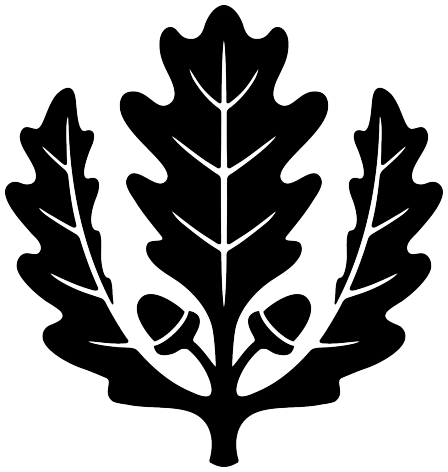
SCarborSNV

Efficient Phylogeny-aware Single Nucleotide Variant Detection for Single Cells

Christopher Oldham

A thesis presented for honors
in the degree of
Bachelor of Science in Engineering



Department of Computer Science and Engineering
School of Engineering
University of Connecticut
United States of America
August 2019

Abstract

Ongoing somatic mutations during cancer development lead to genetically distinct subclonal populations of cells within a tumor, each with a distinct subset of acquired mutations. These subclonal populations diverge genetically as new mutations occur and are subject to Darwinian selection pressures. This leads to a complex intra-tumor heterogeneity, the subclonal architecture of which is important for understanding cancer evolution and developing individualized therapies. Some approaches have used bulk DNA sequencing coupled with advanced clustering techniques to attempt to tease out this structure. Recent advances in single-cell DNA sequencing (SCS), however, have allowed new computational approaches such as Monovar and SCIΦ to examine this heterogeneity directly, despite the inherent low quality of the SCS data. We here present a new probabilistic algorithm, SCarborSNV, which will more efficiently call point mutations using aligned SCS sequencing data from a sample of multiple cells. After calling candidate variant loci using a detailed prior, SCarborSNV uses a neighbor joining algorithm to reconstruct a phylogeny which is used to genotype individual cells. We compare SCarborSNV with existing methods on simulated and real data and show that SCarborSNV performs competitively in calling single cell genotypes. SCarborSNV has asymptotically quadratic time complexity in the number of cells compared with cubic complexity for state-of-the-art tools, potentially allowing studies on much larger SCS datasets.

1 Introduction

Inferring the various genetic mutations present in a sample of phylogenetically related single cells is a complicated task, especially in light of the low quality of data generated by SCS methods. As such, previous successful methods have pooled information from across all the cells in a sample to more accurately distinguish between bona fide mutations and noise. The commonly used Monovar algorithm pools the information from across cells by marginalizing on the variable total alternate allele count across all cells at each locus, using statistical methods derived from population genetics models [1]. A more recent approach, SciΦ, has found further success by leveraging the fact that the sampled cells are phylogenetically related. By first gaining some understanding of this ancestral relationship, SciΦ has been able to improve calling accuracy with similar asymptotic complexity to Monovar [2].

The SCarborSNV algorithm presented here intends to follow on in the direction pioneered by SciΦ, using phylogenetic inference to distinguish between real mutations and SCS noise. SCarborSNV shows similar calling accuracy to state-of-the-art tools on simulated datasets, with asymptotically improved time complexity in the number of cells sampled, from $O(m^3)$ to $O(m^2)$.

1.1 Cells, mutation and intra-tumor heterogeneity

While individual organisms are mostly genetically homogeneous, mutagenic factors can cause the genotypes of cells within an organism to diverge from the germline. These somatic mutations are often inconsequential, but when they cause a cell to gain "an autonomous will to divide, this aberrant uncontrolled cell division [can create] masses of tissue (tumors) that invade organs and destroy normal tissue." [5]. Such a collection of rapidly dividing cells will share all the genetic mutations of their ancestors, including both germline mutations and the set of somatic mutations that caused the expansion. In the context of tumor evolution, these mutations common to all cells in a tumor are known as clonal, truncal or public mutations. The set of all cells that contain clonal mutations is known as the clone. As the cells continue to divide rapidly, some cells in the clone will undergo further somatic mutations, known as subclonal mutations. Any subset of tumor cells descended from such a further mutated cell (all containing the subclonal mutation) are referred to as a subclone [6–8].

