi.org/project/celluloid-clust/>

### 4.1 Introduction

The rise of *next-generation sequencing* (NGS) technologies has led to the computational problem of tumor phylogeny inference from NGS bulk se-

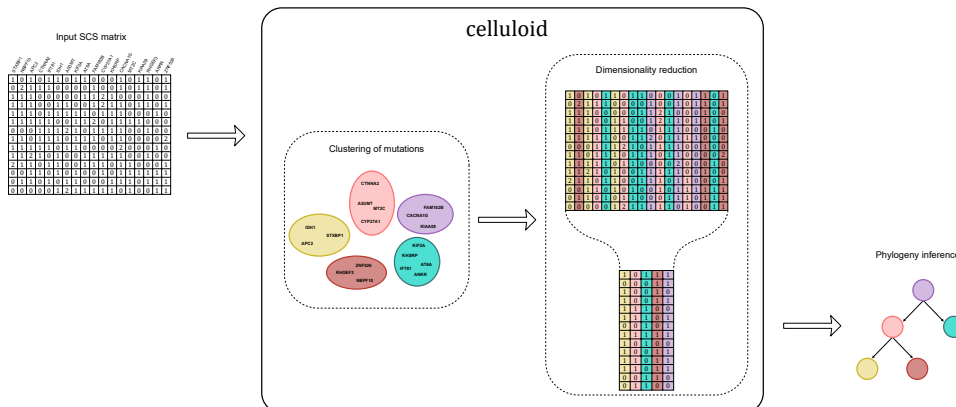


Figure 4.1: A schematic overview of how *celluloid* works, and how it fits into a cancer phylogeny inference pipeline. When an input SCS matrix (upper left) is too large for a phylogeny inference method (lower right), one can use *celluloid* (middle) as an intermediate step to reduce the dimensionality of this instance, making feasible the phylogeny inference.

quencing data [44, 93, 125, 62]. This idea is very cost-effective, since NGS data is cheap to obtain and very reliable. The procedure consists of extracting different samples of the tumor and aligning the NGS reads against the reference genome: this allows to determine the approximate fraction of reads from each sample that are affected by any given mutation. This fraction is taken as a proxy of the fraction of cells in each sample that are affected by that mutation. The main difficulties of this technique are the fact that a sample contains a mix of both healthy cells and cancer cells, while the cancer cells are an unknown mixture of different clones.

The newer *single cell sequencing* (SCS) technology provides a much finer level of resolution: in fact we can determine whether or not a given cell *has* a mutation, therefore avoiding the notion of sample and the approximations implied by the use of samples. Still, SCS is expensive and plagued by high dropout, *i.e.*, missing values, and false negative rates.

In this paper we devise a new method for clustering categorical data, called *celluloid* — a schematic of this depicted in Figure 4.1. Not only does *celluloid* cluster categorical data (based on a  $k$ -modes framework), but because of its novel *conflict dissimilarity* measure — tailored to properties specific to SCS data (see Section 4.2: Methods) — it obtains the best results in a comparison of a variety of different clustering approaches on SCS data. This comparison has three goals: (1) to assess which clustering methods are more precise on SCS data by comparing the actual clustering produced, (2) to evaluate the effect of a clustering step on the downstream phylogeny inference step, and (3) to assess the usefulness of a clustering step on real data, by showing its capability to reduce an instance that is too large for

some of the current methods, so that the clustered instance can be easily solved.

## 4.2 Methods

Starting with the method we devise, *celluloid*, we now give an overview of the methods we consider for reducing the size of single-cell sequencing (SCS) datasets, by clustering mutations. Specifically, the problem is: given  $m$  objects, each with  $n$  values — each value taken from a set of possible categories — we wish to cluster the  $m$  objects into  $k$  groups. In this context, the objects are  $m$  mutations (columns of an SCS matrix), each one over  $n$  cells (rows of an SCS matrix), while each value can be one of three possibilities — that the mutation is *present* (1), *absent* (0), or *missing* (2) from the cell.

Again, the goal of clustering here is to allow faster downstream phylogeny inference. This is why we devise a method which takes, *a priori*, parameter  $k$ : the desired number of clusters, *i.e.*, the largest number that the downstream inference can complete in a reasonable amount of time. For the same reason, we also restrict our comparison to such methods, for which there is already a wide variety. In light of this, we did not consider the entire field of methods which determine the number of clusters based on properties of the data [85, 12, 99], which could also be appropriate.

### 4.2.1 Celluloid

Clustering methods are based on a notion of distance (or of similarity) between the elements that we want to cluster. In our case, we are dealing with a set  $X$  of  $m$  objects on  $n$  *categorical attributes*  $\{A_1, \dots, A_n\}$  — in other words, each object  $x \in X$  is a point  $\langle x[1], \dots, x[n] \rangle$  of the  $n$ -dimensional space  $\mathcal{A} = A_1 \times A_2 \times \dots \times A_n$  — where an object  $x$  has a *categorical value*  $x[i] \in A_i$  for each attribute  $A_i$ .

An intuitive and widely used notion of distance is the the *dissimilarity measure*  $d_M$ , also called *matching dissimilarity*, between two objects  $x = \langle x[1], \dots, x[n] \rangle$  and  $y = \langle y[1], \dots, y[n] \rangle$  of  $\mathcal{A}$ , defined as:

$$d_M(x, y) = \sum_{i=1}^n \delta_M(x[i], y[i]), \quad (4.1)$$

where  $\delta_M$  is a trivially defined dissimilarity, based on the identity of two categorical values:

$$\delta_M(x[i], y[i]) = \begin{cases} 0 & \text{if } x[i] = y[i] \\ 1 & \text{o.w. } (x[i] \neq y[i]) \end{cases} \cdot \quad (4.2)$$

In our context, the value 2 is special, since it represents *missing* data. Therefore, the fact that a given value is equal to 2 should not be penal-

ized. For this reason, we design a new dissimilarity, based on the notion of *conflict* (Equation 4.4) by adapting the previous matching dissimilarity of Equation 4.2 to obtain what we call the *conflict dissimilarity*, defined as:

$$d_C(x, y) = \sum_{i=1}^n \delta_C(x[i], y[i]), \quad (4.3)$$

and  $\delta_C$  is a slight relaxation of  $\delta_M$ , where 2s are not penalized:

$$\delta_C(x[i], y[i]) = \begin{cases} 0 & \text{if } x[i] = 2 \text{ or } y[i] = 2 \\ \delta_M(x[i], y[i]) & \text{o.w.} \end{cases}. \quad (4.4)$$

Our method *celluloid* extends using the previously defined *conflict dissimilarity* function the  $k$ -modes framework [68, 69] which clusters  $m$  objects  $X$  on  $\mathcal{A}$ , by computing an  $m \times k$  *partition matrix* [63]  $p[\cdot, \cdot]$  — a partitioning of these  $m$  objects into  $k$  clusters — and a set  $Q = \{q_1, \dots, q_k\}$  of  $k$  *modes* and that minimize the following objective function:

$$\sum_{l=1}^k \sum_{j=1}^m p[j, l] d_C(x_j, q_l), \quad (4.5)$$

subject to

$$\begin{aligned} \sum_{l=1}^k p[j, l] &= 1, \quad 1 \leq j \leq m, \\ p[j, l] &\in \{0, 1\}, \quad 1 \leq j \leq m, 1 \leq l \leq k, \\ q_l[i] &= \operatorname{argmin}_{a \in A_i} \sum_{j:p[j,l]=1} \delta_M(x_j[i], a), \quad 1 \leq i \leq n, 1 \leq l \leq k, \end{aligned} \quad (4.6)$$

where  $p[j, l] = 1$  if and only if the object  $x_j$  is placed in the cluster  $C_l$  whose mode is  $q_l$ .

That is, for each  $q_l \in Q$ ,  $q_l[i]$  is one of the possible categorical values of the attribute  $A_i$ . Most precisely,  $q_l[i]$  is a *mode* among the values (of the attribute  $A_i$ ) of the objects that are in the cluster  $C_l$ . Note that every  $A_i = \{0, 1, 2\}$  when considering single cell sequencing data for inferring tumor evolution.

Now, a *mode* of cluster  $C_l \subseteq X$  is a vector  $q_l$  which minimizes:

$$D(C_l, q_l) = \sum_{x \in C_l} d_M(x, q_l). \quad (4.7)$$

Note that  $q_l$  is not necessarily an element of  $C_l$ . The  $k$ -modes algorithm then starts with some initial set  $Q^0$  of  $k$  modes, and an initial collection  $\mathbb{C}^0$  of  $k$  disjoint subsets of  $X$ , and then iterates the operations:

1. compute  $d_C(x, q_l)$ , where  $q_l$  is a mode of cluster  $C_l^0 \in \mathbb{C}$ , for each object  $x$  and each cluster  $C_l^0$ ;
2. for each mode  $q_l$  create an empty cluster  $C_l^1$ ;
3. allocate  $x$  to a cluster  $C_{l'}^1$  minimizing  $d_C(x, q_{l'})$ , hence updating  $p[\cdot, \cdot]$ ; and
4. recompute a new set of modes  $Q^1$  according to the new clusters  $\mathbb{C}^1$ , exploiting the last constraint of Equation 4.6;

until convergence, *i.e.*, until  $Q^{t+1} = Q^t$ . The modes in the third step above are found according to the following theorem, where  $c_r[i]$  is the  $r$ -th categorical value of the attribute  $A_i$  and  $f(A_i = c_r[i]|X)$  is its relative frequency in  $X$ .

**Theorem 5** ([69]). *Eq. 4.7 is minimized iff  $f(A_i = q_l[i]|X) \geq f(A_i = c_r[i]|X)$  for  $q_l[i] \neq c_r[i] \forall i \in [1 \dots n]$ .*

In other words, Eq. 4.7 is minimized by selecting any mode value for each attribute. Note that this theorem implies that the mode of  $X$  is not necessarily unique.

Since the solution depends on the initial set  $Q^0$  of  $k$  modes, we consider two procedures for initializing  $Q^0$ . The first one is quite simple: a random selection of  $k$  objects from the set  $X$  of objects as the initial  $k$  modes — which we refer to as the *random initialization* procedure. The second one, devised in [69], is a more complicated procedure, based on the frequencies  $f(A_i = c_r[i]|X)$  of all categories, which we refer to as the *Huang initialization* procedure [69]. We focus on this second procedure, since it achieves the best results:

1. order the categories of each attribute  $A_i$  in descending order of frequency, *i.e.*,  $f(c_{r_1}[i]) \geq f(c_{r_2}[i]) \geq f(c_{r_3}[i])$ , *etc.*;
2. assign uniformly at random the most frequent categories to the initial  $k$  modes;
3. for each mode  $q_l$  obtained in the previous step, select the  $x_j \in X$  most similar to  $q_l$  and make this  $x_j$  the mode  $q_l^0 \in Q^0$ , such that  $q_l^0 \neq q_{l'}^0$  for  $l \neq l'$ .

This final step is to avoid empty clusters in  $\mathbb{C}^0$ . This initial selection procedure is aimed at having a diverse set of initial modes  $Q^0$ , which can lead to better clustering results — see more details in [69].

Our approach, *celluloid: clustering single cell sequencing data around centroids*, is our conflict dissimilarity and the Huang initialization procedure, used within the  $k$ -modes framework — its implementation available at under an MIT license [24]. It is also available on the *Python Package Index* (PyPI) [25]: installable via `pip`, *e.g.*, `pip install celluloid-clust`.



### 4.2.2 *k*-modes

Since there is package for computing a clustering based on the *k*-modes framework available on the *Python Package Index* (PyPI) [84], we decided to use this in our tests as well. The above package comes with a variety of standard dissimilarity measures, as well as the Huang (and random) initialization procedures (see above).

Here we use the above implementation with the options of matching dissimilarity (Equation 4.2) and the Huang initialization procedure, since this combination of options produced the best results. This is what we refer to as *k*-modes in our experiments.

### 4.2.3 *k*-means

Given  $m$  objects  $X = \{x_1, \dots, x_m\}$  on  $n$  real values, *i.e.*, each  $x_i \in \mathbb{R}^n$ , and an integer  $k$ , the *k*-means algorithm [97, 9] finds the vector of  $k$  values, called *means*,  $Q = \{q_1, \dots, q_k\}$  and an  $m \times k$  partition matrix minimizing the following objective function:

$$\sum_{l=1}^k \sum_{j=1}^m p[j, l] d(x_j, q_l), \quad (4.8)$$

subject to the first two constraints in Equation 4.6, while each *mean*  $q_l$  is based on a *distance measure*  $d$ , that is usually the Euclidean distance, *i.e.*,

$$d(x_j, q_l) = \sum_{i=1}^n (x_j[i] - q_l[i])^2. \quad (4.9)$$

The *k*-means algorithm starts with some initial set  $Q^0$  of  $k$  means, and an initial collection  $\mathbb{C}^0$  of  $k$  disjoint subsets of  $X$ , and then iterates the operations 1–3 as in the *k*-modes algorithm, but using the Euclidean distance (Equation 4.9) instead of dissimilarity, and computing *means* instead of modes, that minimize instead the objective function of Equation 4.8, subject to the first two constraints in Equation 4.6.

In our experiments, we used the implementation of *k*-means clustering available in scikit-learn [120].

### 4.2.4 Affinity propagation

The affinity propagation algorithm [49] uses as input a set of similarities between data points. Those similarities are clustered by choosing a point as a representative for each class. Such points gradually emerge iteratively using a message-passing procedure, where each point exchanges messages with all other points.

In particular, there exist two types of messages: (1) *responsibility*  $r(i, k)$  is sent from point  $i$  to the candidate representative point  $k$  reflecting the cumulative evidence of how well-suited  $k$  is to serve as representative of  $i$ ; and (2) *availability*  $a(i, k)$  sent from the candidate representative  $k$  to  $i$  reflecting the cumulative evidence of how well-suited  $k$  should be chosen by  $i$  to be its representative. Both variables take into account other potential candidates.

Such messages are exchanged iteratively, each time refining  $r(\cdot, \cdot)$  and  $a(\cdot, \cdot)$ , until a stop condition is fulfilled. At any point these variables can be combined to identify the representatives. For each point  $i$ , the value of  $k$  that maximizes  $r(i, k) + a(i, k)$  identifies the representative  $k$  of  $i$ , where  $k$  and  $i$  can be the same point.

In our experiments, we used the implementation of affinity propagation clustering available in scikit-learn [117].

#### 4.2.5 Hierarchical agglomerative clustering

The hierarchical agglomerative clustering method [78] produces, in an hierarchical fashion, groups of disjoint subsets of a given set, each maximizing an internal similarity score. The procedure is executed in a bottom-up fashion in which, initially, each point is in a subset by itself. At each step, two sets are merged together, so that the union maximizes a given criterion. The procedure is then repeated until only one group remains, thus having the complete hierarchical structure of the clustering.

In our experiments we used the Manhattan metric, *i.e.*, the sum of the horizontal and vertical distance between two points, to compute the similarity scores, as this metric is more suited for a matrix containing categorical data, as opposed to the Euclidean distance more suited for a continuous-valued matrix. The similarity of two sets of observations is computed using the average of the distances between elements in the respective sets. Consequently, as expected, the hierarchical agglomerative clustering, when using the Manhattan metric, performed better (see Section 4.3: Results) than the Euclidean metric, and so we only included the former in the comparison.

In our experiments, we used the implementation of agglomerative hierarchical clustering available in scikit-learn [118].

#### 4.2.6 BIRCH clustering

The BIRCH clustering procedure [140] takes as input a set of points as well as the desired number of clusters, and operates in four steps. (1) In the first step it computes the *clustering feature* (CF) tree while computing measures using a predefined metric. (2) The second optional phase tries to build a smaller CF tree while removing outliers and regrouping crowded subclusters into larger ones. (3) In the third step, a hierarchical clustering

algorithm, usually an adaptation of the agglomerative clustering (briefly described in the previous subsection), is used to cluster all the leaves of the CF tree. During this phase, the centroids of each cluster are computed. (4) The centroids are then used for the final fourth step to further refine the clusters, since minor and localized misplacements could occur during the previous step. This last step can also be used to label the points with the cluster they are placed in — we used this feature to obtain the groups of mutations in our experiments.

In our experiments, we used the implementation of BIRCH clustering available in scikit-learn [119].

#### 4.2.7 Spectral clustering

The spectral clustering algorithm [122] — see [130] for a gentle and comprehensive treatment of the topic — first performs dimensionality reduction on the data, and then clusters, using a standard clustering technique such as  $k$ -means, the data in this reduced dimension. In order to reduce the dimensionality, it first constructs a similarity *graph* from the initial matrix of similarities of the input set of data points — usually a sparse representation of this similarity matrix, *e.g.*, the  $k$ -nearest neighbor graph [130]. It then takes the Laplacian matrix [23] on this graph, and a subset of the (relevant) eigenvectors (spectrum) of this Laplacian matrix — itself a matrix, is the reduced-dimension data. It is then this matrix which is passed to the clustering technique.

In our experiments, we used the implementation of spectral clustering available in scikit-learn [121].

#### 4.2.8 Generation of simulated data

The simulated data are generated as follows. First we simulate a random tree topology on  $s$  nodes, each representing a tumor clone, by first creating a root (the germline) and then iteratively attaching the  $s - 1$  remaining nodes uniformly at random to any other node in the tree. The nodes are then randomly labeled with  $m$  mutations — meaning that each mutation is acquired at the node that it labels. Then, a total of  $n$  cells are associated to the nodes, uniformly at random. A binary matrix  $M$  is then extracted from these cells giving rise to a genotype profile for each cell (a row in  $M$ ), which is the presence (1) or absence (0) of each mutation (column in  $M$ ) in the cell, given the presence or absence of the mutations on the path in the tree from the root to this cell. The binary matrix is then perturbed according to the false negative, false positive and missing value rates, to simulate a real-case scenario. Each of the  $s$  nodes is therefore considered as a natural (true) cluster of the simulated dataset.

## 4.3 Results

To evaluate the accuracy of the clustering methods, we designed a two-fold experiment on synthetic datasets. We first measure the quality of the clusters found by the methods, and later we evaluate how such clusters impact the quality of the phylogenies returned by a cancer progression inference method.

For each experiment, we consider 100, 200 and 300 cells, respectively experiments 1, 2 and 3. For each such value we generated 50 simulated datasets (according to Section 4.2.8), where we fixed the number  $s$  of clones to 20 and the number  $m$  of mutations to 1000. While this number of mutations is at the high end in terms of currently available real cases, it will be a typical size in the near future — some such cases already existing today (see Section 4.3.3).

We performed clustering on the datasets of our experiments to obtain instances with a reduced number of columns (mutations), which can in turn be given as input to such a cancer progression inference method above. The clustering methods we used were all the ones described in the Methods section (Section 4.2), *i.e.*, *celluloid*, *k*-modes, *k*-means, affinity, agglomerative, BIRCH and spectral clustering. Note that all such clustering methods are general-purpose: given  $m$  objects on  $n$  categorical values, and an integer  $k$ , each method clusters the  $m$  objects into  $k$  groups. Since the cancer inference methods tend to scale quadratically with the number of mutations, we decided to choose a  $k$  of 100 for each method, which is a reasonable number of mutations given the currently available literature.

### 4.3.1 Evaluating a clustering

To evaluate the clusters obtained, we used standard precision and recall measures, adapted to the particular goal, as follows.

**Precision:** measures how well mutations are clustered together. For each pair of mutations appearing in the same clone in the simulated tree, we check if they are in the same cluster, resulting in a *true positive* ( $TP$ ). For each pair of mutations clustered together that are not in the same clone, we encounter a *false positive* ( $FP$ ). The value of the precision is then calculated with the standard formula:  $\frac{TP}{TP+FP}$ .

**Recall:** measures how well mutations are separated. For each pair of mutations in the same clone, we now also check if they are not in the same cluster, resulting in a *false negative* ( $FN$ ). The recall is then calculated as:  $\frac{TP}{TP+FN}$ .

It is important to highlight that we are mostly interested in obtaining a high *precision* since, while cancer phylogeny inference algorithms can later

cluster together mutations by assigning them to same subtree or the same path, they cannot separate mutations that have been erroneously clustered together. It is for this same reason that the *number* ( $k$ ) of clusters we chose is simply the largest such that the downstream inference tool can complete in a reasonable amount of time.

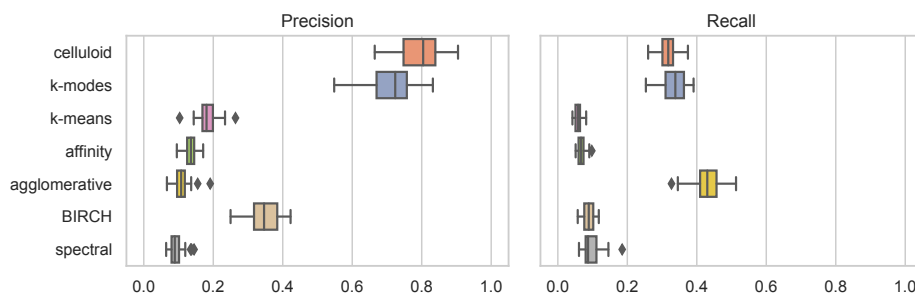


Figure 4.2: Precision and recall results for experiment 1, generated with a total of 1000 mutations, 100 cells and a clustering size of  $k = 100$ . The plots include results for *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering.

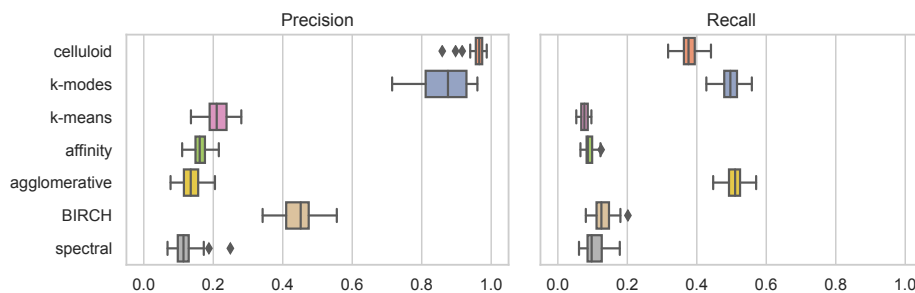


Figure 4.3: Precision and recall results for experiment 2, generated with a total of 1000 mutations, 200 cells and a clustering size of  $k = 100$ . The plots include results for *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering.

In Figures 4.2, 4.3 and 4.4 — representing the respective experiments 1, 2 and 3 — a common trend is evident: indeed, standard clustering methods (*k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering) perform much more poorly than *celluloid* and *k-modes*, presenting a gap from all the other methods in terms of both precision and recall. On the other hand, *celluloid* and *k-modes* differ slightly in terms of precision.

It is interesting to notice that the precision of the conflict dissimilarity rapidly increases when the amount of cells increase, thus being well-suited

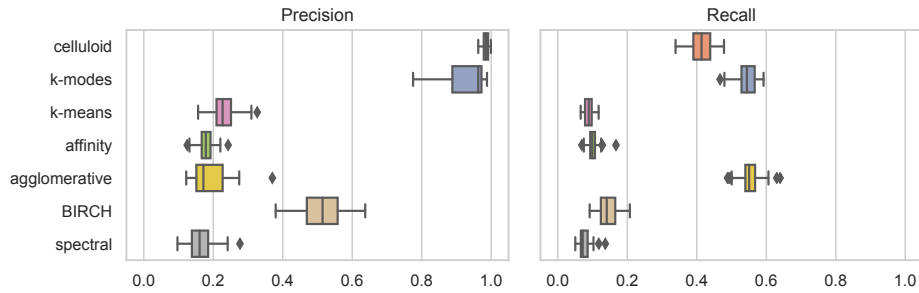


Figure 4.4: Precision and recall results for experiment 3, generated with a total of 1000 mutations, 300 cells and a clustering size of  $k = 100$ . The plots include results for *celluloid*, *k*-modes, *k*-means, affinity, agglomerative, BIRCH and spectral clustering.

for future increases on the size of SCS datasets.

We have also evaluated the quality of the clusters with several standard clustering metrics such as the adjusted Rand index [70], Fowlkes-Mallows index [48], completeness score [112] and V-measure [112]. In this case, we measure how much the computed clustering is similar to the ground truth.

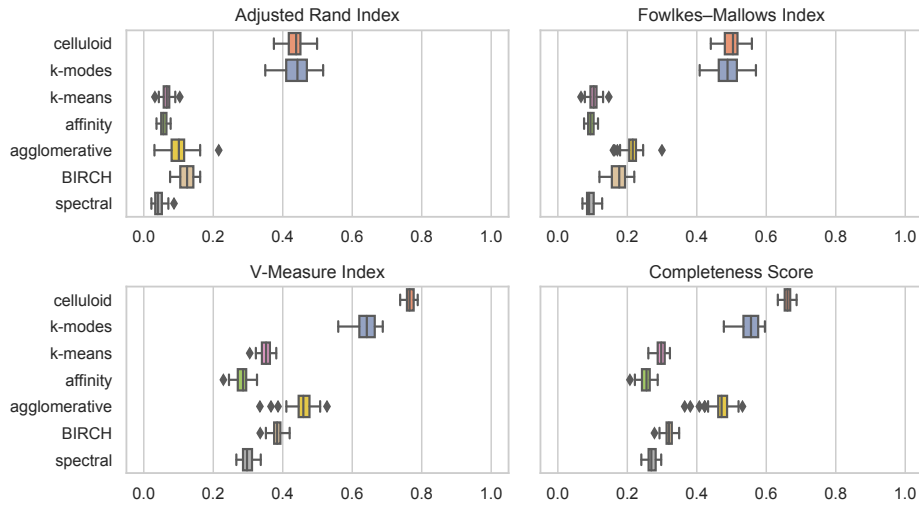


Figure 4.5: The adjusted Rand index, Fowlkes-Mallows index, completeness score and V-measure between all clustering methods and the ground truth for experiment 1, generated with a total of 1000 mutations, 100 cells and a clustering size of  $k = 100$ . The plots include results for *celluloid*, *k*-modes, *k*-means, affinity, agglomerative, BIRCH and spectral clustering.

In Figures 4.5, 4.6 and 4.7 — representing again, respectively, the results of experiments 1, 2 and 3 — we see the similar trend of *celluloid* and *k*-modes

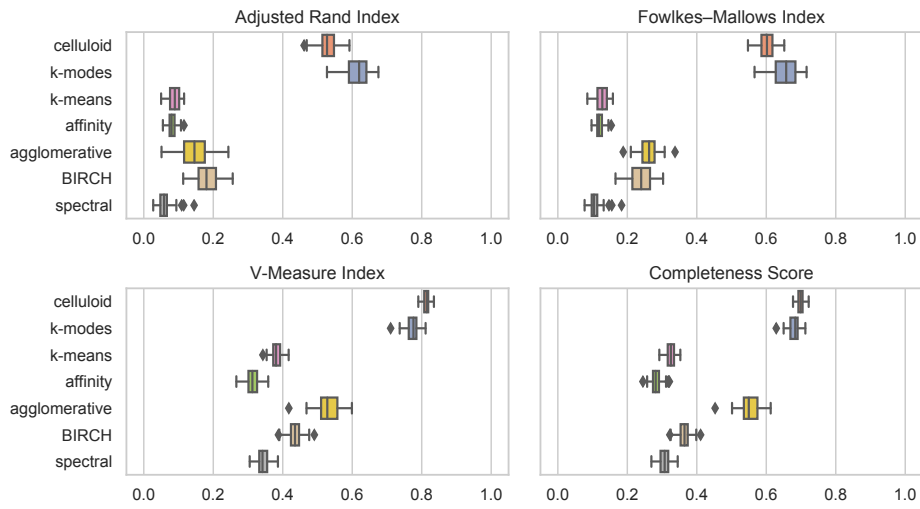


Figure 4.6: The adjusted Rand index, Fowlkes-Mallows index, completeness score and V-measure between all clustering methods and the ground truth for experiment 2, generated with a total of 1000 mutations, 200 cells and a clustering size of  $k = 100$ . The plots include results for *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering.

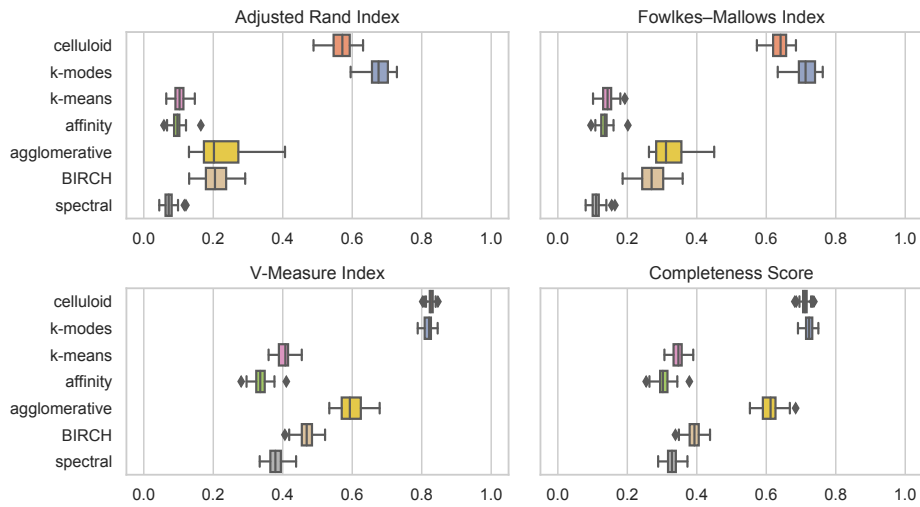


Figure 4.7: The adjusted Rand index, Fowlkes-Mallows index, completeness score and V-measure between all clustering methods and the ground truth for experiment 3, generated with a total of 1000 mutations, 300 cells and a clustering size of  $k = 100$ . The plots include results for *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering.

performing much better than all the other methods, presenting this same gap from all the other methods, while differing slightly from each other.

We decided also to compute the adjusted Rand index, Fowlkes-Mallows index, completeness score and V-measure for *all pairs* of techniques to allow the observation of the similarity, according to the measures, between different algorithms. The heatmaps in figures 4.8, 4.9 and 4.10 — representing, respectively, experiments 1, 2 and 3 — show the average value of the scores of each simulated dataset for each pair of clustering method.

The first column of all the heatmaps represents the comparison with the ground truth, where in all the cases *celluloid* achieves larger values, thus more resembling the ground truth. Furthermore, *celluloid* and *k*-modes are very similar to each other, while the other methods tend to be quite dissimilar to each other.

### 4.3.2 Assessing the impact of a clustering

To better understand the impact of the clustering on the actual cancer progression inference, we considered SCITE [75], SASC [30] and SPhyR [43], three published, publicly available inference tools tailored to SCS data, and have been shown to scale to instances of the size we consider in our study — the latter tool does clustering using *k*-means as part of the inference. We executed SCITE [75] and SPhyR [43] on both the obtained clusters and on the unclustered datasets, to understand the effect of the clustering on the tools. SASC was not able to complete in a reasonable amount of time (with a cut-off of  $< 2$  hours) on the unclustered data, due to the higher complexity of the search space of the solutions it generates. We performed the inference with all of the clustering methods used as a preprocessing step. Note that all methods were run with default parameters, with the exception of SASC and SPhyR being parameterized to output trees with no losses, in order to be able to compare with SCITE, which does not model losses, *i.e.*, rather, it adheres to the infinite sites assumption.

For assessing the accuracy of the methods we used the measures defined in [28, 30]:

**Ancestor-descendant accuracy:** for each pair of mutations in an ancestor-descendant relationship in the ground truth, we check whether the relationship is conserved in the inferred tree (*TP*) or whether it is not (*FN*). For each pair of mutations in an ancestor-descendant relationship in the inferred tree, we also check if such relationship does not exist in the ground truth tree (*FP*).

**Different lineages accuracy:** similarly to the previous measure, we check whether mutations in different branches are correctly inferred or if any pair of mutation is erroneously inferred in different branches.



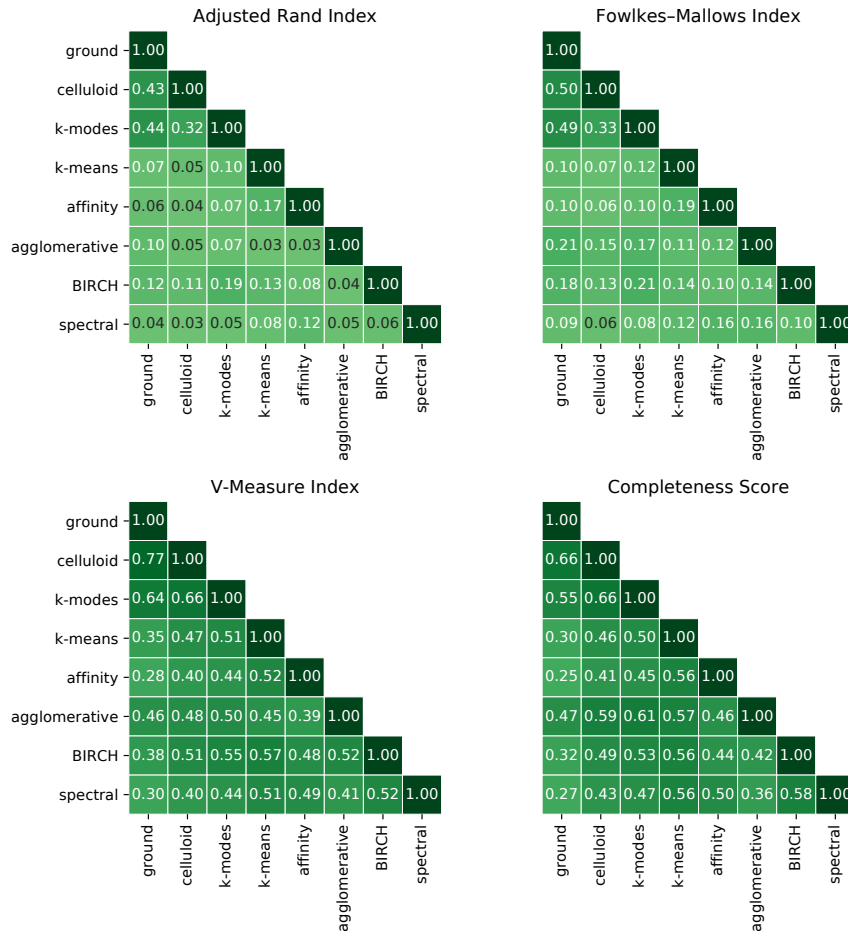


Figure 4.8: The adjusted Rand index, Fowlkes-Mallows index, completeness score and V-measure between all pairs of clustering methods for experiment 1, generated with a total of 1000 mutations, 100 cells and a clustering size of  $k = 100$ . The plots include results for *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering. Each cell is the average of the scores obtained for each simulated instance.

Figures 4.11, 4.12 and 4.13 show very interesting results. Regarding the results of SCITE and SASC, as expected, we observe a severe drop in performance when used in combination with *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering methods. This fact is supported by the gap in the precision of the methods — a low precision indeed leads to a low accuracy in the tree reconstruction. The trend is still present in the different lineages accuracy, but to a lesser extent — this is because, as previously discussed, a cancer inference method can separate *clusters* of mutations, but when a cluster is computed it is not possible to separate mutations *within*

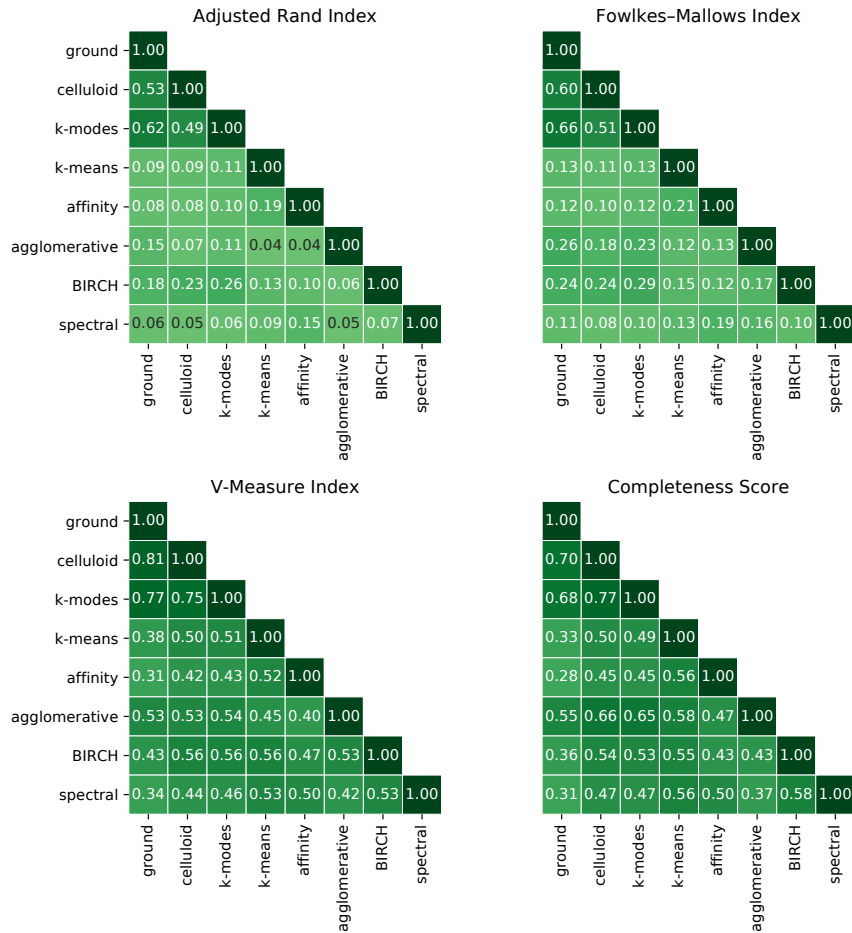


Figure 4.9: The adjusted Rand index, Fowlkes-Mallows index, completeness score and V-measure between all pairs of clustering methods for experiment 2, generated with a total of 1000 mutations, 200 cells and a clustering size of  $k = 100$ . The plots include results for *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering. Each cell is the average of the scores obtained for each simulated instance.

this cluster.

On the other hand, the results obtained by SCITE when *celluloid* and *k-modes* are used are much better, in particular, *celluloid* as a preprocessing step — which leverages our conflict dissimilarity — allows SCITE to score higher in both ancestor-descendant and different lineages accuracies than it is able to using unclustered data as input. For this inference tool, using *celluloid* as a preprocessing step actually helps SCITE to achieve a better score than without it — experiment 3, being the largest of the three, shows the biggest improvements. SCITE on unclustered data scores an average of

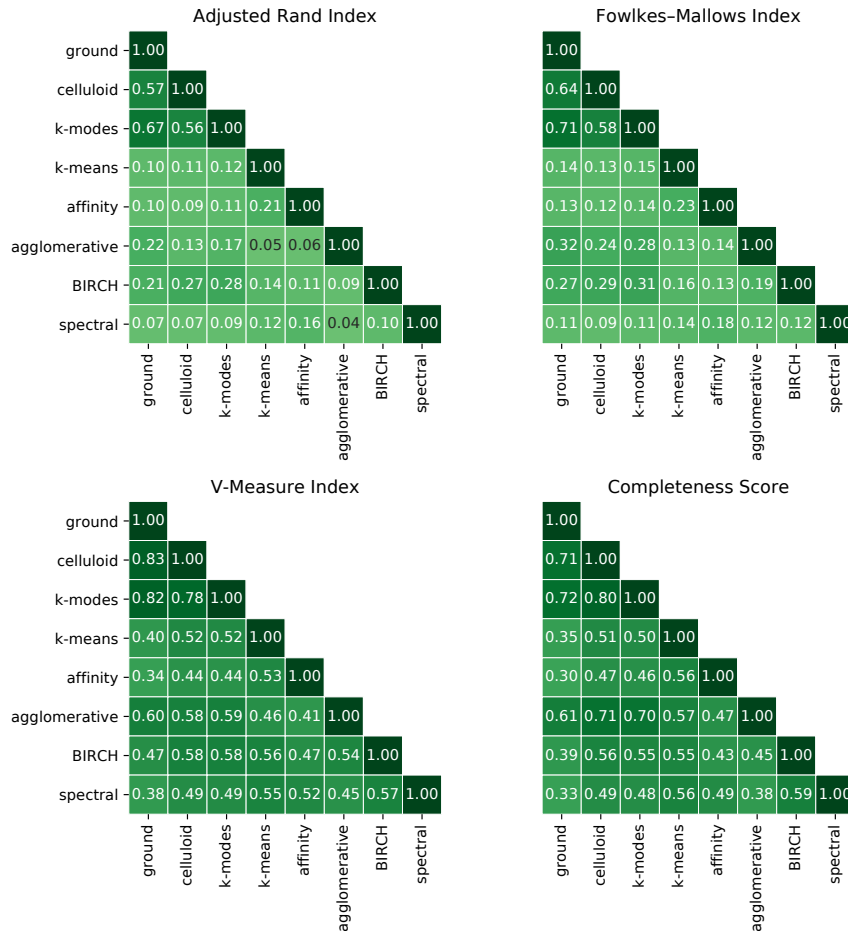


Figure 4.10: The adjusted Rand index, Fowlkes-Mallows index, completeness score and V-measure between all pairs of clustering methods for experiment 3, generated with a total of 1000 mutations, 300 cells and a clustering size of  $k = 100$ . The plots include results for *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering. Each cell is the average of the scores obtained for each simulated instance.

0.7551 and 0.9537 for ancestor-descendant and different lineages respectively, against an average of 0.9358 and 0.9907 after clustering the datasets using *celluloid*. Moreover, *celluloid* allows SCITE to achieve a 20x speedup in runtime, on average.

The results obtained by SASC, shown in Figures 4.11, 4.12 and 4.13, are very similar to what is computed by SCITE — in particular, *celluloid* provides much better results than any other clustering method. Once again, on the larger experiment, the gap between the clustering methods is seen the most — in particular, SASC scores an average of 0.9365 and 0.9909

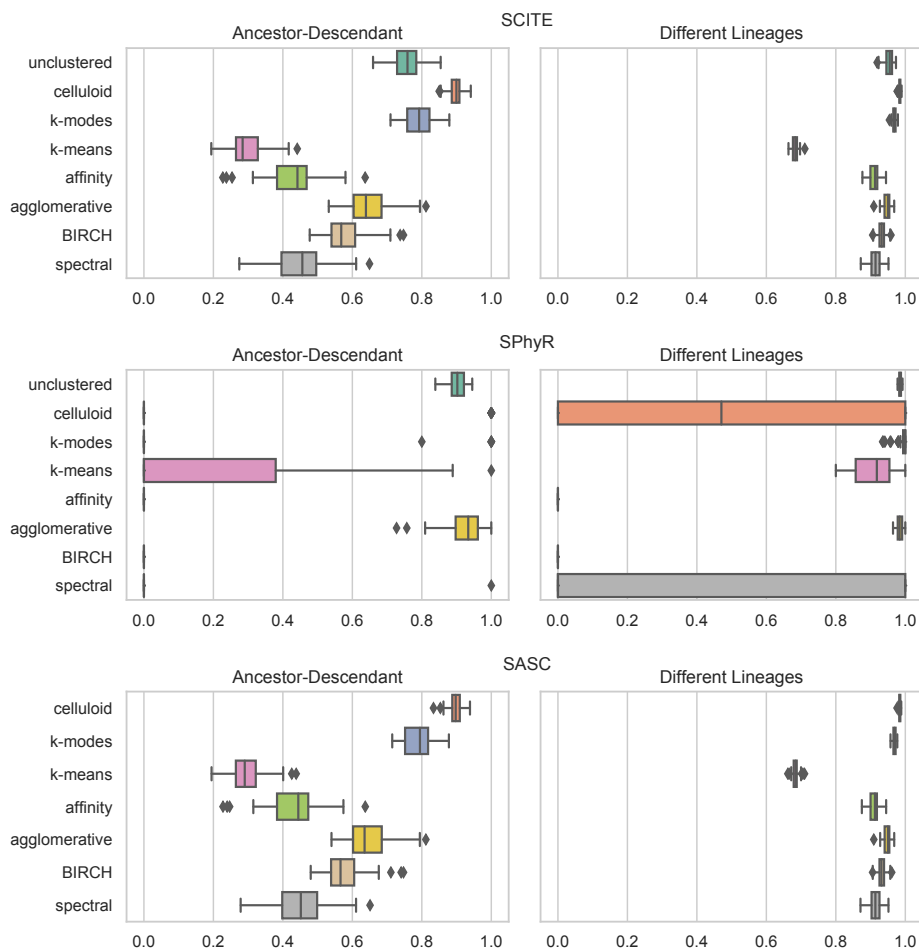


Figure 4.11: Ancestor-descendant and different lineages accuracy measures for experiment 1. Cancer phylogenies are inferred by SCITE (Top), SPhyR (Middle) and SASC (Bottom) using as input both the unclustered data as well as the clusters obtained by *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering.

for ancestor-descendant and different lineages respectively, when *celluloid* is used as a preprocessing step. These results are particularly interesting since SASC is unable to complete in a reasonable amount of time (with a cut-off of  $< 2$  hours) on the unclustered data, hence the reason for this absence from the experiments.