After the tumor has had sufficient time to develop further mutations, the genetic landscape of the cells within the tumor will be one of hierarchical subclones; the mutations within any given cell will correspond to its specific ancestry of subclonal mutations within the tumor. At the time of biopsy, there are often many different subclones, each with a unique ancestry and thus a unique set of clonal and subclonal mutations. This genetic variation between cells in the tumor is called intra-tumor heterogeneity, and is of great importance

for understanding tumor development, metastasis and treatment [7, 8, 10, 11].

Since the genetic mutations by themselves leave few or no clues as to when in the tumor's history they occurred, resolving this subclonal structure can be challenging.

1.2 High throughput sequencing and bulk approaches

Early DNA sequencing methods could only read relatively short DNA sequences less than 1000 bases. Location-specific primers could be used to amplify such small subsequences from a longer sequence in vitro, but the process was expensive and time-consuming. On the other hand, high throughput sequencing (HTS, also called next generation sequencing or NGS) amplifies the whole original sequence many times before breaking it down into small overlapping subsequences. This multitude of short fragments is sequenced in parallel, often resulting in billions of base reads per run [13]. These short subsequences are later reassembled in silico. This massively parallel shotgun approach of HTS has proved far faster and cheaper than other methods, as well as being highly accurate (depending on how many short reads cover a given site) [14]. It is possible to reconstruct de novo sequences by piecing together the overlaps of the short reads, but for known organisms the reads can be quickly aligned to a reference genome, with some flexibility to account for mutations and variations.

The standard and until recently only approach for HTS is to extract the DNA from many thousands or millions of cells en masse before amplifying and sequencing. This method will hereafter be referred to as bulk sequencing, as the genetic material is extracted from a bulk tissue sample (as opposed to from individual cells) [15]. Bulk sequencing is very common practice, requires no specialized technology to separate cells, and often achieves a high level of calling accuracy with a high read depth (many reads mapped to each locus). Because of this bulk HTS data has been used to reconstruct tumor phylogenies, for example by ClonEvol. At each locus, the proportion of reads containing mutant reads is used as an ersatz for the proportion of mutant cells. These values can be used to discover clusters of mutations with similar frequencies. Potential orderings or phylogenies of these mutations can be evaluated based on how likely it is that each cluster descends from the others [17].

The hierarchy of subclones is difficult to accurately resolve from bulk methods, however, and rare subclones can often be lost. To resolve ITH more directly, trace cell lineages, understand rare tumor cell populations and measure mutation rates, the genotype of individual cells must be examined [18].

1.3 Single cell sequencing

As opposed to bulk sequencing, where the DNA from a large number of cells is aggregated before amplification and sequencing, single cell sequencing separates out single cells for individual examination and sequencing. After separating out individual cells, there are two methods of analyzing the genetic material therefrom: single cell RNA sequencing (scRNA-seq) and single cell DNA sequencing (SCS), each method with its own advantages and disadvantages. The more common scRNA-seq first forms co-DNA from the mRNA present in the cell transcribed from its DNA, before amplifying this co-DNA and sequencing it [19]. The advantage of scRNA-seq is that mRNA is abundant in cells, making amplification and sequencing reliable with less noise introduced from over amplification and less chance of the amplification process missing key molecules [15, 18]. scRNA-seq, however is limited by the fact that it only analyzes the transcriptome - it only provides information from the exome and is influenced by transcription regulation and cell processes such as apoptosis and mRNA processing [20, 21].