Unlike the previous tools, SPhyR shows very interesting results since the method itself performs a *k-means* clustering on both mutations and cells given as input. For this reason, clustering already clustered data can be a hit-or-miss case, as seen in Figures 4.11, 4.12 and 4.13. In almost all cases, preprocessing the data causes SPhyR to obtain extremely low values

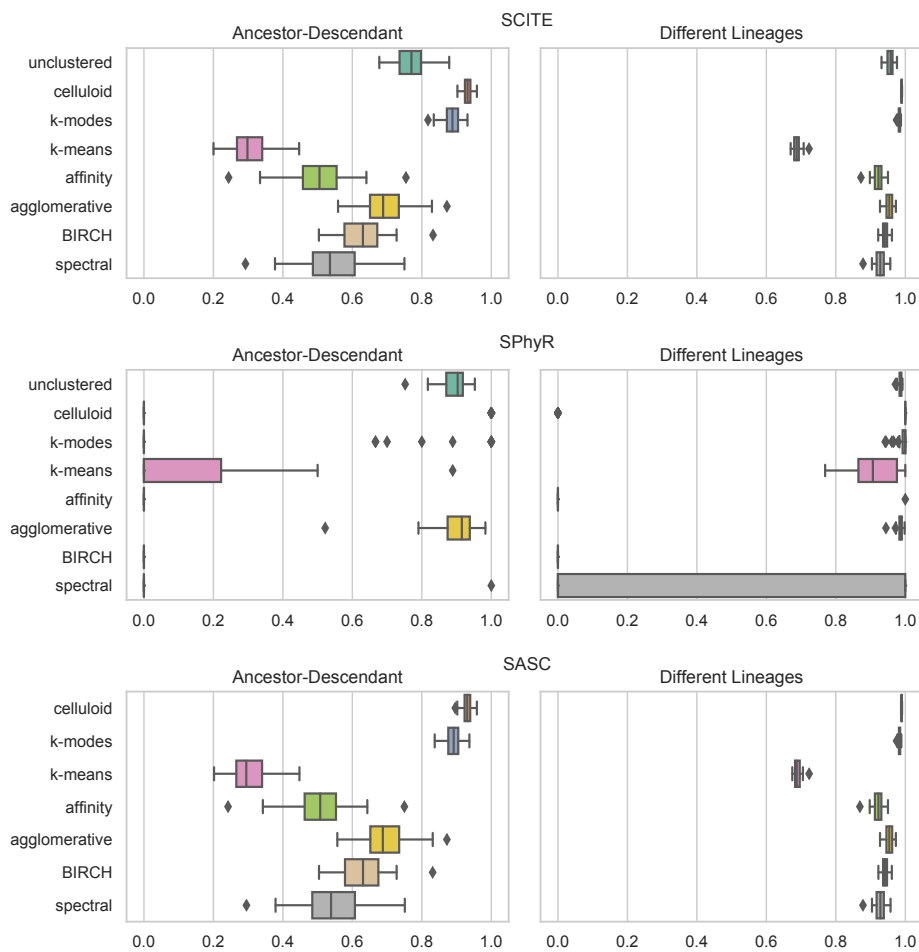


Figure 4.12: Ancestor-descendant and different lineages accuracy measures for experiment 2. Cancer phylogenies are inferred by SCITE (Top), SPhyR (Middle) and SASC (Bottom) using as input both the unclustered data as well as the clusters obtained by *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering.

in the ancestor-descendant measure, while achieving a very good different lineages score, with the exception of *spectral* clustering in all experiments and *celluloid* in experiment 1. On the other hand, the *agglomerative* clustering seems to have no impact or to slightly improve the tool — this could be a consequence of the high recall and low precision achieved by the clustering algorithm — indeed it is possible that separating well clusters but not merging them very much leads the clustering step of SPhyR to achieve a better result.

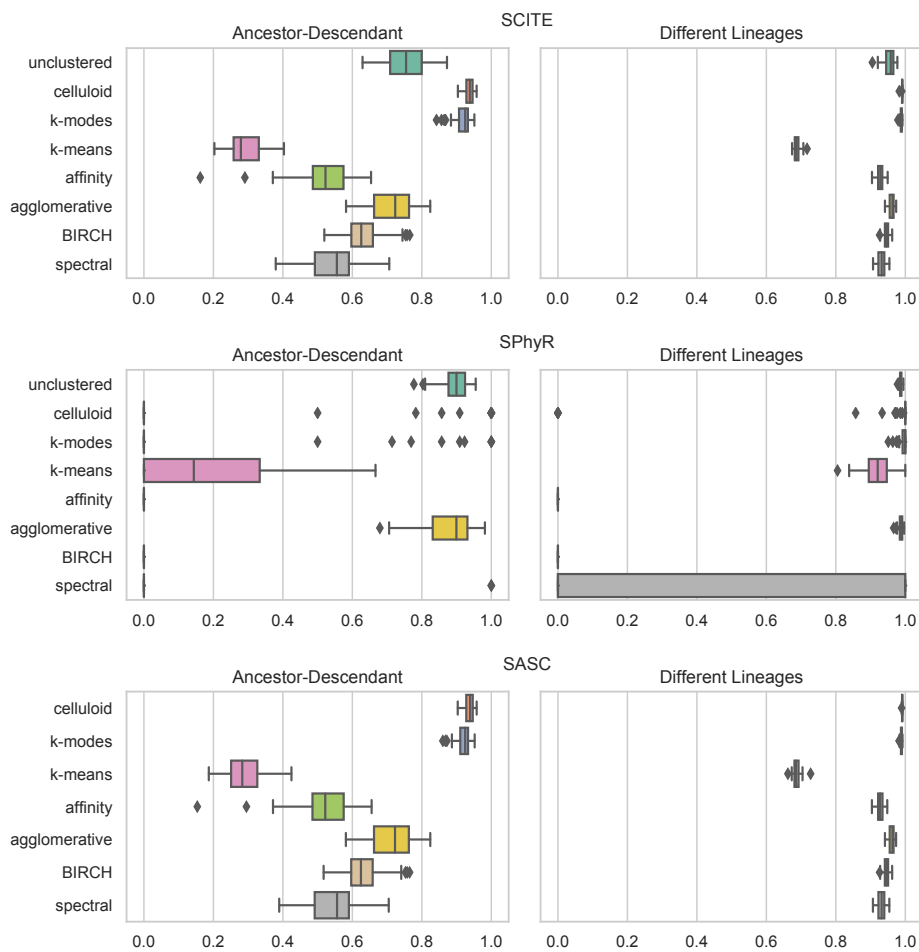


Figure 4.13: Ancestor-descendant and different lineages accuracy measures for experiment 3. Cancer phylogenies are inferred by SCITE (Top), SPhyR (Middle) and SASC (Bottom) using as input both the unclustered data as well as the clusters obtained by *celluloid*, *k-modes*, *k-means*, *affinity*, *agglomerative*, *BIRCH* and *spectral* clustering.

### 4.3.3 Application on real data

Finally, we run the entire pipeline (clustering + inference method) on an oligodendroglioma IDH-mutated tumor [129] consisting of 1842 mutations over 926 cells, to assess if the improvement in the runtimes that we have seen on simulated data carry over to a real dataset.

Such computation was performed using SCITE, which on unclustered data, takes as long as 68 hours, while adding a preprocessing step with *celluloid*, we were able to compute the tree within 1 hour, therefore decreasing the time needed by a factor of 70. Moreover, it is most likely that pre-

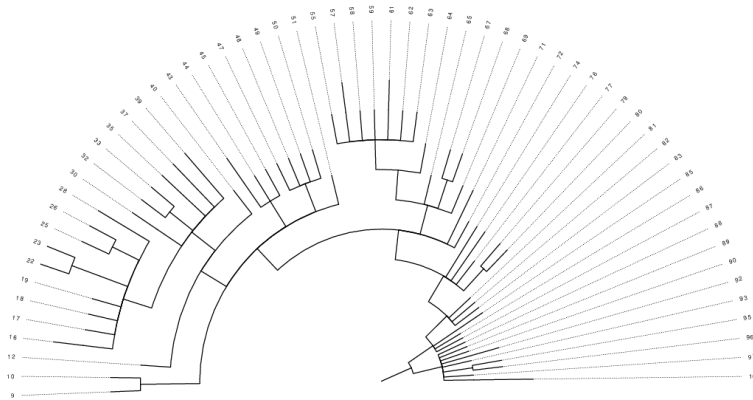


Figure 4.14: Tree computed on the oligodendroglioma IDH-mutated tumor from [129]. The labels on the nodes represent the 100 clusters obtained by *celluloid*. The tree was inferred using the cancer inference tool *SASC*.

processing the data with *celluloid* provides a better phylogeny than SCITE alone, as seen in the experiments on simulated data.

Furthermore we computed the same dataset using *SASC*, Figure 4.14 shows the tree inferred, where each node is one of 100 clusters computed by *celluloid*.

## 4.4 Discussion

Motivated by the advancements in single cell sequencing (SCS) technologies such as decreasing costs and improved quality which will result in larger SCS instances, we proposed a method *celluloid* to reduce the size of such instances, via categorical clustering based on the  $k$ -modes framework, and a novel *conflict dissimilarity* — tailored to properties specific to SCS instances. The idea is that this can allow tools which infer cancer phylogenies purely from SCS data, such as SCITE [75], SPhyR [43] or *SASC* [30] — which work well on the instances of today — to scale to the size of SCS instances of the near future. We hence devised *celluloid* for clustering purely single-cell data, and focused on comparing other methods with this same goal in mind. Notwithstanding, there are important *hybrid* methods for inferring cancer phylogenies from a combination of bulk sequencing and SCS data [92, 114], which even perform clustering as part of the phylogeny inference (like with the purely SCS tool, SPhyR).

We have shown how to compare various clustering methods on single cell sequencing data, with three distinct experiments. More precisely, we describe two experiments on synthetic data: the first experiment measures

the quality of the clusters computed by the methods, while the second experiment considers the effect of the clustering on the quality of trees obtained downstream from a phylogeny inference tool. Finally, we have described an experiment on real data that measures the usefulness of the clustering procedure, by computing the improvement in the runtime for a large instance.

We have made available the entire pipeline [24] that runs the clustering tools on those data, and computes the plots and tables used in the cluster analysis. Our comparison is reproducible and can be easily extended by modifying a Snakemake [86] file.

One of the main conclusions that we can draw from our comparison is that  $k$ -means is not an adequate choice for clustering single cell data for inferring tumor phylogenies. The second main finding of our paper is that a suitable clustering step, such as *celluloid*, not only decreases the runtime of the phylogeny inference methods, but can also *improve* the quality of the inferred phylogenies, as we have shown for SCITE and SASC.

Future work includes a more complete and comprehensive study of clustering methods and downstream cancer phylogeny inference methods, but also of simulated and real datasets.

Since the  $k$ -modes framework saw a boost in performance when coupled with our novel conflict dissimilarity, an interesting future work is to determine if there are other dissimilarities, or even other categorical clustering frameworks which perform even better. For example, the conflict dissimilarity takes into account the high drop-out rate in SCS data; maybe there are refinements that could account for other aspects of SCS data such as heterogeneous false-negative rate [30]. More generally, maybe there are other dimensionality reduction techniques, or ways to select or extract a compact set of representative features of this dataset (not necessarily mutations or cells), which could in turn be passed to some downstream phylogeny inference.

Finally, while we focused on single cell data in this study, a more long-term research direction is to develop tools — which use some combination of clustering + phylogeny inference — for inferring cancer phylogenies using multiple sources of information together (*e.g.*, transcriptomics data, health informatics data, phylogenetic information of other closely-related cancers, *etc.*).



## Chapter 5

# Comparing tumor phylogenies

The latest advances in cancer sequencing, and the availability of a wide range of methods to infer the evolutionary history of tumors, have made it important to evaluate, reconcile and cluster different tumor phylogenies. Recently, several notions of distance or similarities have been proposed in the literature, but none of them has emerged as the golden standard. Moreover, none of the known similarity measures is able to manage mutations occurring multiple times in the tree, a circumstance often occurring in real cases.

To overcome these limitations, in this paper we propose **MP3**, the first similarity measure for tumor phylogenies able to effectively manage cases where multiple mutations can occur at the same time and mutations can occur multiple times. Moreover, a comparison of **MP3** with other measures shows that it is able to classify correctly similar and dissimilar trees, both on simulated and on real data.

### 5.1 Introduction

While there is a wide range of measures to compare leaf-labeled trees in the literature, ad-hoc methods for tumor phylogenies are starting to appear in the last few years [38, 79, 54, 10, 11]; in particular, a detailed study of some notions of distance [38] has introduced two new measures complementing some more established definitions used in various cancer inference studies [30, 28]. Those new measures are more nuanced, in order to capture some aspects of the mutation inheritance process, while still being very efficient to compute. A common trait of all the latter distances is their reliance on the analysis of *pairs* of nodes.

On the other hand, some of the most widely used distances on classical phylogenies are based on rooted triples [19, 39, 3] (for rooted phylogenies) or quartets [41] (for unrooted phylogenies) of labeled leaves. Although such

metrics have major limitations for our purposes, as they do not apply directly to fully-labeled trees, they also have some desirable properties that we would like to transfer in our setting. Specifically, this kind of metric captures well the differences in the topology of the trees; a feature that, to the best of our knowledge, lacks in most of the existing methods for tumor phylogenies. Therefore we expect a triplet-based measures to provide additional insights on the different evolutionary histories, when applied to cancer progression.

In this paper, we generalize the notion of rooted triples similarity for classical phylogenies to tumor phylogenies. Moreover, we further extend this to multi-labeled trees (that is, where each node is labeled by a set of labels) and poly-occurring labels (that is, each label can be assigned to more than one node). The latter feature is needed since recent studies [87, 20] suggest widespread recurrence and loss of mutations, and more and more methods designed to infer tumor phylogenies considering such a possibility are starting to appear [43, 30, 28]. In a phylogenetic tree a mutation loss is represented by a special character in the label, such as a minus sign: the design of our measure allows to handle such evolutionary events effectively, as they uniquely correspond to their label like any other kind of mutation.

Through an extensive experimental analysis, we show that our novel measure is able to overcome the limitations in the existing literature and to provide a better alternative to both the direct comparison of evolutionary histories and the application to established clustering techniques, following the approach of [38]. Such a performing measure can also be incorporated in recent works [2, 53] designed to cluster and build consensus across multiple cancer progressions.

## 5.2 Methods

A classical phylogenetic tree is a rooted, unordered, leaf-labeled tree. The set of all the labels occurring in  $T$  is denoted by  $\lambda(T)$ , and a function  $\mathbf{N}(\cdot)$  maps each element of  $\lambda(T)$  to a leaf of  $T$ . We denote with  $\text{LCA}(u, v)$  the Lowest Common Ancestor of nodes  $u$  and  $v$ . Given three leaves  $u, v, z \in V_T$ , the *minimal tree topology* they induce on  $T$ , denoted as  $\text{MTT}_T(u, v, z)$ , is the smallest subtree of  $T$  that includes the nodes  $V_T^{u,v,z} = \{u, v, z\} \cup \text{LCA}(u, v) \cup \text{LCA}(v, z) \cup \text{LCA}(u, z)$ , and where all the nodes with degree 2 not in  $V_T^{u,v,z}$  are contracted.

The rooted triplet distance measures the dissimilarity between two leaf-labeled trees with identical labels. It is given by the number of rooted triplets that induce different minimal topologies (Figure 5.1) in the two trees over the total number of triplets [76]. As tumor progression trees are fully-labeled, such metric cannot be directly applied: in this section we propose a novel similarity measure, inspired by the triplet distance, specifically designed for these more general trees.

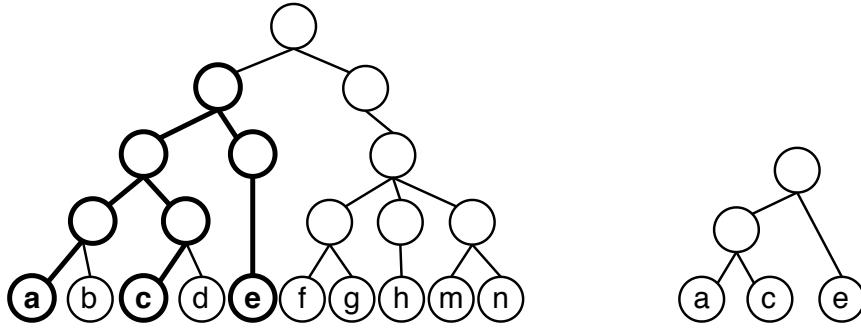


Figure 5.1: Rooted triplet on labels  $(a, c, e)$ . (Left) Tree  $T$  where the smallest subtree that contains all three labels is highlighted. (Right) The minimal topology induced by  $(a, c, e)$ .

### 5.2.1 Extension to fully labeled trees and multi-labeled trees

A tree  $T$  on a set  $V_T$  of  $n$  nodes is *fully-labeled* by a set  $\lambda(T)$  of labels if there is a bijection  $\mathbb{N} : \lambda(T) \rightarrow V_T$ . The definition of minimal topology of three leaves can be trivially extended to the minimal topology of three nodes: we next show that there are only five possible configurations (see Figure 5.2).

**Lemma 6.** *Given nodes  $u, v, z \in V_T$ , there exist only five possible configurations for  $\text{MTT}_T(u, v, z)$ .*

*Proof.* We start by dividing two possible cases: (i)  $\text{LCA}(u, v) = \text{LCA}(v, z) = \text{LCA}(u, z)$ , or (ii) just two LCAs are the same, say  $\text{LCA}(v, z) = \text{LCA}(u, z)$ . There are no other possibilities, as  $\text{LCA}(u, v) \neq \text{LCA}(v, z) \neq \text{LCA}(u, z)$  is impossible: indeed, suppose without loss of generality that  $\text{LCA}(u, v)$  is a descendant of  $\text{LCA}(u, z)$ ,  $\text{LCA}(u, v) \neq \text{LCA}(u, z)$ : they cannot be unrelated, as by definition they are both ancestors of  $u$ .  $\text{LCA}(u, z)$  is thus a common ancestor for  $v$  and  $z$ . Suppose towards a contradiction that  $\text{LCA}(v, z) \neq \text{LCA}(u, z)$ , thus it is a descendant of  $\text{LCA}(u, z)$  and an ancestor of  $\text{LCA}(u, v)$ . But then it is an ancestor of both  $u$  and  $z$  and it is lower than  $\text{LCA}(u, z)$ , a contradiction.

Case (i) has two subcases: either  $\text{LCA}(u, v) \in \{u, v, z\}$ , corresponding

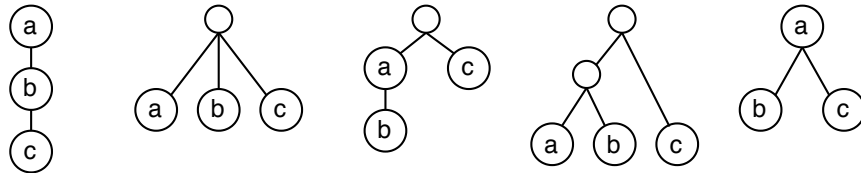


Figure 5.2: The five possible configurations for the minimal tree topology induced by an unordered set of three labels.

to the rightmost configuration in Figure 5.2, or  $\text{LCA}(u, v) \notin \{u, v, z\}$ , corresponding to the second configuration from the left.

Case (ii) has three subcases: either both the distinct LCAs are in  $\{u, v, z\}$ , or none of the two is, or finally one is in  $\{u, v, z\}$  and the other is not. The first subcase corresponds to the leftmost configuration in Figure 5.2, the second subcase to the fourth configuration from the left. For the third subcase, either the external LCA is an ancestor of all of the three  $\{u, v, z\}$ , corresponding to the third configuration, or it is an ancestor of two nodes and a descendant of the third one, say  $u$ . In the latter case, though, the external node would be the only child of  $u$ , and thus would be contracted by definition of  $\text{MTT}_T(u, v, z)$ , leading again to the rightmost configuration of Figure 5.2.  $\square$

In the case of fully-labeled trees, the definition of LCA of two nodes and MTT of three nodes can trivially be extended to the LCA of two labels and the MTT of three labels, as there is a one-to-one correspondence between nodes and labels. From now on, for ease of presentation, given two nodes  $u$  and  $v$  and their respective labels  $a$  and  $b$ , we will use  $\text{LCA}(u, v)$  or  $\text{LCA}(a, b)$  interchangeably. When modeling tumor progression, though, to have a bijection between nodes and labels (i.e., mutations) is quite a strong assumption, as multiple mutations often appear at the same time in the evolutionary history of cancer. We thus relax our assumptions and consider *multi-labeled* instead of fully-labeled trees.

A rooted, unordered tree  $T$  is multi-labeled if there exists a surjective function  $\mathbb{N} : \lambda(T) \rightarrow V_T$  that labels each node of  $T$  with a set of labels from  $\lambda(T)$ : note that, in this model, each label is assigned to one and only one node of  $T$ . We extend the definition of lowest common ancestor of two labels for a multi-labeled tree as follows: if  $a \in \lambda(T)$  and  $b \in \lambda(T)$  label the same node  $u$ , then  $\text{LCA}(a, b) = u$ ; if they label two distinct nodes  $u, v$ , then  $\text{LCA}(a, b) = \text{LCA}(u, v)$ . This allows us to straightforwardly extend the definition of minimal tree topology of three labels for multi-labeled trees. There are only four possible additional configurations for the minimal tree topology of multi-labeled trees, shown in Figure 5.3.

**Lemma 7.** *Given  $T$  multi-labeled and  $a, b, c \in \lambda(T)$ , there exist nine configurations for  $\text{MTT}_T(a, b, c)$ .*

*Proof.* Besides the five possible configurations already listed in the proof of Lemma 1, the multi-labeled model admits four additional configurations, due to the extension of the definition of LCA of two labels. In the first three additional cases, two labels are assigned to the same node and the third one to a different one. Without loss of generality, suppose that  $a$  and  $b$  label the same node. These cases are: (i)  $\text{LCA}(a, b) = \text{LCA}(b, c) = \text{LCA}(a, c)$  and the LCA is a node in  $\{a, b, c\}$ , (ii)  $\text{LCA}(a, b) \neq \text{LCA}(b, c) = \text{LCA}(a, c)$  and both LCA are nodes in  $\{a, b, c\}$ , and (iii)  $\text{LCA}(a, b) \neq \text{LCA}(b, c) = \text{LCA}(a, c)$  and only

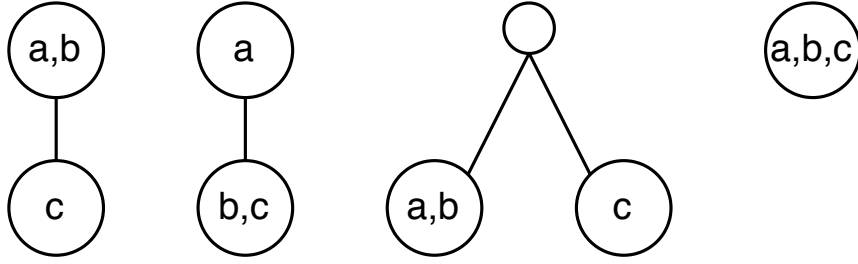


Figure 5.3: The four additional possible configurations for the minimal tree topology of multi-labeled trees induced by an unordered set three labels.

$\text{LCA}(a, b)$  is a node in  $\{a, b, c\}$ . The remaining case (iv) is the simplest one in which  $a, b, c$  label the same node of  $T$ , implying that  $\text{LCA}(a, b) = \text{LCA}(b, c) = \text{LCA}(a, c)$ .  $\square$

### 5.2.2 Extension to poly-occurring labels

We further extend our model of tumor phylogeny by allowing the same label of  $\lambda(T)$  to be assigned to multiple nodes of  $T$ . An element of  $\lambda(T)$  that labels more than one node of  $T$  is said to be a *poly-occurring* label. To the best of our knowledge, none of the existing tools is able to handle poly-occurring labels: indeed, although some of them accept input trees with poly-occurring labels, they simply disregard the multiple occurrences of a same label.

Since it is often the case where the inferred evolutionary history involves the appearance of the same mutation in multiple events, a meaningful comparison between tumor phylogenies cannot overlook such a phenomenon. To consider poly-occurring labels in our similarity measure, we extend the definition of minimal tree topology. First, note that if a label occur multiple times in the tree, then  $\mathbb{N}$  maps each label to one or more nodes in  $V_T$ . Then, we define the minimal tree topology of poly-occurring labels  $a, b, c$ , denoted by  $\mathbb{M}$ , as follows, where  $\sqcup$  indicates the multiset union:

$$\mathbb{M}_T(a, b, c) = \bigsqcup_{u \in \mathbb{N}(a), v \in \mathbb{N}(b), z \in \mathbb{N}(c)} \text{MTT}_T(u, v, z)$$

In other words, the minimal tree topology of three labels is the multiset of all the minimal tree topologies of the nodes where  $a, b$ , and  $c$  appear. We remark that in this setting  $\mathbb{M}_T$  is a multiset of configurations, thus the same configuration may appear multiple times in  $\mathbb{M}_T$ .

### 5.2.3 Similarity measure between trees

We are now able to define a similarity measure between fully-labeled trees with poly-occurring labels. Let  $S$  be a multiset and let  $|S|$  be its cardinality.

We define the number of shared configurations of labels  $a, b, c$  between two trees  $T_1$  and  $T_2$  as  $N(a, b, c) = |\mathbb{M}_{T_1}(a, b, c) \cap \mathbb{M}_{T_2}(a, b, c)|$ , *i.e.* the cardinality of the multiset intersection, and the maximum number of configurations of the triplet in the trees as  $D(a, b, c) = \max\{|\mathbb{M}_{T_1}(a, b, c)|, |\mathbb{M}_{T_2}(a, b, c)|\}$ .

Based on these two values we define multiple variations of the Multi Poly-occurring labels triplet-based (MP3) similarity measure that we will later combine into a single score. We define  $\text{MP3}_\cap$  as the similarity computed between triplets of labels shared by the two trees:

$$\text{MP3}_\cap = \frac{\sum_{(a,b,c) \in I} N(a, b, c)}{\sum_{(a,b,c) \in I} D(a, b, c)} \quad (5.1)$$

where  $I$  is the set of triples in  $\lambda(T_1) \cap \lambda(T_2)$ . Due to the nature of only considering the subset of labels that appears in both trees,  $\text{MP3}_\cap$  is a conservative measure, therefore we present a variation that consider all possible configurations in both trees, thus having a wider view:

$$\text{MP3}_\cup = \frac{\sum_{(a,b,c) \in J} N(a, b, c)}{\sum_{(a,b,c) \in J} D(a, b, c)} \quad (5.2)$$

where  $J$  is the set of triples in  $\lambda(T_1) \cup \lambda(T_2)$ . Differently from  $\text{MP3}_\cap$ ,  $\text{MP3}_\cup$  weighs also the the labels that appear only in one of the trees. Note that, for every pair of trees,  $\text{MP3}_\cup \leq \text{MP3}_\cap$ , as the numerator remains identical in both, while the denominator of  $\text{MP3}_\cup$  has all the elements in  $\text{MP3}_\cap$  with the addition of the values of  $D$  for the triples present only in one of the input trees.

Although  $\text{MP3}_\cap$  and  $\text{MP3}_\cup$  are closely related, they provide two different views of a tumor phylogeny. Indeed, on one hand  $\text{MP3}_\cap$  measures how similar the shared history of two tumor phylogenies is, *i.e.* it provides an idea of how well the two progressions can be reduced to the same subsequence of common mutations. On the other hand,  $\text{MP3}_\cup$  measures how similar the whole histories of the two evolutions are, *i.e.* it considers the impact of mutations acquired (or lost) in only one progression.

Since the previous measures capture different aspects of the progressions, we want to combine them into a single, usable and powerful similarity measure that couples the strengths of both. The most intuitive method is to simply use a mean. We opted for the geometric mean:  $\text{MP3}_G = \sqrt{\text{MP3}_\cap \cdot \text{MP3}_\cup}$ .

This function is not completely satisfactory, as a uniform function of 5.1 and 5.2 is not able to comprehensively capture the nuances in the input trees. Therefore we developed a weighted mean with an intentional bias towards  $\text{MP3}_\cap$  to catch inner similarities in different trees. Such combination then tends to be closer to  $\text{MP3}_\cap$  when the trees are similar while moving

towards  $\text{MP3}_U$  as the trees are less similar:

$$\text{MP3}_\sigma = \text{MP3}_U + \sigma(\text{MP3}_\cap) \cdot \min\{\text{MP3}_\cap - \text{MP3}_U, \text{MP3}_U\},$$

where  $\sigma(x) = (1 + e^{-\mu(x-\frac{1}{2})})^{-1}$  is the classic sigmoid function centered in  $1/2$  and  $\mu$  is used to adjust the slopeness of the curve; we set  $\mu = 10$  in our experimentation. In addition, the sigmoid polarizes the values close to  $1/2$ , thus helping to decide whether they are closer to 1 or 0, therefore moving the final score closer to  $\text{MP3}_\cap$  or  $\text{MP3}_U$ .

We show here an experimental comparison of the different versions of  $\text{MP3}$ , demonstrating the reason we decided to use  $\text{MP3}_\sigma$  as our default measure. We show in Figure 5.4 (a) that  $\text{MP3}_\sigma$  combines the best aspect of  $\text{MP3}_\cap$  and  $\text{MP3}_U$  while  $\text{MP3}_G$  is, as expected, the average of the two. The same result can be seen in Figure 5.4 (b), while Figure 5.4 (c) display the effect of the sigmoid, where as the trees become less similar the value move towards the union.

While all four measures are available in our implementation, we decided to use  $\text{MP3}_\sigma$  as default measure and is denoted simply as  $\text{MP3}$ .

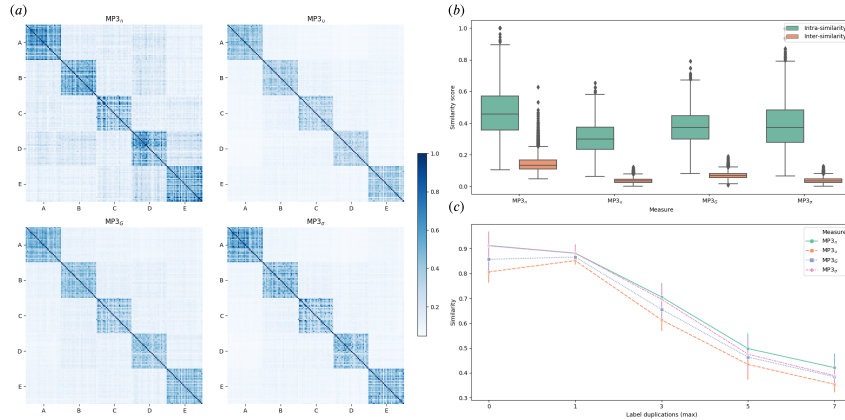


Figure 5.4: (a) Heatmaps displaying the scores between all the 150 simulate trees from the second experimental setting. (b) Distribution of the similarities between the trees in the same class (Intra-similarity) and in different classes (Inter-similarity) for the 5 classes. (c) Effect of label duplication on the similarity scores. Similarities are the average of 15 trees generate from the same base with the specified value of maximum duplication from the previous experiment.

## 5.3 Results

### 5.3.1 Simulated Data

To perform our experiments we follow an approach similar to the one performed in [38]. We start from a base tree on which we apply a series of perturbations selected from: label swapping, label removal, label duplication, node swapping and node removal. Both the perturbations and the nodes and labels on which they are applied are chosen at random: our procedure allows to select a user-specified total number of actions and a probability vector that will be used to select the perturbations from the previous list.

For the measure comparison experiments, we generated 30 perturbations from each of the 5 base trees, for a total of 150 trees. For the clustering evaluation, 3 base trees are entirely different from each other, and another 2 are perturbations of two of the others, to simulate similar sub-families of the same tumor type: we perform a total of 10 perturbations on such 5 trees. More details on the perturbation parameters will be described in each section, while the entire configuration is available and reproducible at [https://github.com/AlgoLab/mp3treesim\\_supp](https://github.com/AlgoLab/mp3treesim_supp).

### 5.3.2 Measures comparison

We compared MP3 against all the different versions of DISC and CASet from [38] and MLTD [79]. While MP3 and MLTD provide similarity scores, DISC and CASet compute a dissimilarity score, that we convert into a similarity measure by simply subtracting their value from 1.

#### Effect of changes in the tree topology

A key feature a measure on tumor phylogenies should have is to discern changes at different tree depths; indeed, a change close to the root should be more impactful than a change towards the leaves. Such a behavior is fundamental, as driver mutations are often acquired early in the evolutionary history, while less important passenger mutations usually happen at later stages: to mistake the two types of mutations should therefore have a high impact on a good similarity measure.

To estimate this effect on all the measures, we start from a linear base tree ( $T_0$  in Figure 5.5 (*Left*)); we then raise its only leaf one level at the time and compute its similarity to the base tree, expecting a drop in similarity as the leaf raises to the root, similarly to experiment proposed in [38]. Figure 5.5 (*Left*) clearly displays such effect for MP3, showing that it has the highest similarity decrease among all measures; DISC and CASet also have similar trends, but to a lower extent. Since the set of labels is the same for all trees, there is no difference between union and intersection versions of



DISC and CAsE. Contrarily, as already observed in [38], MLTD plateaus after the first change.

Another interesting aspect to investigate is how the presence of poly-occurring labels influences the similarity scores, as the more sophisticated the inference tools get, the more is common to have tumor phylogenies with multiple acquisitions or losses of the same mutation. To evaluate this aspect we started from a multi-labeled base tree with all labels occurring only once. We then created 15 perturbed trees for 5 different configurations. In the first one (on the abscissa 0 in Figure 5.5 (*Right*)) we allowed one operation excluding label duplication; for the others we allowed a total of 1, 3, 5 and 7 operations with much higher chance of selecting a label duplication. Since perturbations occur randomly, we are only sure that at most the specified number of duplication occurred, and not necessarily to the same label.

Figure 5.5 (*Right*) shows that  $CASet_{\cap}$ ,  $CASet_{\cup}$ ,  $DISC_{\cap}$  and  $DISC_{\cup}$  have similar trends in this setting, MP3 being the only one that differs. In particular, the other measures assign an higher similarity score to the second configuration than to the first one, despite they are both obtained with one perturbing operation, allowing label duplication only in the second one. MP3 is the only measure that positively displays a monotonic decrease in similarity as the number of poly-occurring labels increases, being markedly steeper than the others. We believe that a larger steepness will be more informative than a plateauing curve, since while being true that after many of poly-occurrences no more information is gained, all the duplications will inevitably add more and more noise to the tree. Since MLTD assumes that every label appears only once, it failed to run on this experiment and was therefore excluded.

## Results on simulated data

To analyze the differences between all measures we designed two experimental settings: from 5 different base trees we generated 30 perturbations for each class and computed similarities scores between all the 150 resulting trees. In the first configuration we allowed a total of 3 operations excluding label duplications, while in the second one we allowed them. All the parameters and the different probabilities used for applying perturbations are available at our supplementary repository [https://github.com/AlgoLab/mp3treesim\\_supp](https://github.com/AlgoLab/mp3treesim_supp).

Results for the first configuration are shown in Figure 5.6. The heatmaps (*Left*) show that MP3 discerns the best between the trees in the same class (main diagonal) and the others: the results of  $DISC_{\cup}$  are really close to ours, but there is a more noticeable noise outside the main diagonal.  $DISC_{\cap}$  and  $CASet_{\cup}$  present even more noise than the others, but are still mostly able to distinguish the different classes;  $CASet_{\cap}$  seems to struggle the most on this setting, while MLTD displays high values of similarities for every couple of trees, but it is still able to differentiate between the bases.

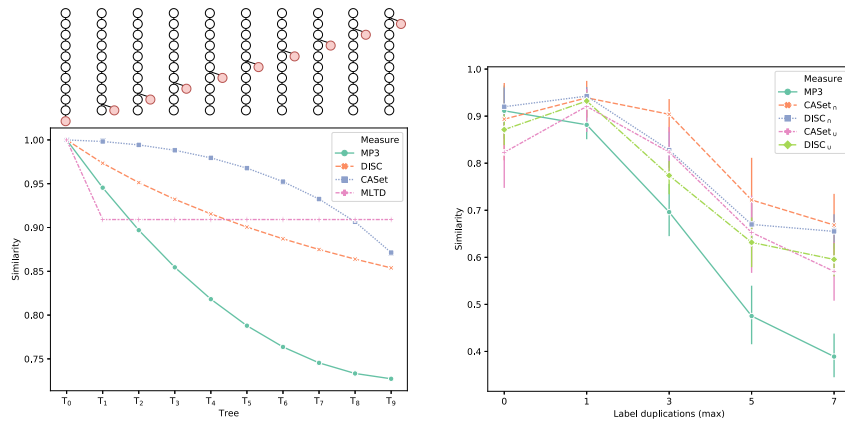


Figure 5.5: (*Left*) Effect of a node (highlighted in red) that ascends from leaf to child of the root,  $T_0$  is the base tree to which the others are compared. (*Right*) Effect of label duplication on the similarity scores. Similarities are the average of 15 trees generated from the same base with the specified maximum number of duplications. MLTD was excluded since it failed to run on instances with poly-occurring labels.

The boxplots in Figure 5.6 (*Top-Right*) show the same result quantitatively: the crucial feature is to correctly distinguish the different classes. The values represent the distribution of the similarities between the trees in the same class (Intra-similarity) and in different classes (Inter-similarity). MP3 differentiates better between intra and inter similarity, exhibiting the most compact distribution for the inter-similarities scores, while being a little more dispersed on the intra-similarity due to the action of the sigmoid, that pulls apart the values around  $1/2$ . Similarly to the previous case, DISC $_{\cup}$ , CASet $_{\cup}$  and MLTD show similar trends, while CASet $_{\cap}$  displays the most overlapping distributions.

Lastly, in Figure 5.6 (*Bottom-Right*), we computed a silhouette score from the data using a hierarchical linkage clustering with cuts from 2 to 15 to simulate a clustering scenario. Once again, MP3 performs the best expressing the maximum value for 5 cuts, being the five classes. DISC $_{\cap}$ , DISC $_{\cup}$  also show the largest value at the same cut. MLTD was excluded from the plot since it scored values close to  $-1$  for every cut, thus causing the figure to be hard to interpret.

In the second experimental setting we introduced poly-occurring labels to the simulation. Figure 5.7 exhibits results very similar to the previous ones. The main difference is that in the silhouette score (*Bottom-Right*) MP3, while still having its maximum value in correspondence of 5 cuts, is slightly lower than the other measures. On this experiment MLTD, not

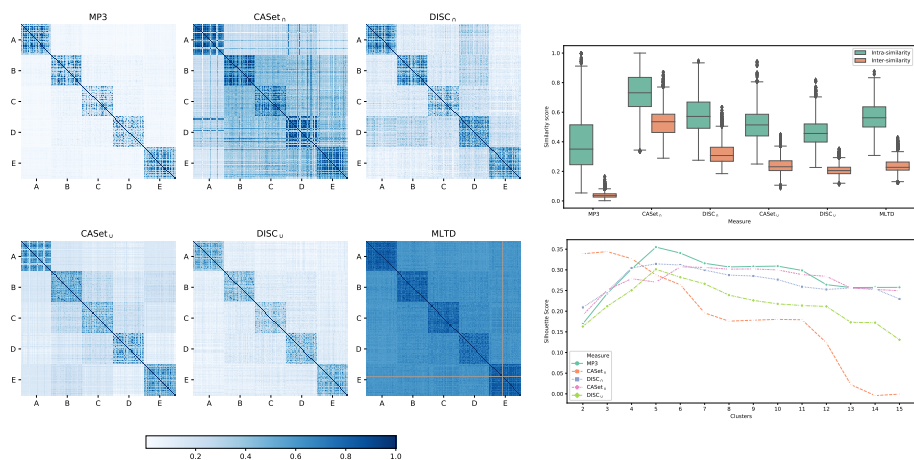


Figure 5.6: Results for the first experimental configuration: (*Left*) Heatmaps displaying the scores between all-pairs 150 simulate trees. (*Top-Right*) Distribution of the similarities between the trees in the same class (Intra-similarity) and in different classes (Inter-similarity). (*Bottom-Right*) Silhouette score computed using a hierarchical linkage clustering with cuts from 2 to 15.

allowing poly-occurring labels, failed to compute the score in most of the instances, shown in grey in the heatmaps (*Left*); it was excluded from the other plots given the high amount of failed runs. On the other hand, CASet and DISC accept input trees with poly-occurring labels, but they disregard the multiple occurrences of a same label, considering only one occurrence for each label in the computation.

### 5.3.3 Application to clustering of trees

A very important application of a tree similarity measure is clustering, e.g., to classify cancer type of patients by the similarity of their inferred phylogenies. This is of crucial interest for the development of precision therapies based on the topological structure and the evolution of mutations. Since to curate such classifications manually would be unfeasible as the size and the number of mutations increases, a good measure to use in conjunction with a clustering method is necessary.

To evaluate a similar scenario we started from 3 different bases, then perturbing two of such trees chosen at random; these new trees are then considered as additional base trees. Given this 5 bases we created a total of 10 perturbed trees from each class. The goal was to simulate an experiment with three separate classes, with two of them further split in two subclasses,

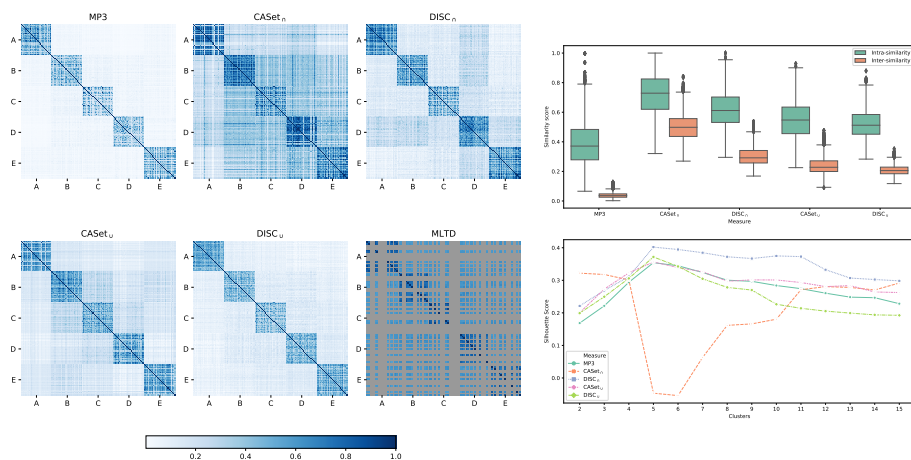


Figure 5.7: Results for the second experimental configuration: (*Left*) Heatmaps displaying the scores between all the 150 simulate trees. (*Top-Right*) Distribution of the similarities between the trees in the same class (Intra-similarity) and in different classes (Inter-similarity). (*Bottom-Right*) Silhouette score computed using a hierarchical linkage clustering with cuts from 2 to 15.

to obtain subtypes of the same cancer families.

Results for the clustering experiment are reported in Figure 5.8; (*a*) shows the clustermaps computed using hierarchical linkage clustering. MP3,  $\text{DISC}_n$  and  $\text{DISC}_U$  correctly cluster the three main families as well as the two sub-families, while both versions of CAsE $_t$  struggle the most in this experiment. Figure 5.8 (*b*) displays the distribution of intra- and inter-similarity between the five bases; MP3 has the most compact inter-similarity distribution and is the only method that completely separates intra- and inter-distributions. The high number of outliers for all methods is due to the high similarity of the two subclasses. To confirm this hypothesis we computed the same distributions only for the three main classes, remapping the subclasses to the original corresponding base class in (*d*), where we note that the number of outliers is significantly reduced. Finally, Figure 5.8 (*c*) shows the silhouette scores for the dataset; all measures express a higher score with 3 cuts, suggesting that the two subclasses are very similar to the two main bases they are derived from. The scores are very similar for all measures, with  $\text{DISC}_U$  having a higher value with 3 cuts and MP3 having a slightly higher with 5 clusters. CAsE $_n$  is the only method that have a much higher score in 5, however, as shown in (*a*), the five clusters it reports are not the correct ones. MLTD was excluded from this experiment because it failed to run on most instances due to poly-occurring labels.

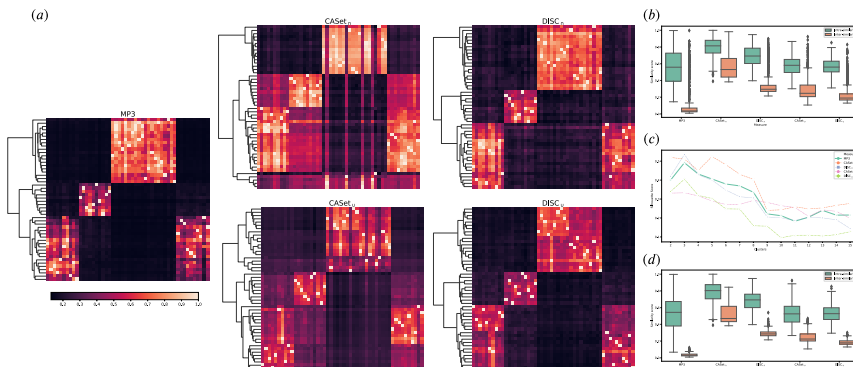


Figure 5.8: Results for the clustering experiment: (a) Clustermaps of the 50 simulated trees computed using hierarchical linkage clustering. (b) Distribution of the similarities between the trees in the same class (Intra-similarity) and in different classes (Inter-similarity) for the 5 classes. The high number of outliers for all methods is due to the high similarity of the two subclasses. (c) Silhouette score computed using a hierarchical linkage clustering with cuts from 2 to 15. (d) Distribution of the similarities between the trees in the same class (Intra-similarity) and in different classes (Inter-similarity) for the three main classes, remapping the subclasses to the original corresponding base. MLTD was excluded from this experiment because it failed to run on most instances due to the presence of poly-occurring labels.

### 5.3.4 Application to real dataset

To further evaluate our similarity measure, we applied it to two publicly available real datasets: breast cancer xenograftment in immunodeficient mice [42] and ultra-deep-sequencing of clear cell renal cell carcinoma [51]. Both datasets were previously considered for analyses by the two cancer phylogeny reconstruction methods LICHeE [107] and MIPUP [72]. Data from [42] was also used in [38] for evaluation. An interesting feature of the data in [51] is that most samples in the study present poly-occurring labels, suggesting recurrent mutations at different evolutionary stages. We recall that DISC and CASet compute dissimilarity scores, that we convert into a similarity measure subtracting their value from 1.