Single cell (or single nucleus) DNA sequencing, on the other hand, allows the whole genome of each cell to be amplified and sequenced directly. Since for each locus on a autosomal (diploid) chromosome only two molecules of DNA are present in each cell, however, the amplification process can lead to significant noise and erroneous artifacts in the data. Staring from only two (or one) molecules of DNA, the amplification process must go through many iterations (with more chance of error at each) and amplification errors at earlier iterations may overwhelm the amplified result. This can lead to various false-positive (FP) errors (with mutations observed from welltype DNA) and false negative (FN) errors (with mutant DNA sequenced as welltype) in the final sequencing data.

Furthermore, in the case of autosomal diploid chromosomes, a further artifact known as allelic dropout (ADO) may be introduced in the SCS process. For a heterozygous locus on such a chromosome, only one of the pair of DNA molecules may be amplified leading to the locus being read as homozygous. If neither molecule succeeds in being amplified, there may be gaps the size of whole chromosomes in the resultant data [1, 2, 15].

Despite the FP and FN errors, as well as ADO artifacts and uneven coverage, the advantage of DNA SCS to potentially sequence the entire genome of single cells has proven a useful tool for analyzing genetic mutations on a per-cell basis. Algorithms analyzing such SCS data, however, must account for this distinct error profile and treat the data with appropriate flexibility [1, 2].

1.4 Simplifications and problem scope

The biochemical processes underlying DNA mutations are incredibly complex, and modeling all the known (often context dependent) ways in which somatic DNA mutations may arise would be beyond the scope of any algorithm. For example, mutations may include insertions and deletions (indels) of various lengths; copy number variations (CNVs) where sequences of DNA are repeated a variable number of times; aneuploidy, where the normally diploid genome may become haploid or polyploid and even situations where the specific welltype nucleotides (ACGT) may make certain mutations more or less probable [1, 7–9, 22]. With so much complexity in only the known patterns of mutation, modelling and analyzing all types of DNA mutation would be currently infeasible for any bioinformatic algorithm. Some simplifications must therefore be introduced.

Following the example of state-of-the-art tools such as Monovar and SciΦ, only single nucleotide variations (SNVs, also called point mutations) are considered by SCarborSNV. For example indels are completely ignored, and CNVs are not inferred from read depth.

Since aneuploidy is so prevalent in the context of cancers (present in around 90% of breast cancers [22]) they are somewhat accounted for by SCarborSNV. SCarborSNV does account for changes in ploidy from diploid to haploid, which may result in a loss of heterozygosity (LOH). We shall model such a chromosomal aberration resulting in a LOH as a sudden switch from a heterozygous to a homozygous genotype. Furthermore, SCarborSNV considers all loci to be biallelic: only a reference and alternate allele are considered to be possible at each locus. A further area of research would be to infer and call all the various known types of DNA mutation, but it is outside the scope of this algorithm.

Finally, SCarborSNV assumes that the genome is composed of an infinite number of independent sites. This is to say that the prior probability of finding a reference or alternate allele at any locus is independent of surrounding loci, and furthermore at most one point mutation is possible at any given locus. The result is not only the biallelic assumption mentioned above, but also that each site may be analyzed in complete statistical independence of its neighbors. As such, SCarborSNV (like Monovar and SciΦ) has linear time complexity in the number of sites (denoted n). Although many mutations are in fact context dependent, and some loci are more or less vulnerable to mutation, the infinite sites assumption provides a great mathematical simplification and is therefore common among most SNV calling algorithms [9, 23, 24].

1.5 Phylogenetic inference

For a set of m sequenced cells, there are $(2m-3)!!$ possible rooted phylogenetic trees that may relate them. Furthermore, if a point mutation has occurred at a locus, the mutation could have possibly occurred on any of the $2m-1$ branches of the tree resulting in a total of $(2m-1)!!$ possible structures to consider at any

locus [3]. A truly rigorous statistical analysis of the phylogenetic relationship between cells would have to compute the likelihood of the read data marginalized on all of these $(2m-1)!!$ trees, which for any large number of single cells is practically infeasible.