To evaluate the effectiveness of the measures in real scenarios, we selected the manually curated trees, published in the corresponding original sequencing studies, for case SA501 from [42] and for patient RMH002 from [51]. We then computed similarities between these reference trees and the ones inferred by LICHeE and MIPUP, as reported in [72].

The reference RMH002 is very similar to the evolutions inferred by LICHeE and MIPUP, thus most of the measures agree on a high similarity score, as reported in Figure 5.9 (*Left*), with the exception of  $\text{CASet}_{\cup}$ . The

scores computed by MP3 are higher than the others, possibly because it is the only method to correctly identify and process poly-occurring labels in the reference trees, due to the discovered recurring mutations. Differently from the previous analysis, the measures disagree considerably for SA501, as depicted in in Figure 5.9 (*Center*). Indeed, MP3 reports a similarity value close to 0, suggesting that the considered trees are quite different, whereas the other measures report a higher similarity, especially DISC scoring up to 60% similarity.

To thoroughly investigate this behavior, we defined some naïve approaches used as a proxy to analyze some basic aspects of the trees, such as the count of pairs of labels appearing in the same node in both trees. Even with such a naïve measure, the reference tree for SA501 from [42] and the trees inferred by MIPUP and LICHeE disagree considerably. The base tree contains only 50 labels, whereas the trees inferred by LICHeE and MIPUP contain 95 and 158 labels, respectively; of these, the reference shares a total of 24 label with LICHeE and 37 with MIPUP. Most importantly, only 54 out of 1759 pairs of labels appear in the same node both in the reference and LICHeE and 124 out of 8424 in MIPUP. Such evaluations, albeit very simplistic, suggest that the trees are indeed dissimilar and thus a lower score, as provided by MP3, is more reasonable than a high value of similarity.

To better understand this phenomenon, we created the edge case of a single-node tree with all the 158 labels from MIPUP, and compared it against the reference SA501. The resulting values in Figure 5.9 (*Right*) show a high similarity score for DISC with values up to 69%, with CASet and MLTD being less influenced by this aspect with scores up to 11% and 20%. On the other hand, MP3 clearly defines the trees as extremely dissimilar, with a score of 0.04%. Such results for trees that are clearly extremely different show a strong bias for DISC towards high similarity values.

	LICHeE	MIPUP		LICHeE	MIPUP		Edge case
MP3	0.997	0.897	MP3	0.017	0.004	MP3	0.0004
CASet $\cap$	0.805	0.779	CASet $\cap$	0.139	0.111	CASet $\cap$	0.0927
DISC $\cap$	0.930	0.876	DISC $\cap$	0.627	0.624	DISC $\cap$	0.5571
CASet $\cup$	0.569	0.551	CASet $\cup$	0.260	0.113	CASet $\cup$	0.1120
DISC $\cup$	0.764	0.725	DISC $\cup$	0.405	0.610	DISC $\cup$	0.6933
MLTD	0.842	0.807	MLTD	0.182	0.205	MLTD	0.2046

Figure 5.9: (*Left*) Similarities between the manually curated tree reported in [51] for patient RMH002 and the trees inferred by LICHeE and MIPUP. (*Center*) Similarities between the manually curated tree reported in [42] for sample SA501 and the trees inferred by LICHeE and MIPUP. (*Right*) Similarities between the manually curated tree reported in [42] for sample SA501 and the edge case with all mutations appearing in a single node.

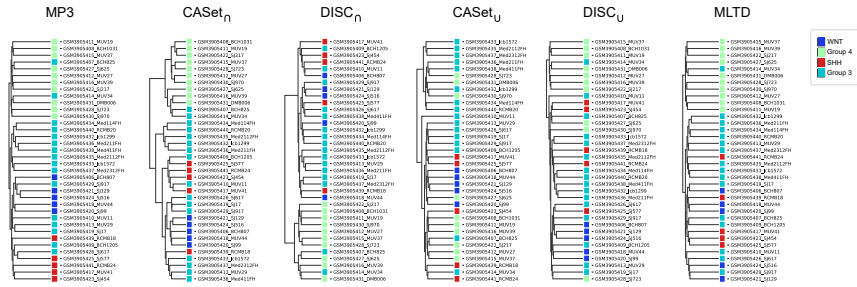


Figure 5.10: Results for the clustering experiment: Hierarchical clustering obtained from 36 medulloblastoma patients [66]. The trees were computed from the available scRNA-seq datasets using the inference tool SCITE [75]. The colors indicate the true tumor subtype of each patient.

### 5.3.5 Application to clustering of patients

As a final evaluation of the measures we computed a clustering of 36 medulloblastoma patients from [66]; the patients are classified according to four different subtypes of tumor. From the available scRNA-seq data we inferred the cancer phylogeny of each patient using SCITE [75]; we then computed the similarities between all the inferred trees and used them to perform a hierarchical clustering.

Figure 5.10 displays the clustering results for all the measures; in particular using MP3 is possible to distinctively group the patients in their relative subtypes with only a few mismatched trees. A similar result is achieved by CAsEtn, while the other measures tend to cluster together subtypes SHH and WNT, without a clear distinction between them.

## 5.4 Discussion

We identified two major limitations in the existing methods to compare tumor phylogenies: first, they are not sensitive enough to detect even major differences in the topology of the trees, as we demonstrated with ad-hoc experiments. Second, they are not able to meaningfully compare trees where the same label is assigned to more than one node.

We addressed the latter by representing tumor phylogenies as multi-labeled trees with poly-occurring labels. Such model is best suited to cancer progression than the ones previously adopted, as it allows the same mutation to appear in multiple evolutionary events, a circumstance often occurring in real applications. Being inspired by the triplet distance for classical phylogenies, our new similarity measure correctly detects differences in the topology of the trees.

Our experiments show that our method performs very well both on synthetic and real data and, unlike the other existing tools, it is able to detect differences regarding poly-occurring labels and it suitably distinguish trees with different topologies. Moreover, when applied to hierarchical clustering, it outperforms every other method.



## Chapter 6

# Tumor phylogenetic pipeline

In the recent years there has been an increasing amount of single-cell sequencing (SCS) studies, producing a considerable number of new datasets. This has particularly affected the field of cancer analysis, where more and more papers are published using this sequencing technique that allows for capturing more detailed information regarding the specific genetic mutations on each individually sampled cell.

As the amount of information increases, it is necessary to have more sophisticated and rapid tools for analyzing the samples. To this goal we developed **plastic**, an easy-to-use and quick to adapt pipeline that integrates three different steps: (1) to simplify the input data; (2) to infer tumor phylogenies; and (3) to compare the phylogenies.

We have created a pipeline submodule for each of those steps, and developed new in-memory data structures that allow for easy and transparent sharing of the information across the tools implementing the above steps.

While we use existing open source tools for those steps, we have extended the tool used for simplifying the input data, incorporating two machine learning procedures — which greatly reduce the running time without affecting the quality of the downstream analysis. Moreover, we have introduced the capability of producing some plots to quickly visualize results.

### 6.1 Introduction

During the last few years we have witnessed an explosion of computational tools to infer tumor phylogenies (also called cancer progressions) from single-cell sequencing (SCS) data.

Most of the algorithmic research has so far focused on bulk sequencing data for inferring tumor phylogenies, mainly because of the widespread availability and affordability of the next-generation sequencing (NGS) data that are used, producing a large number of tools [125, 77, 60, 136, 107, 93, 45, 95, 116, 18, 127, 135]. From a computational point of view, the main

characteristic of bulk sequencing data is that only the approximate proportion of cells with any given mutation is observable, without distinguishing the cells that carry them. Moreover, each sample contains a mixture of both healthy and cancerous cells — the latter belonging to different and unknown clones — therefore further introducing uncertainty.

More recently, the introduction of single-cell sequencing (SCS) technologies promises to greatly reduce such uncertainty, since the presence or absence of mutation is determined at the level of the cell. Unfortunately, SCS is still much more expensive than bulk sequencing, hence limiting its adoption in practice. Moreover, the quality of the data obtained from SCS is not yet at par with bulk sequencing [80]. In fact, those datasets are affected by some clearly identified problems: (1) doublet cell captures, that is, data originating from two cells instead of one; (2) false negatives from allelic dropout, that is, the presence of a mutation is not detected; and (3) missing values due to low coverage. However, all three of these problems are slowly fading away — the latter two driven by the reduction in cost that allows a higher coverage, and the first due to the development of state-of-the-art approaches [37] are able to remove such artifacts.

Various methods have been recently developed for inferring tumor phylogenies given current SCS data [75, 113, 138, 137, 30, 43, 123], some of them introducing a hybrid approach of combining both SCS and VAF (variant allele frequency, from bulk sequencing) data [110, 92, 115]. To trim the search space and reduce the time needed to infer a phylogeny, most methods rely on the infinite sites assumption (ISA), which essentially states that each mutation is acquired at most once in the phylogeny and is never lost. This assumption leads to a computationally tractable model of evolution called perfect the phylogeny [55, 81].

However, some studies [87, 20] on cancer data provide strong hints that the ISA does not always hold, the main reason is that cancers usually have large independent deletions on separate branches of the phylogeny [20]. When those deletions span a shared locus, we observe multiple deletions of the same mutation.

Relaxing the ISA greatly expands the search space, making it more difficult to develop efficient approaches. For this reason, the number of possible mutation losses is usually bounded in any (more general) model which relaxes the ISA. While the general Dollo model [47, 111] does not impose any restriction on the number of losses, more restricted models are the Dollo- $k$  and the Dollo-1 (also known as the persistent phylogeny[14, 15]).

Even though relaxing the ISA increases the complexity, some methods have appeared, such as TRaIT [110], SiFit [138], SASC [30] and SPhyR [43]. The latter is extremely relevant to our context, as it introduces the idea of clustering the input SCS matrix to reduce the running time. In [29] this idea is further developed, by devising a clustering method tailored to SCS data — while SPhyR instead relied on the ubiquitous  $k$ -means [97, 9] algorithm.

Since SCS data are becoming cheaper to produce, we expect the datasets to increase more rapidly than computing power, as mentioned explicitly in [80]. Therefore including some steps to reduce the instance matrix will become even more common in the next years.

The availability of so many tools for inferring tumor phylogenies, not to mention the parameters that those methods routinely have, means that it is easy to have several different phylogenies from the same SCS dataset, motivating the search for methods that are able to compare and cluster those phylogenies — a large cluster with several highly similar phylogenies is likely to be more reliable, since the underlying evolution is confirmed by multiple methods. In this direction, some methods to measure the distance between two tumor phylogenies have recently been proposed [38, 79, 54, 10, 11, 26, 74]. While these measures vary greatly in practical applicability, they all express the need for incorporating the evolutionary process into the definition of distance.

Our discussion so far shows that tumor phylogeny inference is crystalizing into different, well-established steps that are combined to obtain a complete tool that starts from an SCS dataset and ends with one or more phylogenies together with some rough idea of their relationships. Still, how to combine those steps is largely ad-hoc, making it more time consuming than is necessary to develop a complete pipeline.

With the goal of making the analysis of cancer data more streamlined, we developed **plastic** (PipeLine Amalgamating Single-cell Tree Inference Components), an integrated and easy to use tool that includes clustering, phylogeny inference, and comparison steps. The **plastic** tool is developed in Python submodules and can be easily integrated into any script or used inside an interactive notebook, such as Jupyter Notebook, to facilitate the reproduction of research results. Currently, **plastic** incorporates the publicly available tools *celluloid*, *SASC*, and *MP3* — respectively for the clustering, inference, and distance steps — but provides the infrastructure to easily extend it to incorporate any other tool. In fact, **plastic** provides unified in-memory data structures to manage the communication between steps.

Moreover, the current strategies for trimming SCS datasets focus on reducing the number of mutations, while leaving the set of cells unchanged. We have divided the clustering step into two parts: first reducing the number of cells, then reducing the number of mutations. We have explored two strategies for the first part that has been proven to be useful in the past, namely, ridge regression [65, 96] and autoencoder [133].

Ridge regression is a variant of least squares regression, which deals with the trade-off between bias and variance while fitting the regression line. As compared to least squares regression, ridge regression consist of an  $l_2$  penalty on the regression coefficients (see Equation 6.1). This penalty term in ridge regression helps to introduce some bias, which eventually helps to reduce the variance while fitting the regression line (hence regularizing the fit). In

least squares regression, since we do not have the bias, there will be a high variance while fitting the line. Ridge regression has been successfully used in the literature for dimensionality reduction [21, 73, 90].

Autoencoder is an unsupervised approach based on artificial neural networks. It takes as input the data matrix, encodes it into a latent space (of reduced dimension), and then tries to reconstruct the original input from the latent space. During the reconstruction process, it tries to minimize least squared error. Autoencoder has been successfully used in the literature for dimensionality reduction [109, 67, 33, 1]. For further detail about ridge regression and autoencoder, please refer to Section 6.2.1. The idea with both ridge regression and autoencoder is that we can perform downstream clustering (tree inference, etc.) in reduced dimensional data, rather than applying clustering, etc., on the high dimensional data, resulting in reduced downstream runtimes.

We have run two different experiments: one on real data to showcase all features of `plastic`, including its capability to plot clustering of mutations (*i.e.*, the output of the clustering step) and trees; and the second on simulated data, to assess the reduction in running time stemming from the reduction in the number of cells provided by the two dimensionality reduction strategies mentioned above. The `plastic` tool and all data needed to reproduce the analysis can be found at <https://github.com/plastic-phy>, including the source code of the Jupyter notebook used for the real data experiment, witnessing the simplicity of our approach. The `plastic` tool is available under the MIT license.

## 6.2 Methods

We have developed an integrated, modular, and extendable tool for inferring and comparing cancer progressions (also referred to as tumor phylogenies) called `plastic`, which integrates three separate steps into a single program that shares the same data structure among these steps. These three steps are (1) input matrix reduction, (2) tumor phylogeny inference, and (3) tumor phylogeny comparison.

One of the main contributions of our paper is the integration of different tools that usually have specific on-disk input and output file formats, therefore needing a parsing step to process the input, and a dedicated procedure to produce the output. Instead, `plastic` uses in-memory data structures to share all information across the methods.

In particular, `plastic` provides an SCS matrix data structure that is enriched with some additional information that can be shared among the different steps, and a phylogeny tree structure that is used for communication between the phylogeny inference and the phylogeny comparison steps. Such structures are transparent to the user and contribute many additional

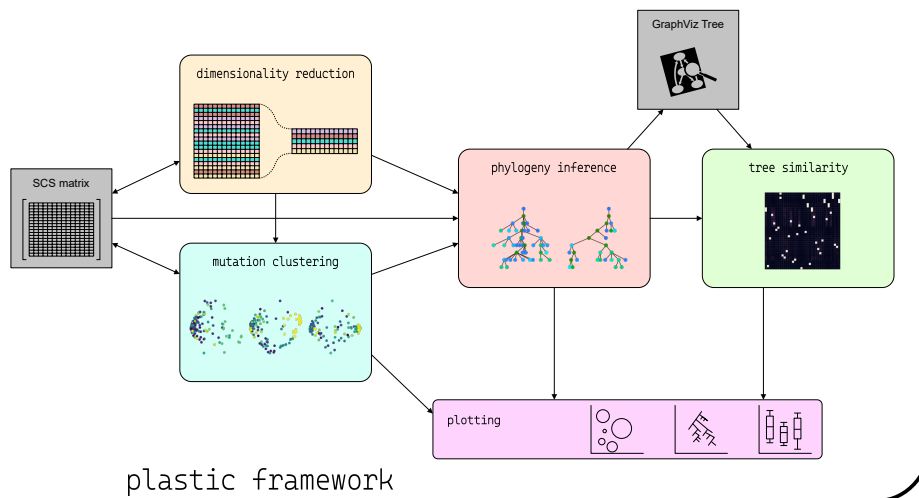


Figure 6.1: Graphical representation of the `plastic` framework and the interaction between its components and with the input/output files.

functionalities without complications from the end-user perspective.

Furthermore, given the nature of `plastic`, we added some graphical capabilities, so that each step of the pipeline can be displayed within an interactive notebook or be exported to separate files.

Finally, notice that each step is optional, therefore allowing the execution of the entire pipeline, or only of a part of it. A schematic of our `plastic` tool is depicted in Figure 6.1. We now introduce some of the new dimensionality reduction features we have added to `plastic`, as follows.

### 6.2.1 Dimensionality Reduction

Dimensionality reduction is a popular approach to enhance the performance of machine learning algorithms and to avoid the problem of the “curse-of-dimensionality” [5, 6]. To increase the clustering performance and reduce the runtime for clustering, we use two dimensionality reduction approaches, namely ridge regression [65, 96] and autoencoder [133].

The main goal of ridge regression (RR) is to find a linear function, which models the dependencies between covariate variables and univariate labels. Although ridge regression is an older approach, it is still successfully used in order to reduce the dimensions of current datasets [21, 141]. Ridge regression help to find a subspace, which most compactly expresses the target and rejects other possible but less compact candidates [73]. It works by introducing a bias term — the goal of ridge regression is to increase the bias (by changing the slope of the regression line) in order to improve the variance (generalization capability). The general expression for ridge regression is as

follows:

$$\min(\text{sum of square residuals} + \alpha \times \text{slope}^2) \quad (6.1)$$

where  $(\alpha \times \text{slope}^2)$  is an  $l_2$  penalty term. Ridge regression gives insights on which independent variables are not informative (independent variables for which we can reduce the slope close to zero). We can eliminate those independent variables to reduce the dimensions of the data. Note that after the dimensionality reduction, we are left with the variables from the actual data and not some latent variables.

Autoencoder (AE) is an unsupervised artificial neural network based approach used for dimensionality reduction [132, 134]. It consists of three the components: encoder; code; and decoder (pictured in Figure 6.2). The encoder component compresses the input data and produces the code (low dimensional latent-space representation). The decoder component then reconstructs the input only using this low dimensional representation only (hence an unsupervised approach). The low-dimensional data (in the code component) is a compact summary, or compression of the input. This compact data is used as the reduced dimensional representation of the original data. The activation function used for the encoding component is the rectified linear activation function (ReLU), while we used the sigmoid function for the decoding. The loss function that we used in our experiments is the least squared error. The optimizer that we use is *Adadelta* [106], a stochastic gradient descent method which is based on adaptive learning rate per dimension. It is a popular optimizer used in autoencoder because it avoids the continual decay of learning rates throughout training. It also helps to decide the global learning rate, hence not requiring to select it manually.

## 6.3 Results

Here we present our results on both real and simulated data.

### 6.3.1 Real data

As a result, we show a complete workflow example on a real medulloblastoma dataset [66]. In particular, we will explore a full pipeline analysis where multiple samples are taken into account. For each sample: mutations are clustered using *celluloid*; phylogeny trees are reconstructed using *SASC*; and finally, the patients are clustered using *MP3* as similarity measure between them.

The sequencing study consists of 36 patients that we want to cluster into different subtypes. To simulate a real case scenario, where such information is not available, we started from *SCS* binary matrices, one for each patient, and we clustered mutations using *celluloid* ( $k = 50$ ), for which we can see a summary in Figure 6.3.

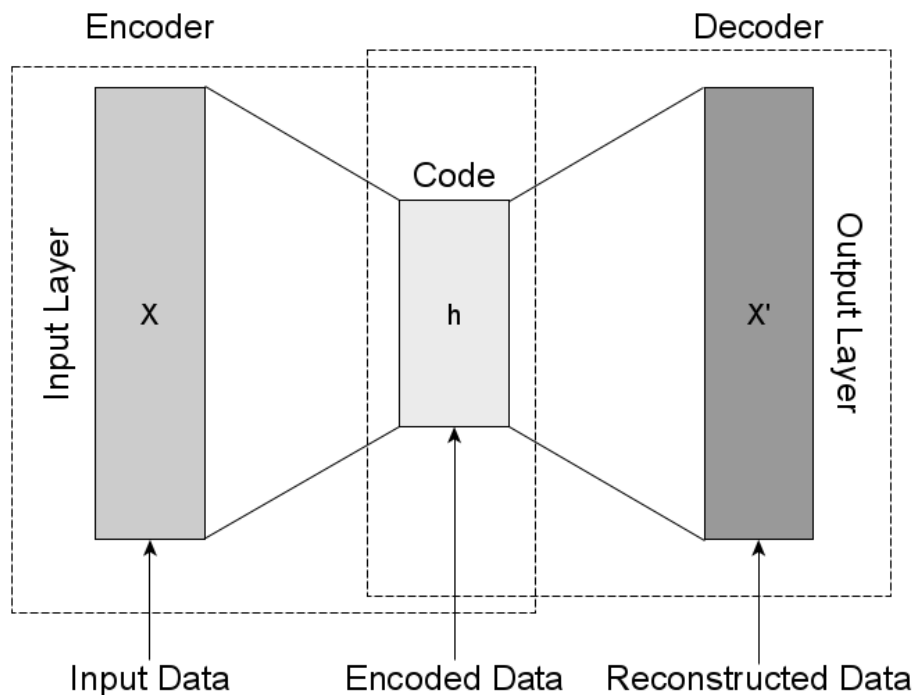


Figure 6.2: The autoencoder architecture. The encoder takes input data and maps it to a latent space (code component) of reduced dimension. The decoders code back the latent representation to the output. This architecture learns to compress data by reducing the reconstruction error (least squared error).

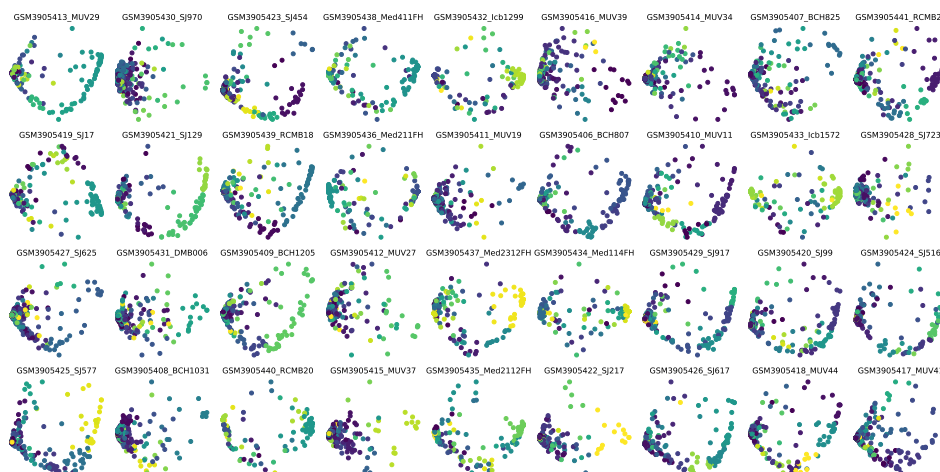


Figure 6.3: Mutations clustered on SCS data computed by the *celluloid* submodule and displayed by *plastic* for the 36 medulloblastoma patients in the dataset of [66]. Different colors represent different clusters.

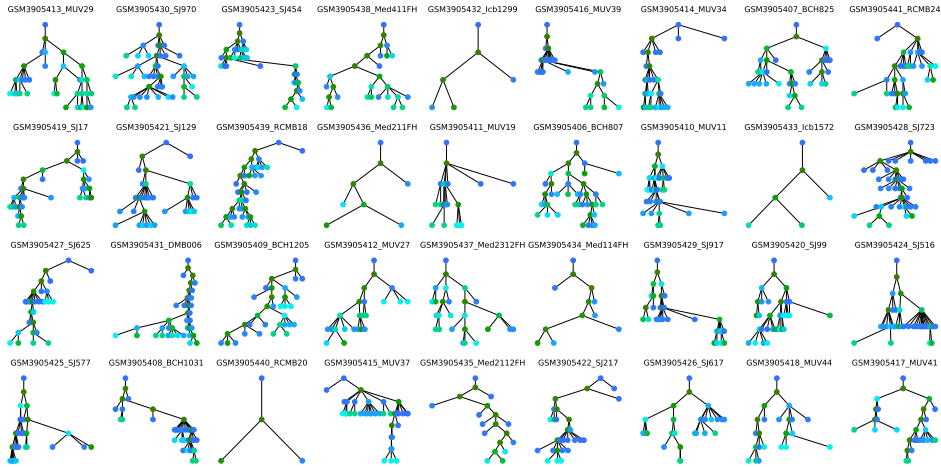


Figure 6.4: Trees computed by the **SASC** submodule and displayed by **plastic** for the 36 medulloblastoma patients in the dataset of [66]. Different colors represent different SCS support, as defined in [30].

After the first preprocessing step, we utilized the **SASC** submodule of **plastic** to infer the evolutionary trees of each patient. As a proof of concept we run it using the following parameters  $\alpha = 0.25$ ,  $\beta = 1 \times 10^{-4}$ ,  $k = 0$ ,  $d = 0$  for whose definitions we refer to *SASC*'s manuscript [30]. Once computed we used the newly-added plotting feature to display the phylogenies using the *SASC-viz* plotting feature to prettify the trees; the result is shown in Figure 6.4.

As the last step, we then used the **MP3** submodule to compute a matrix of similarity-scores between all the trees and then used it to cluster the patients according to a hierarchical clustering method. We then displayed the final clustering of the patients in Figure 6.5.

### 6.3.2 Simulated data

To evaluate the performance of the input matrix reduction methods that we present here, we designed a two-fold experiment on synthetic datasets. We first measure the quality of the matrix obtained from the reduction step — in this case we have a ground truth that we can use for the evaluation. Second, we evaluate the accuracy and runtime of the downstream phylogeny inference tools when given as input the reduced instance from the first step, to assess that our reduction step is actually useful. This is similar to the experimental approach taken in [29].

The simulated data are generated as follows. First we generate a random tree topology on  $s$  nodes, each representing a tumor clone, by first creating a root (the germline) and then iteratively attaching the  $s - 1$  remaining nodes



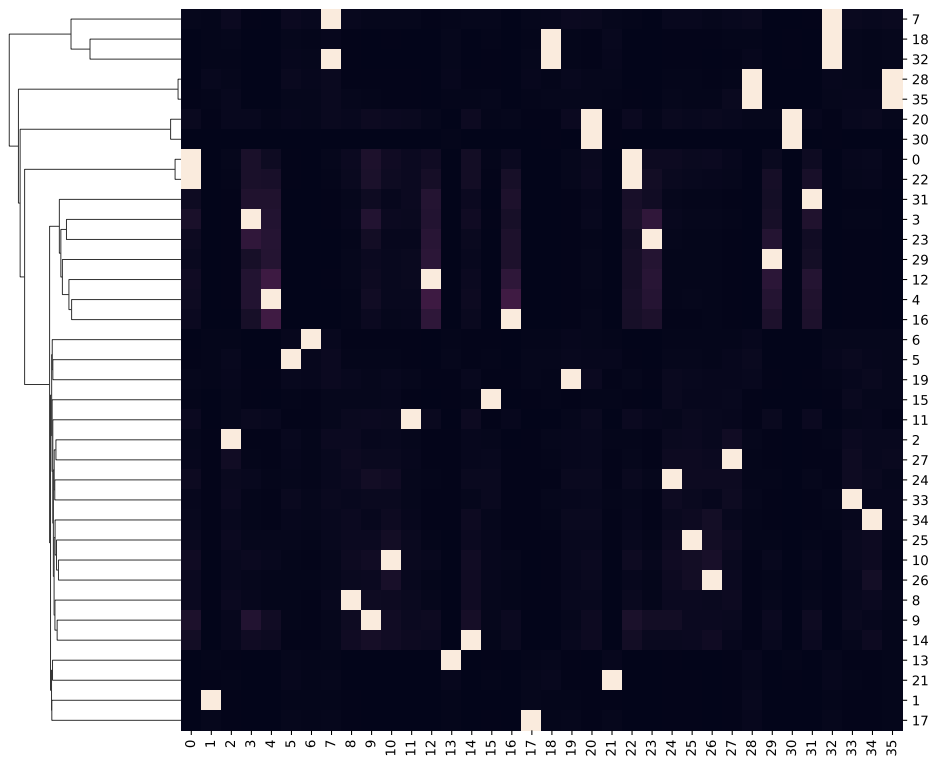


Figure 6.5: Clustermap based on the similarity between the trees computed by the mp3 submodule of `plastic` for the 36 medulloblastoma patients in the dataset of [66].

uniformly at random to any other node in the tree. The nodes are then randomly labeled with  $m$  mutations — each mutation understood as being acquired at the node that it labels. Then, a total of  $n$  cells are randomly associated to the  $s$  nodes. Each cell harbours all of the mutations on the path in the tree from the root to the node that the cell labels. A binary  $n \times m$  matrix  $M$  is then obtained from the cells, where  $M[i, j] = 1$  if cell  $i$  harbours mutation  $j$ , otherwise  $M[i, j] = 0$ . Noise is then added to this matrix according to the false negative, false positive and missing value rates, to simulate a real single cell sequencing experiment. Each of the  $s$  nodes is therefore considered as a natural (true) cluster of the simulated dataset.

We design three experiments, each with 100, 200 and 300 cells, respectively. In each experiment, we generate 50 simulated datasets with its corresponding number of cells, using the procedure mentioned above. In all three experiments, the number  $s$  of clones is 20, and the number  $m$  of mutations is 1000.

To each dataset of our experiments, we first performed dimensionality reduction in the cells using both the ridge regression and autoencoder techniques mentioned in Section 6.2. The number of dimensions selected in case of autoencoder are 50, 100, and 150 for experiments 1, 2 and 3, respectively. This number of dimensions, in each case, is selected empirically when the least squared error for the loss function is minimized. The loss function gives us the error value when autoencoder tries to reconstruct the input in the decoder component. The main goal in this case is to reconstruct the input as accurately as possible, giving rise to the respective numbers of dimensions above. The number of dimensions selected in the case of ridge regression is different for each dataset because ridge regression is a data driven technique, however the average number of dimensions for each of experiments 1, 2 and 3 is roughly 49, 97, and 146, respectively (see Table 6.1). Following dimensionality reduction in the cells, we then clustered the mutations using *celluloid*.

### Evaluating input matrix reduction.

Because we are performing dimensionality reduction only on the cells, we can still use the same measures of precision and recall used in [29], which are based on how mutations are clustered together, as follows:

**Precision:** measures how well mutations are clustered together. For each pair of mutations appearing in the same clone in the simulated tree, we check if they are in the same cluster, resulting in a *true positive* ( $TP$ ). For each pair of mutations clustered together that are not in the same clone, we encounter a *false positive* ( $FP$ ). The value of the precision is then calculated with the standard formula:  $\frac{TP}{TP+FP}$ .

**Recall:** measures how well mutations are separated. For each pair of

Experiment	mean $\pm$ SD
1	48.7 $\pm$ 2.51
2	97.0 $\pm$ 4.71
3	146 $\pm$ 5.85

Table 6.1: Number of cells selected by ridge regression from the experiments. The mean ( $\mu$ )  $\pm$  standard deviation ( $\sigma$ ) of the number of cells selected from the 50 simulated datasets after applying dimensionality reduction using ridge regression. Calculations reported to three significant digits. The original (full) table, from which these aggregates were performed, can be found at <https://github.com/plastic-phy/plastic/blob/master/data/cells-ridge.csv>

mutations in the same clone, we now also check if they are not in the same cluster, resulting in a *false negative* ( $FN$ ). The recall is then calculated as:  $\frac{TP}{TP+FN}$ .

Notice that, just as in [29], we are mostly interested in obtaining a high precision. The reason for focusing on obtaining high precision is that since cancer phylogeny inference algorithms can later cluster together mutations — for example, by assigning them to the same node or the same non-branching path — however they cannot separate mutations that have been erroneously clustered together. It is for this same reason that the number ( $k$ ) of clusters is carefully chosen with high precision in mind.

On the other hand, ridge regression does not allow to determine a priori the number of cells that will be obtained. For this reason we report, in Table 6.1, the distributions<sup>1</sup> of the actual number of cells obtained.

Figures 6.6, 6.7, and 6.8 correspond to the precision and recall of the clusterings of the mutations obtained after first applying dimensionality reduction in the cells of experiments 1, 2 and 3, respectively.

### Evaluating the effect of the reduction step on downstream phylogeny inference

The goal of dimensionality reduction (in the cells) and clustering (of the mutations) is to allow for a significant decrease in the runtime of the phylogeny inference, the most expensive step in the pipeline — as long as it does not worsen the accuracy. The idea is that, since the reduced matrix has much fewer rows and columns, using it in place of the original matrix as input to the phylogeny inference should result in much lower runtimes, at the risk of decreasing the accuracy of downstream analysis, due to errors introduced in the dimensionality reduction and clustering steps. The main observation is

<sup>1</sup>note that the variance of all three distributions is small

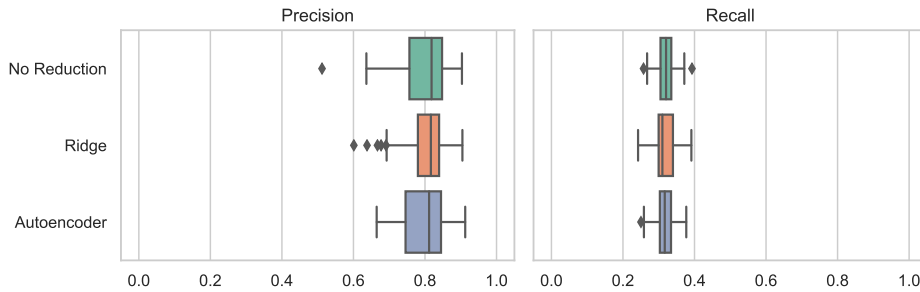


Figure 6.6: Precision and recall results of the clusterings of the mutations obtained for experiment 1, generated with 1000 mutations and 100 cells. The first row corresponds to the results obtained by clustering the mutations to  $k = 100$  clusters using *celluloid* (no dimensionality reduction). The second and third rows correspond to first applying a dimensionality reduction step on the 100 cells, using ridge regression and autoencoder, respectively, followed by the clustering of the 1000 mutations to  $k = 100$  clusters (in the reduced number of cells) using *celluloid*.

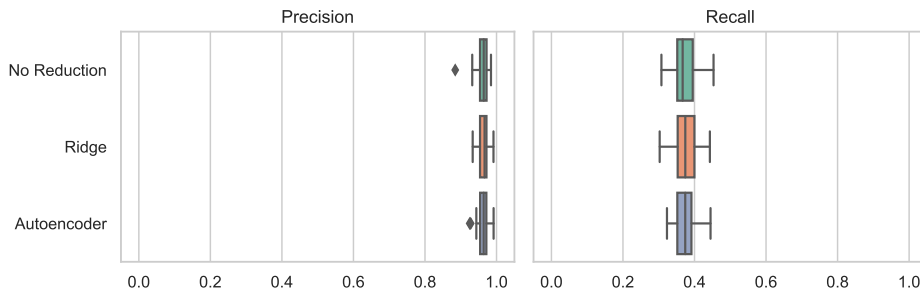


Figure 6.7: Precision and recall results of the clusterings of the mutations obtained for experiment 2, generated with 1000 mutations and 200 cells. The first row corresponds to the results obtained by clustering the mutations to  $k = 100$  clusters using *celluloid* (no dimensionality reduction). The second and third rows correspond to first applying a dimensionality reduction step on the 200 cells, using ridge regression and autoencoder, respectively, followed by the clustering of the 1000 mutations to  $k = 100$  clusters (in the reduced number of cells) using *celluloid*.

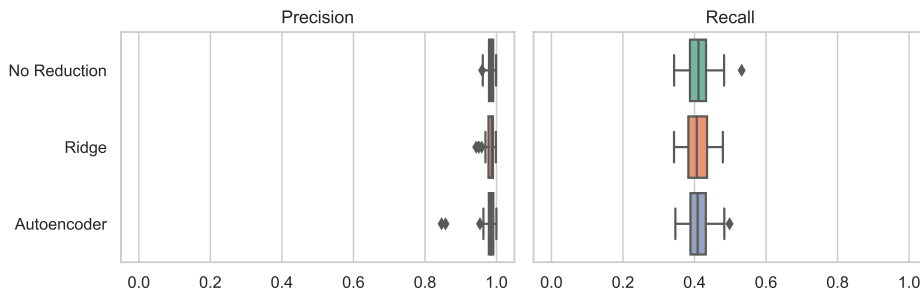


Figure 6.8: Precision and recall results of the clusterings of the mutations obtained for experiment 3, generated with 1000 mutations and 300 cells. The first row corresponds to the results obtained by clustering the mutations to  $k = 100$  clusters using *celluloid* (no dimensionality reduction). The second and third rows correspond to first applying a dimensionality reduction step on the 300 cells, using ridge regression and autoencoder, respectively, followed by the clustering of the 1000 mutations to  $k = 100$  clusters (in the reduced number of cells) using *celluloid*.

that, if the clusters are highly precise, the decrease of the accuracy of the downstream inference is negligible or small. Assessing this fact is the main goal of our experimental analysis, based on the measures used in [28, 30]:

**Ancestor-descendant accuracy:** For each pair  $(m_1, m_2)$  of mutations such that  $m_1$  is an ancestor of  $m_2$  in the ground truth, we check whether  $m_1$  is an ancestor of  $m_2$  also in the inferred tree (*TP*) or whether it is not (*FN*). Moreover, each pair  $(m_1, m_2)$  of mutations such that  $m_1$  is an ancestor of  $m_2$  in the inferred tree but not in the ground truth, we consider that pair a false positive.

**Different lineages accuracy:** Similarly to the previous measure, we check whether mutations in different branches (*i.e.*, neither is an ancestor of the other) are correctly inferred or if any pair of mutation is erroneously inferred in different branches.

Since these new dimensionality reduction techniques are built into the **plastic** framework, one can customize a cancer phylogeny inference task by:

1. choosing any (or none of the) dimensionality reduction steps from ridge regression or autoencoder; followed by
2. clustering (or not) with *celluloid*; and finally
3. phylogeny inference with **SASC** in either the reduced set of cells, or not.

An extensive study of how clustering can reduce the runtime (and sometimes improve accuracy) of the downstream phylogeny inference appears in [29], and so here we explore the further gains in runtime and/or accuracy that might be achieved by adding a dimensionality reduction step. Hence, in our experiments, we try three different choices for the reduction step — ridge regression, autoencoder, and no dimensionality reduction to produce a matrix to be given to *celluloid* (notice that **SASC** will receive the matrix on the original set of cells, but with the clustered mutations). Moreover, when applying ridge regression, we also provide the reduced set of cells to **SASC**, hence resulting in four different settings — when no dimensionality reduction is performed, the reduced and the original sets of cells are the same, while autoencoder produces reduced matrices in its learned latent tensor space, which is not an SCS matrix that can be fed to **SASC**. We always perform the clustering step, because 1000 mutations is prohibitive for the downstream phylogeny inference (see [29]).

Figures 6.9, 6.10 and 6.11 report the ancestor-descendant and different lineages accuracy measures of the trees obtained by running **plastic** with the four settings mentioned above. From the results, it is clear that when dimensionality reduction is followed by inference in the original set of cells, there is no noticeable loss in accuracy. In these two cases, the clustering is still performed in the reduced set of cells, providing a speedup at no cost. On the other hand, we see a slight loss of accuracy in the ancestor-descendant measure when ridge regression is applied, and the resulting reduced set of cells is used also in the inference step.

We have investigated the speedup provided by ridge regression. In Figure 6.12 are represented the running times of *celluloid* with and without first applying ridge regression, as a function of the number of mutations. More precisely, we take the first columns of the datasets of Experiment 1 (the number of mutations is the  $x$  axis).

Finally, in Table 6.2 we report the running times of **SASC** when ridge regression is applied, and the resulting reduced set of cells is used also in the inference step, that is, the case where we have observed a slight loss of accuracy in the ancestor-descendant measure. In this case the phylogeny inference is up to 100 times faster, making therefore possible to run **SASC** on more mutations.

## 6.4 Discussion

Given the large amount of SCS cancer studies that are being published, there is a need for a fast and easy-to-use framework that allows to perform the needed analyses. We developed **plastic** with this goal in mind, a pipeline composed of different submodules that allow for dimensionality reduction on SCS cells, clustering of SCS mutations, inference of tumor phylogenies,

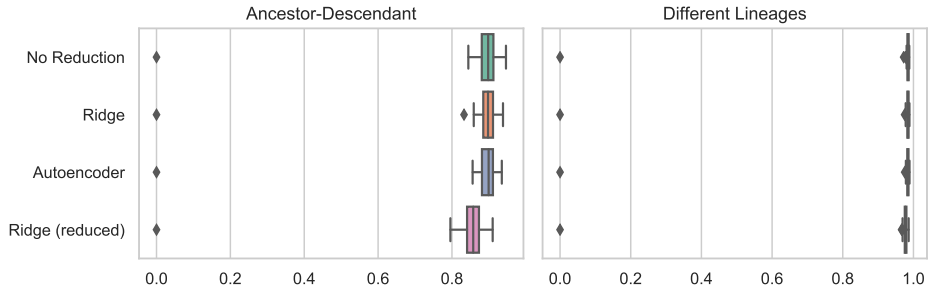


Figure 6.9: Ancestor-descendant and different lineages accuracies of the trees obtained by running `plastic` on the data of experiment 1. From the top, the four rows correspond to the settings: no dimensionality reduction, just clustering — No Reduction —; ridge regression + clustering in the original set of cells — Ridge —; autoencoder + clustering in the original set — Autoencoder —; and ridge regression + clustering in the reduced set — Ridge(reduced).

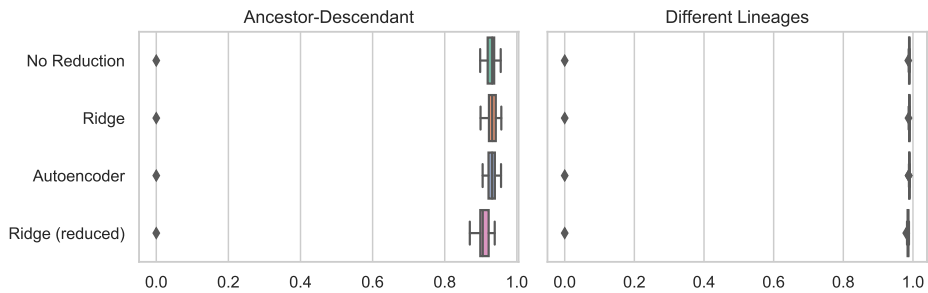


Figure 6.10: Ancestor-descendant and different lineages accuracies of the trees obtained by running `plastic` on the data of experiment 2. From the top, the four rows correspond to the settings: no dimensionality reduction, just clustering — No Reduction —; ridge regression + clustering in the original set of cells — Ridge —; autoencoder + clustering in the original set — Autoencoder —; and ridge regression + clustering in the reduced set — Ridge(reduced).

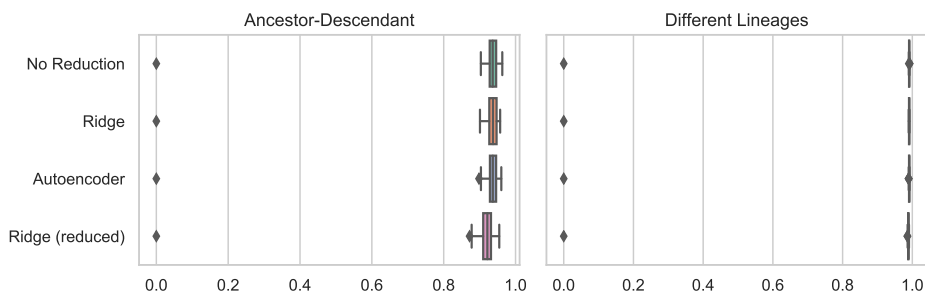


Figure 6.11: Ancestor-descendant and different lineages accuracies of the trees obtained by running `plastic` on the data of experiment 3. From the top, the four rows correspond to the settings: no dimensionality reduction, just clustering — No Reduction —; ridge regression + clustering in the original set of cells — Ridge —; autoencoder + clustering in the original set — Autoencoder —; and ridge regression + clustering in the reduced set — Ridge(reduced).

Experiment	<b>celluloid</b>	<b>RR + celluloid</b>
1	$37.7 \pm 24.3$	$0.414 \pm 0.0710$
2	$44.3 \pm 21.2$	$0.832 \pm 0.0682$
3	$34.2 \pm 25.6$	$1.31 \pm 0.112$

Table 6.2: Runtimes of `SASC` on the experiments. The mean ( $\mu$ )  $\pm$  standard deviation ( $\sigma$ ) of the runtime of `SASC` on the 50 datasets of each of the three experiments (on the original set of cells), after having either: clustered the mutations of the input matrix using `celluloid` (**celluloid**); or performed dimensionality reduction on the cells using ridge regression, followed by clustering the mutations with `celluloid` (**RR + celluloid**). Times are reported in hours, and to three significant digits.