Since a rigorous search through the tree space is therefore highly impractical, a heuristic search through a subspace of trees or an approximate phylogeny is required. Herein lies a major algorithmic challenge: using the knowledge of a shared ancestral relationship between cells to improve SNV calling accuracy while maintaining practical efficiency. The SciΦ algorithm uses a Markov chain Monte Carlo (MCMC) technique to heuristically search through an increasingly likely subspace of cell phylogenies and weight the final inference based on this subset of trees. The success of this method has proven the possibility of using phylogenetic inference to improve SNV calling accuracy without incurring massive algorithmic complexity. The goal of SCarborSNV is to reduce this complexity further while still leveraging the prior information of cell kinship to improve calling accuracy over phylogeny-agnostic methods. In lieu of the MCMC algorithm employed by SciΦ, SCarborSNV uses an efficient neighbor joining algorithm to infer information from an approximate phylogeny.

Neighbor-joining is an agglomerative algorithm for constructing unrooted phylogenetic trees from evolutionary distances [4]. The closest taxa (in this case cells) are joined as neighbors, and the distance between this new cluster and all other taxa is computed by assuming that the evolutionary distances are approximately additive [3]. This process is repeated, replacing neighboring taxa with a new node representing their cluster until all taxa are joined in a phylogeny.

To compute the pairwise evolutionary distance between two cells we first find the expected frequency of differing alleles between them and then convert this frequency into a more approximately additive distance resembling the Jukes-Cantor distance [3, 25].

The asymptotic time complexity of the core neighbor joining algorithm is $O(m^3)$. However, only computing the pairwise distances involves scanning through the entire genome, and the neighbor joining itself only needs to be completed once. Therefore the overall complexity of the phylogeny inference algorithm is $O(nm^2+m^3) = O(m^2(n+m))$, and since the size of n (possibly upwards of 3×10^9) dominates m (currently < 100, possibly up to the order of 10^3 in the future), for all practical purposes the phylogeny estimation in SCarborSNV is linear in n and quadratic in m. This potential for an asymptotic speedup over the state-of-the-art is the core motivation for SCarborSNV, and may hopefully expand the possibility of single cell SNV calling to samples of thousands of cells.

2 Methods

SCarborSNV's mathematical outline for calling SNVs is as follows. We begin by considering the total alternate allele count at any given locus, $\sigma \in [0, 2m]$, which represents the sum over all cells of the number of non-reference alleles at the locus [1, 29]. For a locus, we can compute the prior probability $P(\sigma)$ by considering various possible mutation histories and incorporating constants such as the expected frequency of somatic mutations (λ).

Next, we consider the likelihood of individual alternate allele count (g) on a per-cell basis: $P(d_{ij} \mid g)$, where d_{ij} is the sequencing data at site i for cell j. Using the prior distribution $P(\sigma)$ and the cell likelihoods $P(d_{ij} \mid g)$, we use a dynamic programming algorithm to determine the posterior probability $P(\sigma \mid D_i)$ for each locus in $O(nm^2)$ time [1, 29, 30]. At this stage we can exclude loci from further consideration for which $P(\sigma = 0 \mid D_i)$ is high.

Having found the posterior probability distribution for σ at each site using information from across all cells, we use this in a simple binomial model to gain locus-specific priors for individual cell genotypes. Using these priors and the cell genotype likelihoods we quickly find a posterior probability on individual cell genotypes, $P(g_{ij} \mid D_i)$ from a weighted sum over the σ distribution at each locus.

Using these posterior probability distributions of g for each cell at each locus, we define a new quantity $\overline{p_{ab}}$ that corresponds to the expected frequency with which alleles from cell a will differ from those from cell b. This expectation value $\overline{p_{ab}}$ can be used to define a genetic distance between any two cells d_{ab} inspired by the Jukes-Cantor distance [3, 25]. Computing all pairwise distances between cells can be completed in $O(nm^2)$.