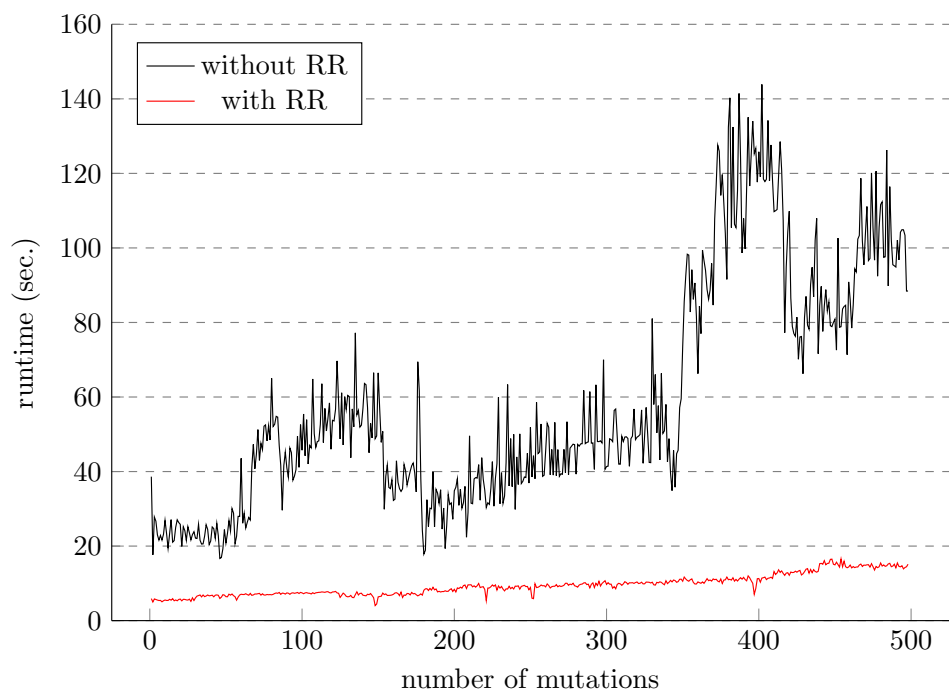


Figure 6.12: Runtime analysis, as a function of the number of mutations, of the clustering step (*celluloid*) without dimensionality reduction using ridge regression (in black), and with ridge regression (in red).

comparison of such trees, and convenient plotting of the results.

Each of the submodules can be used independently or they can be used in conjunction with each other to create complex operations, due to special data structures developed for the interaction between different methods.

The `plastic` pipeline is open-source and available at <https://github.com/plastic-phy> along with extensive documentation and a Jupyter Notebook that replicates the real-case scenario depicted in Section 6.3.1.

Future improvements for `plastic` would be to include more tools into the pipeline to extend the breadth of analyses available and to provide additional algorithmic alternatives for the same task. Moreover, we could incorporate more steps, such as the removal of doublets, to streamline the entire pipeline even more.

## Chapter 7

# Future work

In this manuscript we described the main computational challenges of the cancer phylogenetic field and we proposed different solutions for the three main problems of (i) the progression reconstruction of a tumor sample, (ii) the clustering of SCS data to allow for a cleaner and faster inference and (iii) the evaluation of different phylogenies. Furthermore we combined them into a usable pipeline to allow for a faster analysis.

The methods described, while designed for tumoral evolution, are general enough that they can be used directly, or with small modifications, for many different phylogenetic computational problems. In particular we are developing methods to apply these concepts to other type of evolution, such as viral comparison for which an example is shown in [27] where we developed a method for genotyping variants of viruses in a samples; we analyzed the method in the settings of the SARS-CoV-2 pandemic and we included phylogenomic assignment of the viral strain.

In conjunction with the viral analysis we are expanding our methods for the reconstruction of tumoral evolution on multiple samples in different time-points at various stages of the disease. We are also working on more theoretical advancement for the definition of distances between phylogenies.

## 7.1 List of publications

### Journal publications

- [29] *Simone Ciccolella*, Murray Patterson, Paola Bonizzoni, Gianluca Della Vedova. **Effective Clustering for Single Cell Sequencing Cancer Data**. IEEE J Biomed Health Inform. 2021 May 18; doi: 10.1109/JBHI.2021.3081380.
- [30] *Simone Ciccolella*, Camir Ricketts, Mauricio Soto Gomez, Murray Patterson, Dana Silverbush, Paola Bonizzoni, Iman Hajirasouliha, Gianluca Della Vedova. **Inferring Cancer Progression from Single Cell Sequencing while allowing loss of mutations**. Bioinformatics (2021); doi: 10.1093/bioinformatics/btaa722.
- [26] *Simone Ciccolella*, Giulia Bernardini, Luca Denti, Paola Bonizzoni, Marco Previtali, Gianluca Della Vedova, **Triplet-based similarity score for fully multilabeled trees with poly-occurring labels**. Bioinformatics (2020); doi: 10.1093/bioinformatics/btaa676
- [28] *Simone Ciccolella*, Mauricio Soto Gomez, Murray Patterson, Gianluca Della Vedova, Iman Hajirasouliha, Paola Bonizzoni. **gpps: an ILP-based approach for inferring cancer progression with mutation losses from single cell data**. BMC Bioinformatics 21, 413 (2020). doi: 10.1186/s12859-020-03736-7
- [94] Salem Malikic, Farid Rashidi Mehrabadi, *Simone Ciccolella*, Md Khale-  
dur Rahman, Camir Ricketts, Eshan Haghshenas, Daniel Seidman,  
Faraz Hach, Iman Hajirasouliha, S. Cenk Sahinalp. **PhISCS: a combinatorial approach for subperfect tumor phylogeny reconstruction via integrative use of single-cell and bulk sequencing data**. Genome Res. 2019 Nov; doi: 10.1101/gr.234435.118.
- [18] Paola Bonizzoni, *Simone Ciccolella*, Gianluca Della Vedova, Mauricio Soto. **Does Relaxing the Infinite Sites Assumption Give Better Tumor Phylogenies? An ILP-Based Comparative Approach**. IEEE/ACM Trans Comput Biol Bioinform. 2019 Sep-Oct; doi: 10.1109/TCBB.2018.2865729.

### Work in progress

- [27] *Simone Ciccolella*, Luca Denti, Paola Bonizzoni, Gianluca Della Vedova, Yuri Pirola, Marco Previtali **MALVIRUS: an integrated web application for viral variant calling**.
- [4] Sarwan Ali, *Simone Ciccolella*, Lorenzo Lucarella, Gianluca Della Vedova, Murray D Patterson. **Simpler and faster development of**

**tumor phylogeny pipelines.** Journal of Computation Biology, to appear.

# Glossary

**CSP** Constraint Satisfaction Problems; they are a set of mathematical problems modeled as Boolean expressions of constraints that need to be satisfied in order to be solved. 19, 47, 48

**Dollo** Dollo phylogeny; it is a phylogeny model that allows losses of mutations. In its most theoretical form each mutation can be lost infinite times, but more realistic approaches utilize variations in the form of Dollo- $k$  model, where each mutation can be lost at most  $k$  times in the entire tree. It is the most used loss model and commonly referred to in this manuscript. 16–18, 21, 23, 25–29, 33, 38–43, 97

**ILP** Integer Linear Programming; it is a mathematical optimization of a problem defined in term of an objective function over a set of integer variable that are constrained by a set linear equations or inequalities. ILP are typically modified by allowing the use of real variables, transforming the problem in a Mixed Integer Linear Programming (MILP). 15, 19, 33–35, 47, 48, 50, 54, 55

**ISA** Infinite Sites Assumption; it is the most used assumption used in cancer evolution inference that states that no two mutations can occur at the same site (or locus), meaning that once a mutation is acquired it cannot be lost or modified. 7, 13, 14, 16, 21–23, 32, 42, 47, 48, 50, 51, 55, 97

**PP** Perfect Phylogeny; it is the simplest phylogeny model following directly the ISA. Given its simplicity it is the most widely used, since in its basic form – not accounting for errors – can be solved in polynomial time. 7, 14–18, 48–50

**SA** Simulated Annealing; it is one of many nature-inspired heuristic algorithms, in this case inspired by the annealing of metals. It is commonly used to explore large search space for the optimal solutions; it utilizes a *temperature* value that decreases over time. The probability of accepting a new solution is dependent on the value of the temperature;

with high values it can easily move to less optimal solution, while the temperature decreases the probability also decreases, transforming the algorithms from a global to a local search. 19, 27–29

**SAT** Satisfiability problem; it is a logic problem that states whether a Boolean expression is satisfied or not. They are the basis for the CSP modeling. 47, 48, 54, 55

**SCS** Single Cell Sequencing; it is a technology to sequence genomic DNA or RNA from specific cells. SCS is one of the most recent methods and it is rapidly gaining popularity given the higher amount of details it provides; however in its current state it tends to contain many errors although they are bound to decrease as the technology improves. 7, 12, 13, 19, 21–23, 30, 47, 48, 51, 59–61, 69, 71, 78, 79, 96–99, 101, 102, 109, 114

**SNV** Single Nucleotide Variation; also called Single Nucleotide Polymorphism (SNP) is the biological occurrence of a different nucleotide in a position in respect to what should be in the reference genome. Some SNVs are germline to the individual – meaning that they are born with them – and they define its phenotypes; others are somatic – meaning that they arise during the course of life – and can be deleterious to the individual. SNVs of either type can be benign (or have no effect at all) or can be predictor (or driver) of particular diseases. Most tumors are driven by specific somatic SNVs. 24, 35, 37, 51

**VAF** Variant Allele Frequency; it is the fraction (percentage) of sequenced reads that match a specific genomic variant. VAFs are a typical method in cancer analysis for estimating the fraction of the samples that express a variation of interest. 12, 22, 48, 51–54, 56, 57, 97

# Bibliography

- [1] Jakob Abeßer, Stylianos Ioannis Mimilakis, Robert Gräfe, Hanna Lukashevich, and IDMT Fraunhofer. Acoustic scene classification by combining autoencoder-based dimensionality reduction and convolutional neural networks. In *Proc. of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, pages 7–11, 2017.
- [2] Nuraini Aguse, Yuanyuan Qi, and Mohammed El-Kebir. Summarizing the solution space in tumor phylogeny inference by multiple consensus trees. *Bioinformatics*, 35(14):i408–i416, 07 2019.
- [3] Alfred V. Aho, Yehoshua Sagiv, Thomas G. Szymanski, and Jeffrey D. Ullman. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM Journal on Computing*, 10(3):405–421, 1981.
- [4] Sarwan Ali, Simone Ciccolella, Lorenzo Lucarella, Gianluca Della Vedova, and Murray Patterson. Simpler and faster development of tumor phylogeny pipelines. *bioRxiv*, 2021.
- [5] Sarwan Ali, Haris Mansoor, Naveed Arshad, and Imdadullah Khan. Short term load forecasting using smart meter data. In *Proceedings of the Tenth ACM International Conference on Future Energy Systems*, pages 419–421, 2019.
- [6] Sarwan Ali, Haris Mansoor, Imdadullah Khan, Naveed Arshad, Muhammad Asad Khan, and Safiullah Faizullah. Short-term load forecasting using ami data. *arXiv preprint arXiv:1912.12479*, 2019.
- [7] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.
- [8] Mario Alviano, Carmine Dodaro, and Francesco Ricca. A maxsat algorithm using cardinality constraints of bounded size. In *IJCAI*, pages 2677–2683, 2015.



- [9] M.R. Anderberg. *Cluster Analysis for Applications*. Academic Press, 1973.
- [10] Giulia Bernardini, Paola Bonizzoni, Gianluca Della Vedova, and Murray Patterson. A rearrangement distance for fully-labelled trees. In *30th Annual Symposium on Combinatorial Pattern Matching (CPM 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.
- [11] Giulia Bernardini, Paola Bonizzoni, and Paweł Gawrychowski. On Two Measures of Distance Between Fully-Labelled Trees. In Inge Li Gørtz and Oren Weimann, editors, *31st Annual Symposium on Combinatorial Pattern Matching (CPM 2020)*, volume 161 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6:1–6:16, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
- [12] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.
- [13] P. Bonizzoni, C. Braghin, R. Dondi, and G. Trucco. The binary persistent perfect phylogeny. *Theoretical Computer Science*, 454:51–63, 2012.
- [14] Paola Bonizzoni, Chiara Braghin, Riccardo Dondi, and Gabriella Trucco. The binary perfect phylogeny with persistent characters. *Theoretical computer science*, 454:51–63, 2012.
- [15] Paola Bonizzoni, Anna Paola Carrieri, Gianluca Della Vedova, Raffaella Rizzi, and Gabriella Trucco. A colored graph approach to perfect phylogeny with persistent characters. *Theoretical Computer Science*, 658:60–73, 2017.
- [16] Paola Bonizzoni, Simone Ciccolella, Gianluca Della Vedova, and Mauricio Soto. Beyond Perfect Phylogeny: Multisample Phylogeny Reconstruction via ILP. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, ACM-BCB '17, pages 1–10, New York, NY, USA, 2017. ACM.
- [17] Paola Bonizzoni, Simone Ciccolella, Gianluca Della Vedova, and Mauricio Soto. Does relaxing the infinite sites assumption give better tumor phylogenies? an ilp-based comparative approach. *IEEE/ACM transactions on computational biology and bioinformatics*, 16(5):1410–1423, 2018.

- [18] Paola Bonizzoni, Simone Ciccolella, Gianluca Della Vedova, and Mauricio Soto Gomez. Does relaxing the infinite sites assumption give better tumor phylogenies? an ILP-based comparative approach. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pages 1–1, 2018.
- [19] Gerth Stølting Brodal, Rolf Fagerberg, Thomas Mailund, Christian NS Pedersen, and Andreas Sand. Efficient algorithms for computing the triplet and quartet distance between trees of arbitrary degree. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1814–1832. SIAM, 2013.
- [20] David Brown, Dominiek Smeets, Borbála Székely, et al. Phylogenetic analysis of metastatic progression in breast cancer using somatic mutations and copy number aberrations. *Nature Communications*, 8:14944, Apr 2017.
- [21] Si-Bao Chen, Chris HQ Ding, and Bin Luo. Linear regression based projections for dimensionality reduction. *Information Sciences*, 467:74–86, 2018.
- [22] Markus Chimani, Sven Rahmann, and Sebastian Böcker. Exact ilp solutions for phylogenetic minimum flip problems. In *Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology*, pages 147–153. ACM, 2010.
- [23] F. Chung. Spectral graph theory. In *Conference Board of the Mathematical Sciences Regional Conference Series in Mathematics*, volume 92, 1997.
- [24] Simone Ciccolella. <https://github.com/algolab/celluloid/>.
- [25] Simone Ciccolella. <https://pypi.org/project/celluloid-clust/>.
- [26] Simone Ciccolella, Giulia Bernardini, Luca Denti, Paola Bonizzoni, Marco Previtali, and Gianluca Della Vedova. Triplet-based similarity score for fully multi-labeled trees with poly-occurring labels. *Bioinformatics*, 2020.
- [27] Simone Ciccolella, Luca Denti, Paola Bonizzoni, Gianluca Della Vedova, Yuri Pirola, and Marco Previtali. Malvirus: an integrated web application for viral variant calling. *bioRxiv*, 2020.
- [28] Simone Ciccolella, Mauricio Soto Gomez, Murray Patterson, Gianluca Della Vedova, Iman Hajirasouliha, and Paola Bonizzoni. gpps: an ilp-based approach for inferring cancer progression with mutation losses from single cell data. In *8th IEEE International Conference on*

*Computational Advances in Bio and Medical Sciences, ICCABS 2018, Las Vegas, NV, USA, October 18-20, 2018*, page 1. IEEE Computer Society, Oct 2018. to appear.

- [29] Simone Ciccolella, Murray Patterson, Paola Bonizzoni, and Gianluca Della Vedova. Effective clustering for single cell sequencing cancer data. *IEEE Journal of Biomedical and Health Informatics*, pages 1–1, 2021.
- [30] Simone Ciccolella, Camir Ricketts, Mauricio Soto Gomez, Murray Patterson, Dana Silverbush, Paola Bonizzoni, Iman Hajirasouliha, and Gianluca Della Vedova. Inferring cancer progression from Single-Cell Sequencing while allowing mutation losses. *Bioinformatics*, 37(3):326–333, 08 2020.
- [31] International Human Genome Sequencing Consortium. Finishing the euchromatic sequence of the human genome. *Nature*, 431:931 EP –, Oct 2004. Article.
- [32] International Human Genome Sequencing Consortium, Eric S. Lander, Lauren M. Linton, Bruce Birren, Chad Nusbaum, Michael C. Zody, Jennifer Baldwin, Keri Devon, Ken Dewar, Michael Doyle, William FitzHugh, Roel Funke, Diane Gage, Katrina Harris, Andrew Heaford, John Howland, Lisa Kann, Jessica Lehoczky, Rosie LeVine, Paul McEwan, Kevin McKernan, James Meldrim, Jill P. Mesirov, Cher Miranda, William Morris, Jerome Naylor, Christina Raymond, Mark Rosetti, Ralph Santos, Andrew Sheridan, Carrie Sougnuez, Nicole Stange-Thomann, Nikola Stojanovic, Aravind Subramanian, Dudley Wyman, Jane Rogers, John Sulston, Rachael Ainscough, Stephan Beck, David Bentley, John Burton, Christopher Clee, Nigel Carter, Alan Coulson, Rebecca Deadman, Panos Deloukas, Andrew Dunham, Ian Dunham, Richard Durbin, Lisa French, Darren Grafham, Simon Gregory, Tim Hubbard, Sean Humphray, Adrienne Hunt, Matthew Jones, Christine Lloyd, Amanda McMurray, Lucy Matthews, Simon Mercer, Sarah Milne, James C. Mullikin, Andrew Mungall, Robert Plumb, Mark Ross, Ratna Shownkeen, Sarah Sims, Robert H. Waterston, Richard K. Wilson, LaDeana W. Hillier, John D. McPherson, Marco A. Marra, Elaine R. Mardis, Lucinda A. Fulton, Asif T. Chinwalla, Kymberlie H. Pepin, Warren R. Gish, Stephanie L. Chissoe, Michael C. Wendl, Kim D. Delehaunty, Tracie L. Miner, Andrew Delehaunty, Jason B. Kramer, Lisa L. Cook, Robert S. Fulton, Douglas L. Johnson, Patrick J. Minx, Sandra W. Clifton, Trevor Hawkins, Elbert Branscomb, Paul Predki, Paul Richardson, Sarah Wenning, Tom Slezak, Norman Doggett, Jan-Fang Cheng, Anne Olsen, Susan Lucas, Christopher Elkin, Edward Uberbacher, Marvin Frazier, Richard A.

Gibbs, Donna M. Muzny, Steven E. Scherer, John B. Bouck, Erica J. Sodergren, Kim C. Worley, Catherine M. Rives, James H. Gorrell, Michael L. Metzker, Susan L. Naylor, Raju S. Kucherlapati, David L. Nelson, George M. Weinstock, Yoshiyuki Sakaki, Asao Fujiyama, Masahira Hattori, Tetsushi Yada, Atsushi Toyoda, Takehiko Itoh, Chiharu Kawagoe, Hidemi Watanabe, Yasushi Totoki, Todd Taylor, Jean Weissenbach, Roland Heilig, William Saurin, Francois Artiguenave, Philippe Brottier, Thomas Bruls, Eric Pelletier, Catherine Robert, Patrick Wincker, André Rosenthal, Matthias Platzer, Gerald Nyakatura, Stefan Taudien, Andreas Rump, Douglas R. Smith, Lynn Doucette-Stamm, Marc Rubenfield, Keith Weinstock, Hong Mei Lee, JoAnn Dubois, Huanming Yang, Jun Yu, Jian Wang, Guyang Huang, Jun Gu, Leroy Hood, Lee Rowen, Anup Madan, Shizen Qin, Ronald W. Davis, Nancy A. Federspiel, A. Pia Abola, Michael J. Proctor, Bruce A. Roe, Feng Chen, Huaqin Pan, Juliane Ramser, Hans Lehrach, Richard Reinhardt, W. Richard McCombie, Melissa de la Bastide, Neilay Dedhia, Helmut Blöcker, Klaus Hornischer, Gabriele Nordsiek, Richa Agarwala, L. Aravind, Jeffrey A. Bailey, Alex Bateman, Serafim Batzoglou, Ewan Birney, Peer Bork, Daniel G. Brown, Christopher B. Burge, Lorenzo Cerutti, Hsiu-Chuan Chen, Deanna Church, Michele Clamp, Richard R. Copley, Tobias Doerks, Sean R. Eddy, Evan E. Eichler, Terrence S. Furey, James Galagan, James G. R. Gilbert, Cyrus Harmon, Yoshihide Hayashizaki, David Haussler, Henning Hermjakob, Karsten Hokamp, Wonhee Jang, L. Steven Johnson, Thomas A. Jones, Simon Kasif, Arek Kasprzyk, Scot Kennedy, W. James Kent, Paul Kitts, Eugene V. Koonin, Ian Korf, David Kulp, Doron Lancet, Todd M. Lowe, Aoife McLysaght, Tarjei Mikkelsen, John V. Moran, Nicola Mulder, Victor J. Pollara, Chris P. Ponting, Greg Schuler, Jörg Schultz, Guy Slater, Arian F. A. Smit, Elia Stupka, Joseph Szustakowki, Danielle Thierry-Mieg, Jean Thierry-Mieg, Lukas Wagner, John Wallis, Raymond Wheeler, Alan Williams, Yuri I. Wolf, Kenneth H. Wolfe, Shiao-Pyng Yang, Ru-Fang Yeh, Francis Collins, Mark S. Guyer, Jane Peterson, Adam Felsenfeld, Kris A. Wetterstrand, Richard M. Myers, Jeremy Schmutz, Mark Dickson, Jane Grimwood, David R. Cox, Maynard V. Olson, Rajinder Kaul, Christopher Raymond, Nobuyoshi Shimizu, Kazuhiko Kawasaki, Shinsei Minoshima, Glen A. Evans, Maria Athanasiou, Roger Schultz, Aristides Patrinos, and Michael J. Morgan. Initial sequencing and analysis of the human genome. *Nature*, 409:860 EP –, Feb 2001.

- [33] Danny D’Agostino, Andrea Serani, Emilio F Campana, and Matteo Diez. Deep autoencoder for off-line design-space dimensionality reduction in shape optimization. In *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1648,

2018.

- [34] Jessica Davies and Fahiem Bacchus. Solving MAXSAT by solving a sequence of simpler SAT instances. In *Principles and Practice of Constraint Programming - CP 2011 - 17th International Conference, CP 2011, Perugia, Italy, September 12-16, 2011. Proceedings*, pages 225–239, 2011.
- [35] Jessica Davies and Fahiem Bacchus. Exploiting the power of mip solvers in maxsat. In *International conference on theory and applications of satisfiability testing*, pages 166–181. Springer, 2013.
- [36] Jessica Davies and Fahiem Bacchus. Postponing optimization to speed up MAXSAT solving. In *Principles and Practice of Constraint Programming - 19th International Conference, CP 2013, Uppsala, Sweden, September 16-20, 2013. Proceedings*, pages 247–262, 2013.
- [37] Erica A.K. DePasquale, Daniel J. Schnell, Pieter-Jan Van Camp, Íñigo Valiente-Alandí, Burns C. Blaxall, H. Leighton Grimes, Harinder Singh, and Nathan Salomonis. Doubletdecon: Deconvoluting doublets from single-cell rna-sequencing data. *Cell Reports*, 29(6):1718–1727.e8, 2019.
- [38] Zach DiNardo, Kiran Tomlinson, Anna Ritz, and Layla Oesper. Distance measures for tumor evolutionary trees. *Bioinformatics*, 11 2019. btz869.
- [39] Annette J Dobson. Comparing the shapes of trees. In *Combinatorial Mathematics III*, pages 95–100. Springer, 1975.
- [40] Riccardo Dondi, Nadia El-Mabrouk, and Krister M. Swenson. Gene tree correction for reconciliation and species tree inference: Complexity and algorithms. *Journal of Discrete Algorithms*, 25:51–65, 2014. 23rd Annual Symposium on Combinatorial Pattern Matching.
- [41] Bartłomiej Dudek and Paweł Gawrychowski. Computing quartet distance is equivalent to counting 4-cycles. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 733–743, 2019.
- [42] Peter Eirew, Adi Steif, Jaswinder Khattra, Gavin Ha, Damian Yap, Hossein Farahani, Karen Gelmon, Stephen Chia, Colin Mar, Adrian Wan, et al. Dynamics of genomic clones in breast cancer patient xenografts at single-cell resolution. *Nature*, 518(7539):422–426, 2015.
- [43] Mohammed El-Kebir. SPhyR: tumor phylogeny estimation from single-cell sequencing data under loss and error. *Bioinformatics*, 34(17):i671–i679, 09 2018.

- [44] Mohammed El-Kebir, Layla Oesper, Hannah Acheson-Field, and Benjamin J. Raphael. Reconstruction of clonal trees and tumor composition from multi-sample sequencing data. *Bioinformatics*, 31(12):i62–i70, Jun 2015. 26072510[pmid].
- [45] Mohammed El-Kebir, Gryte Satas, Layla Oesper, and Benjamin J. Raphael. Inferring the mutational history of a tumor using multi-state perfect phylogeny mixtures. *Cell Systems*, 3(1):43–53, 2018/01/25 2016.
- [46] Mohammed El-Kebir, Gryte Satas, Layla Oesper, and Benjamin J. Raphael. Inferring the mutational history of a tumor using multi-state perfect phylogeny mixtures. *Cell systems*, 3(1):43–53, 2016.
- [47] J.S. Farris. Phylogenetic analysis under Dollo’s law. *Systematic Biology*, 26(1):77–88, Mar 1977.
- [48] E. B. Fowlkes and C. L. Mallows. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, 78(383):553–569, 1983.
- [49] Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.
- [50] Charles Gawad, Winston Koh, and Stephen R. Quake. Dissecting the clonal origins of childhood acute lymphoblastic leukemia by single-cell genomics. *Proceedings of the National Academy of Sciences*, 111(50):17947–17952, 2014.
- [51] Marco Gerlinger, Stuart Horswell, James Larkin, Andrew J. Rowan, Max P. Salm, Ignacio Varela, Rosalie Fisher, Nicholas McGranahan, Nicholas Matthews, Claudio R. Santos, Pierre Martinez, Benjamin Phillimore, Sharmin Begum, Adam Rabinowitz, Bradley Spencer-Dene, Sakshi Gulati, Paul A. Bates, Gordon Stamp, Lisa Pickering, Martin Gore, David L. Nicol, Steven Hazell, P. Andrew Futreal, Angus Stewart, and Charles Swanton. Genomic architecture and evolution of clear cell renal cell carcinomas defined by multiregion sequencing. *Nature Genetics*, 46(3):225–233, Feb 2014.
- [52] Leslie Ann Goldberg, Paul W. Goldberg, Cynthia A. Phillips, Elizabeth Sweedyk, and Tandy Warnow. Minimizing phylogenetic number to find good evolutionary trees. *DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 71(1):111–136, December 1996.
- [53] Kiya Govek, Camden Sikes, and Layla Oesper. A consensus approach to infer tumor evolutionary histories. In *Proceedings of the 2018 ACM*

*International Conference on Bioinformatics, Computational Biology, and Health Informatics*, BCB '18, page 63–72, New York, NY, USA, 2018. Association for Computing Machinery.

- [54] Kiya Govek, Camden Sikes, Yangqiaoyu Zhou, and Layla Oesper. Graphyc: Using consensus to infer tumor evolution. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pages 1–1, 2020.
- [55] Dan Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks*, 21:19–28, 1991.
- [56] Dan Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks*, 21(1):19–28, 1991.
- [57] Dan Gusfield. *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge University Press, Cambridge, 1997.
- [58] Dan Gusfield. Persistent phylogeny: a galled-tree and integer linear programming approach. In *Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics*, pages 443–451. ACM, 2015.
- [59] Dan Gusfield, Yelena Frid, and Dan Brown. Integer programming formulations and computations solving phylogenetic and population genetic problems with missing or genotypic data. In Guohui Lin, editor, *Computing and Combinatorics: 13th Annual International Conference, COCOON 2007, Banff, Canada, July 16-19, 2007. Proceedings*, pages 51–64. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
- [60] Iman Hajirasouliha, Ahmad Mahmoody, and Benjamin J. Raphael. A combinatorial approach for analyzing intra-tumor heterogeneity from high-throughput sequencing data. *Bioinformatics*, 30(12):i78–i86, June 2014. btu284[PII].
- [61] Iman Hajirasouliha, Ahmad Mahmoody, and Benjamin J Raphael. A combinatorial approach for analyzing intra-tumor heterogeneity from high-throughput sequencing data. *Bioinformatics*, 30(12):i78–i86, 2014.
- [62] Iman Hajirasouliha and Benjamin J. Raphael. *Reconstructing Mutational History in Multiply Sampled Tumors Using Perfect Phylogeny Mixtures*, pages 354–367. Lecture Notes in Computer Science. Springer Nature, 2014.
- [63] D.J. Hand. *Discrimination and Classification*. John Wiley & Sons, 1981.

- [64] Dan He, Arthur Choi, Knot Pipatsrisawat, Adnan Darwiche, and Eleazar Eskin. Optimal algorithms for haplotype assembly from whole-genome sequence data. *Bioinformatics*, 26(12):i183–i190, 2010.
- [65] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.
- [66] Volker Hovestadt, Kyle S. Smith, Laure Bihannic, Mariella G. Filbin, McKenzie L. Shaw, Alicia Baumgartner, John C. DeWitt, Andrew Groves, Lisa Mayr, Hannah R. Weisman, Alyssa R. Richman, Marni E. Shore, Liliana Goumnerova, Celeste Rosencrance, Robert A. Carter, Timothy N. Phoenix, Jennifer L. Hadley, Yiai Tong, Jim Houston, Richard A. Ashmun, Michael DeCuypere, Tanvi Sharma, Diane Flasch, Antonina Silkov, Keith L. Ligon, Scott L. Pomeroy, Miguel N. Rivera, Orit Rozenblatt-Rosen, Jessica M. Rusert, Robert J. Wechsler-Reya, Xiao-Nan Li, Andreas Peyrl, Johannes Gojo, Dominik Kirchofer, Daniela Lötsch, Thomas Czech, Christian Dorfer, Christine Haberler, Rene Geyeregger, Angela Halfmann, Charles Gawad, John Easton, Stefan M. Pfister, Aviv Regev, Amar Gajjar, Brent A. Orr, Irene Slavic, Giles W. Robinson, Bradley E. Bernstein, Mario L. Suvà, and Paul A. Northcott. Resolving medulloblastoma cellular architecture by single-cell genomics. *Nature*, 572(7767):74–79, Aug 2019.
- [67] Qiwen Hu and Casey S Greene. Parameter tuning is a key part of dimensionality reduction via deep variational autoencoders for single cell rna transcriptomics. In *BIOCOMPUTING 2019: Proceedings of the Pacific Symposium*, pages 362–373. World Scientific, 2018.
- [68] Z. Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In *the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 1–8, 1997.
- [69] Z. Huang. Extensions to the k-modes algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2(3):283–304, 1998.
- [70] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, Dec 1985.
- [71] Laura A. Hug, Brett J. Baker, Karthik Anantharaman, Christopher T. Brown, Alexander J. Probst, Cindy J. Castelle, Cristina N. Butterfield, Alex W. Hermsdorf, Yuki Amano, Kotaro Ise, Yohey Suzuki, Natasha Dudek, David A. Relman, Kari M. Finstad, Ronald Amundson, Brian C. Thomas, and Jillian F. Banfield. A new view of the tree of life. *Nature Microbiology*, 1:16048 EP –, Apr 2016. Letter.



- [72] Edin Husić, Xinyue Li, Ademir Hujdurović, Miika Mehine, Romeo Rizzi, Veli Mäkinen, Martin Milanič, and Alexandru I Tomescu. Mipup: minimum perfect unmixed phylogenies for multi-sampled tumors via branchings and ilp. *Bioinformatics*, 35(5):769–777, 2019.
- [73] Akira Imakura, Momo Matsuda, Xiucan Ye, and Tetsuya Sakurai. Complex moment-based supervised eigenmap for dimensionality reduction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3910–3918, 2019.
- [74] Katharina Jahn, Niko Beerenwinkel, and Louxin Zhang. The Bourque distances for mutation trees of cancers. *Algorithms for Molecular Biology*, 16(1):9, 2021.
- [75] Katharina Jahn, Jack Kuipers, and Niko Beerenwinkel. Tree inference for single-cell data. *Genome Biology*, 17(1):86, May 2016.
- [76] Jesper Jansson and Ramesh Rajaby. A more practical algorithm for the rooted triplet distance. *Journal of Computational Biology*, 24(2):106–126, 2017. PMID: 27983874.
- [77] Wei Jiao, Shankar Vembu, Amit G. Deshwar, Lincoln Stein, and Quaid Morris. Inferring clonal evolution of tumors from single nucleotide somatic mutations. *BMC Bioinformatics*, 15(1):35, Feb 2014.
- [78] Joe H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.
- [79] Nikolai Karpov, Salem Malikic, Md Khaledur Rahman, and S. Cenk Sahinalp. A multi-labeled tree dissimilarity measure for comparing "clonal trees" of tumor progression. *Algorithms for Molecular Biology*, 14(1):17, 2019.
- [80] Peter V. Kharchenko. The triumphs and limitations of computational methods for scRNA-seq. *Nature Methods*, pages 1–10, 2021.
- [81] Motoo Kimura. The number of heterozygous nucleotide sites maintained in a finite population due to steady flux of mutations. *Genetics*, 61(4):893–903, 1969.
- [82] S. Kirkpatrick, C. D. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science*, 4598(220):671–680, 1983.
- [83] W. Klossgen and J.M. Zytkow. Knowledge discovery in databases terminology. In *Advances in Knowledge Discovery and Data Mining*, pages 573–592. AAAI Press/The MIT Press, 1996.

- [84] KModes. K-modes clustering.
- [85] M. Kordestani, A. Alkhateeb, I. Rezaeian, L. Rueda, and M. Saif. A new clustering method using wavelet based probability density functions for identifying patterns in time-series data. *IEEE EMBS International Student Conference*, pp, pages 1–4, 2016.
- [86] Johannes Köster and Sven Rahmann. Snakemake - a scalable bioinformatics workflow engine. *Bioinformatics*, 2012.
- [87] Jack Kuipers, Katharina Jahn, Benjamin J. Raphael, and Niko Beerenwinkel. Single-cell sequencing data reveal widespread recurrence and loss of mutational hits in the life histories of tumors. *Genome Research*, 27:1885–1894, October 2017.
- [88] L Ding L, TJ Ley, DE Larson, CA Miller, DC Koboldt, JS Welch, JK Ritchey, MA Young, T Lamprecht, MD McLellan, et al. Clonal evolution in relapsed acute myeloid leukaemia revealed by whole-genome sequencing. *Nature*, 481:506–10, 2012.
- [89] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, Richard Durbin, and 1000 Genome Project Data Processing Subgroup. The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079, 2009.
- [90] Yun Liu, Rui Zhang, Feiping Nie, Xuelong Li, and Chris Ding. Supervised dimensionality reduction methods via recursive regression. *IEEE transactions on neural networks and learning systems*, 31(9):3269–3279, 2019.
- [91] Inês Lynce and João Marques-Silva. Efficient haplotype inference with boolean satisfiability. In *Proceedings of the AAAI*, 2006.
- [92] Salem Malikic, Katharina Jahn, Jack Kuipers, S. Cenk Sahinalp, and Niko Beerenwinkel. Integrative inference of subclonal tumour evolution from single-cell and bulk sequencing data. *Nature Communications*, 10(1):2750, 2019.
- [93] Salem Malikic, Andrew W. McPherson, Nilgun Donmez, and Cenk S. Sahinalp. Clonality inference in multiple tumor samples using phylogeny. *Bioinformatics*, 31(9):1349–1356, 2015.
- [94] Salem Malikic, Farid Rashidi Mehrabadi, Simone Ciccolella, Md. Khaledur Rahman, Camir Ricketts, Ehsan Haghshenas, Daniel Seidman, Faraz Hach, Iman Hajirasouliha, and S. Cenk Sahinalp.

- Phiscs: a combinatorial approach for subperfect tumor phylogeny reconstruction via integrative use of single-cell and bulk sequencing data. *Genome Research*, 29(11):1860–1877, 2019.
- [95] Francesco Marass, Florent Mouliere, Ke Yuan, Nitzan Rosenfeld, and Florian Markowetz. A phylogenetic latent feature model for clonal deconvolution. *Ann. Appl. Stat.*, 10(4):2377–2404, 12 2016.
- [96] Donald W Marquardt and Ronald D Snee. Ridge regression in practice. *The American Statistician*, 29(1):3–20, 1975.
- [97] J.B. McQueen. Some methods for classification and analysis of multivariate observations. In *the 5th Berkely Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.
- [98] P. Medvedev, E. Scott, B. Kakaradov, and P. Pevzner. Error correction of high-throughput sequencing datasets with non-uniform coverage. *Bioinformatics*, 27(13):137–41, 2011.
- [99] Andrew Melnyk et al. *Clustering based identification of SARS-CoV-2 subtypes*. Springer, LNBI post-proceedings of ICCABS to appear, 2020.
- [100] A. Sorana Morrissy, Livia Garzia, David J. H. Shih, et al. Divergent clonal selection dominates medulloblastoma at recurrence. *Nature*, 529(7586):351–357, 01 2015.
- [101] Jost Neigenfind, Gabor Gyetvai, Rico Basekow, Svenja Diehl, Ute Achenbach, Christiane Gebhardt, Joachim Selbig, and Birgit Kersten. Haplotype inference from unphased snp data in heterozygous polyploids based on sat. *BMC genomics*, 9:356, 2008.
- [102] Sergey I. Nikolenko, Anton I. Korobeynikov, and Max A. Alekseyev. Bayeshammer: Bayesian clustering for error correction in single-cell sequencing. *BMC Genomics*, 14:1, 2013.
- [103] Peter C. Nowell. The clonal evolution of tumor cell populations. *Science*, 194(4260):23–28, October 1976.
- [104] Laxmi Parida and Esko Ukkonen, editors. *18th International Workshop on Algorithms in Bioinformatics, WABI 2018, August 20-22, 2018, Helsinki, Finland*, volume 113 of *LIPICs*, Dagstuhl, Germany, 2018. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.
- [105] I. Pe’er, T. Pupko, R. Shamir, and R. Sharan. Incomplete directed perfect phylogeny. *SIAM Journal on Computing*, 33(3):590–607, Jan 2004.

- [106] Marsela Polic, Ivona Krajacic, Nathan Lepora, and Matko Orsag. Convolutional autoencoder for feature extraction in tactile sensing. *IEEE Robotics and Automation Letters*, 4(4):3671–3678, 2019.
- [107] Victoria Popic, Raheleh Salari, Iman Hajirasouliha, Dorna Kashef-Haghighi, Robert B West, and Serafim Batzoglou. Fast and scalable inference of multi-sample cancer lineages. *Genome biology*, 16(1):91, May 2015. 647[PII].
- [108] Mark A Ragan. Phylogenetic inference based on matrix representation of trees. *Molecular phylogenetics and evolution*, 1(1):53–58, 1992.
- [109] Madhumitha Ramamurthy, Y Harold Robinson, Shanmuganathan Vimal, and Annamalai Suresh. Auto encoder based dimensionality reduction and classification using convolutional neural networks for hyperspectral images. *Microprocessors and Microsystems*, 79:103280, 2020.
- [110] Daniele Ramazzotti, Alex Graudenzi, Luca De Sano, Marco Antoniotti, and Giulio Caravagna. Learning mutational graphs of individual tumor evolution from multi-sample sequencing data. *bioRxiv*, 2017.
- [111] I. Rogozin, Y. Wolf, V. Babenko, and E. Koonin. *Dollo parsimony and the reconstruction of genome evolution*. Oxford University Press, 2006.
- [112] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, 2007.
- [113] Edith M. Ross and Florian Markowetz. Onconem: inferring tumor evolution from single-cell sequencing data. *Genome Biology*, 17(1):69, Apr 2016.
- [114] Andrew Roth, Andrew McPherson, Emma Laks, Justina Biele, Damian Yap, Adrian Wan, Maia A Smith, Cydney B Nielsen, Jessica N McAlpine, Samuel Aparicio, Alexandre Bouchard-Côté, and Sohrab P Shah. Clonal genotype and population structure inference from single-cell tumor sequencing. *Nature Methods*, 13(573):573–576, Jul 2016. Brief Communication.
- [115] Sohrab Salehi, Adi Steif, Andrew Roth, Samuel Aparicio, Alexandre Bouchard-Côté, and Sohrab P. Shah. ddclone: joint statistical inference of clonal populations from single cell and bulk tumour sequencing data. *Genome Biology*, 18(1):44, Mar 2017.

- [116] Gryte Satas and Benjamin J. Raphael. Tumor phylogeny inference using tree-constrained importance sampling. *Bioinformatics*, 33(14):i152–i160, 2017.
- [117] SciKit-Learn. Affinity propagation clustering.
- [118] SciKit-Learn. Agglomerative clustering.
- [119] SciKit-Learn. Birch clustering.
- [120] SciKit-Learn. K-means clustering.
- [121] SciKit-Learn. Spectral clustering.
- [122] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. Technical report, 2000.
- [123] Jochen Singer, Jack Kuipers, Katharina Jahn, and Niko Beerenwinkel. Single-cell mutation identification via phylogenetic inference. *Nature Communications*, 9(1):5144, 2018.
- [124] Michael Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification*, 9(1):91–116, 1992.
- [125] F. Strino, F. Parisi, M. Micsinai, and Y. Kluger. TrAp: a tree approach for fingerprinting subclonal tumor composition. *Nucleic Acids Research*, 41(17):e165, Sep 2013. gkt641[PII].
- [126] Itay Tirosh, Andrew S. Venteicher, Christine Hebert, Leah E. Escalante, Anoop P. Patel, Keren Yizhak, Jonathan M. Fisher, Christopher Rodman, Christopher Mount, Mariella G. Filbin, Cyril Nefitel, Niyati Desai, Jackson Nyman, Benjamin Izar, Christina C. Luo, Joshua M. Francis, Aanand A. Patel, Maristela L. Onozato, Nicolo Riggi, Kenneth J. Livak, Dave Gennert, Rahul Satija, Brian V. Nahed, William T. Curry, Robert L. Martuza, Ravindra Mylvaganam, A. John Iafrate, Matthew P. Frosch, Todd R. Golub, Miguel N. Rivera, Gad Getz, Orit Rozenblatt-Rosen, Daniel P. Cahill, Michelle Monje, Bradley E. Bernstein, David N. Louis, Aviv Regev, and Mario L. Suvà. Single-cell rna-seq supports a developmental hierarchy in human oligodendroglioma. *Nature*, 539:309 EP –, Nov 2016.
- [127] Hosein Toosi, Ali Moeini, and Iman Hajirasouliha. BAMSE: Bayesian model selection for tumor phylogeny inference among multiple samples. *BMC Bioinformatics*, 20(11):282, 2019.
- [128] Gianluca Della Vedova, Murray Patterson, Raffaella Rizzi, and Mauricio Soto. Character-based phylogeny construction and its application to tumor evolution. In *Unveiling Dynamics and Complexity: 13th*

*Conference on Computability in Europe (CiE)*, volume 10307 of *Lecture Notes in Computer Science*, pages 3–13, 2017.

- [129] Andrew S. Venteicher, Itay Tirosh, Christine Hebert, Keren Yizhak, Cyril Neftel, Mariella G. Filbin, Volker Hovestadt, Leah E. Escalante, McKenzie L. Shaw, Christopher Rodman, Shawn M. Gillespie, Danielle Dionne, Christina C. Luo, Hiranmayi Ravichandran, Ravindra Mylvaganam, Christopher Mount, Maristela L. Onozato, Brian V. Nahed, Hiroaki Wakimoto, William T. Curry, A. John Iafrate, Miguel N. Rivera, Matthew P. Frosch, Todd R. Golub, Priscilla K. Brastianos, Gad Getz, Anoop P. Patel, Michelle Monje, Daniel P. Cahill, Orit Rozenblatt-Rosen, David N. Louis, Bradley E. Bernstein, Aviv Regev, and Mario L. Suvà. Decoupling genetics, lineages, and microenvironment in idh-mutant gliomas by single-cell rna-seq. *Science*, 355(6332), 2017.
- [130] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4), 2007.
- [131] Jiguang Wang, Emanuela Cazzato, Erik Ladewig, et al. Clonal evolution of glioblastoma under therapy. *Nature Genetics*, 48(7):768–776, 06 2016.
- [132] Jing Wang, Haibo He, and Danil V Prokhorov. A folded neural network autoencoder for dimensionality reduction. *Procedia Computer Science*, 13:120–127, 2012.
- [133] Wei Wang, Yan Huang, Yizhou Wang, and Liang Wang. Generalized autoencoder: A neural network framework for dimensionality reduction. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 490–497, 2014.
- [134] Yasi Wang, Hongxun Yao, and Sicheng Zhao. Auto-encoder based dimensionality reduction. *Neurocomputing*, 184:232–242, 2016.
- [135] Yufeng Wu. Accurate and efficient cell lineage tree inference from noisy single cell data: the maximum likelihood perfect phylogeny approach. *Bioinformatics*, 36(3):742–750, 08 2019.
- [136] Ke Yuan, Thomas Sakoparnig, Florian Markowetz, and Niko Beerenwinkel. Bitphylogeny: a probabilistic framework for reconstructing intra-tumor phylogenies. *Genome Biology*, 16(1):36, Feb 2015.
- [137] Hamim Zafar, Nicholas Navin, Ken Chen, and Luay Nakhleh. Siclon-eft: Bayesian inference of population structure, genotype, and phylogeny of tumor clones from single-cell genome sequencing data. *Genome Research*, 2019.

- [138] Hamim Zafar, Anthony Tzen, Nicholas Navin, Ken Chen, and Luay Nakhleh. Sifit: inferring tumor trees from single-cell sequencing data under finite-sites models. *Genome Biology*, 18(1):178, Sep 2017.
- [139] Hamim Zafar, Yong Wang, Luay Nakhleh, Nicholas Navin, and Ken Chen. Monovar: single-nucleotide variant detection in single cells. *Nature Methods*, 13:505 EP –, Apr 2016.
- [140] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: An efficient data clustering method for very large databases. *SIGMOD Rec.*, 25(2):103–114, June 1996.
- [141] Zhihua Zhang, Guang Dai, Congfu Xu, and Michael I Jordan. Regularized discriminant analysis, ridge regression and beyond. *The Journal of Machine Learning Research*, 11:2199–2228, 2010.