An approximate phylogenetic tree of cells can be constructed using the agglomerative neighbor joining algorithm [3, 4]. Inferring final cell genotypes from this phylogeny is achieved using a two-stage "upwards-downwards" dynamic programming algorithm for each locus. In the upwards stage, each branch of the tree is assigned a probability of a mutation occurring at that branch for this locus, using the posterior genotype probabilities of the cells on the leaves. Once the branches have been assigned marginal mutation probabilities, these probabilities can be propagated downwards from the root of the tree to return final phylogeny-aware genotype probabilities for all the cells in the leaves.

2.1 Mutated site priors

We first focus on the prior probability of the total alternate allele count being σ at the locus under consideration: $P(\sum_j g_{ij} = \sigma) = P(\sigma)$. The majority of sites will not include a somatic SNV (sSNV); we say that

any site has a prior probability λ of having a somatic SNV, which is set to 0.0001 by default [1, 2].

$$P(\sigma) = P(\sigma \mid \text{sSNV})\lambda + P(\sigma \mid \neg\text{sSNV})(1 - \lambda) \tag{1}$$

Any given sample of single cells will only represent some subtree of a full cell phylogeny. As such, when considering the case where there is a sSNV at the locus we can further break down the prior into the case where the sSNV is ancestral to all sampled cells and the case where the SNV occurs within the subtree rooted at the most recent common ancestor (MRCA) of the cells sampled. We denote the case where the mutation occurs within this subtree as SNV_T.

$$P(\sigma \mid \text{sSNV}) = P(\sigma \mid \text{SNV}_T)P(\text{SNV}_T \mid \text{sSNV}) + P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T)(1 - P(\text{SNV}_T \mid \text{sSNV})) \tag{2}$$

Ploidy changes

A common and distinctive characteristic of cancer is that many tumor cells may exhibit aneuploidy or chromosomal abnormalities [8, 11, 12, 18, 22]. For simplicity, we will disregard polyploidy and focus only on the case where loci become haploid. This sort of mutation can result in lost information regarding SNVs, as a loss of heterozygosity can lead to a locus being read as homozygous [2]. Note that this is an in vivo effect, distinct from allelic dropout which occurs in vitro during DNA amplification. We will model such occurrences as a sudden switch to homozygosity, as we cannot reliably distinguish diploid homozygosity from haploidy in the genomic SCS data, which already has significantly uneven coverage and depth [1, 16]. Let H be the event that the locus under examination has become haploid, and H_T be the case that this mutation has occurred within subtree rooted at the MRCA of all sequenced cells.

$$P(\sigma \mid \text{SNV}_T) = P(\sigma \mid \text{SNV}_T, H)P(H) + P(\sigma \mid \text{SNV}_T, \neg H)(1 - P(H)) \tag{3}$$

By default we set the value of $P(H)$ to .09. In the simplest case, we consider the prior probability of an alternate allele count of σ given a point mutation occurred within the subtree and the locus remained diploid across all sampled cells. Assuming infinite sites, in such a scenario mutations would only be heterozygous.

$$P(\sigma \mid \text{SNV}_T, \neg H) = \begin{cases} \frac{2m-1}{2(m-1)} T(m, \sigma) & 0 < \sigma < m \\ 0 & \text{else} \end{cases}$$

Where $T(m, \sigma)$ is the prior developed by Singer, Kuipers et al. that assumes a mutation may occur on any branch of the sampled subtree with equal probability.

$$T(\alpha, \beta) = \frac{\binom{\alpha}{\beta}^2}{(2\beta - 1)\binom{2\alpha}{2\beta}} \tag{4}$$

Note that the prefactor of $\frac{2m-1}{2(m-1)}$ modifies the original prior so as to exclude clonal mutations in this case.

Now let us consider the case where both a sSNV and a haploid event have both occurred at a locus. Since the haploid mutation may have occurred within the subtree or ancestral to the subtree, we model these cases separately.

$$P(\sigma \mid \mathrm{SNV}_T, H) = P(\sigma \mid \mathrm{SNV}_T, H_T)P(H_T \mid H) + P(\sigma \mid \mathrm{SNV}_T, H, \neg H_T)(1 - P(H_T \mid H)) \tag{5}$$

SNV and loss of heterozygosity within the subtree

In the scenario described by the first term of Equation (5) both a point mutation and a ploidy change have occurred within the sequenced subtree. This can be split into four further subcases:

(a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

Figure 1: SNV and haploid event both within the subtree. (a) The point mutation happens before the haploid event and the mutated allele is dropped. (b) The point mutation happens before the haploid event and the reference allele is dropped. (c) The haploid event occurs before the point mutation. Since haploid cells are modelled as becoming homozygous diploids, this case leads to only even values of alternate allele count. (d) In this case the point mutation and haploid event do not occur in the same lineage. We ignore this case as the haploid event does not affect the alternate allele count.

Considering the above three cases where a ploidy change in the subtree affects the locus alternate allele count, it is twice as likely that the point mutation should occur before the haploid event (cases 1 and 2), compared with the other temporal ordering (case 3). This is because the cells before a haploid event, being diploid, have twice the chance of having a point mutation at a locus than the haploid descendants of such a ploidy change. We are here assuming that there is a constant probability of a point mutation per base pair per unit time. Therefore $P(\text{case 1 or 2}) = 2/3$ and $P(\text{case 3}) = 1/3$.

We also assume the reference and alternate alleles have an equal chance of being dropped in a loss of heterozygosity.

$$P(\text{case 1}) = P(\text{case 2}) = P(\text{case 3}) = 1/3$$

Supposing that that both a point mutation and a haploid event have occurred within the sequenced subtree at the locus in question, we now have

$$P(\sigma \mid \text{SNV}_T, H_T) = \frac{1}{3}\left[P(\sigma \mid \text{case 1}) + P(\sigma \mid \text{case 2}) + P(\sigma \mid \text{case 3})\right]$$

To examine these probabilities we will use the function $T(\alpha, \beta)$ described above, which given a subtree S with α leaves gives the probability of a mutation in S affecting β of its leaves. For case 1, the loss of heterozygosity effectively deletes all alternate alleles from the cells sharing a lineage with the haploid event, and so

$$P(\sigma \mid \text{case 1}) = \frac{2m-1}{2(m-1)} \sum_{a-h=\sigma} T(m,\alpha)T(\alpha,h) \qquad 1 \leq \alpha < m,\ 1 \leq h \leq \alpha$$

While we include one normalization constant, excluding the case of an ancestral point mutation, we do allow the case where a point mutation and a loss of heterozygosity happen on the same branch of the phylogeny. Hence the support for σ in case 1 is $[0, m)$. Since we model a haploid event as a sudden switch to homozygosity, the observed allele count for case 2 is the number of cells affected by the heterozygous point mutation (α) added to the number of cells affected by the loss of heterozygosity (h).

$$P(\sigma \mid \text{case 2}) = \frac{2m-1}{2(m-1)} \sum_{\alpha+h=\sigma} T(m,\alpha)T(\alpha,h) \qquad 1 \leq \alpha < m,\ 1 \leq h \leq \alpha$$

For case 2, the support is [2, 2m-2]. In case 3, only even allele counts can be produced as all cells carrying the point mutation are haploid, which we model as being homozygous mutated. The possible values of σ are $2 \leq \sigma \leq 2m-2$.

$$P(\sigma \mid \text{case 3}) = \frac{2m-1}{2(m-1)} \sum_{2a=\sigma} T(m,h)T(h,a) \qquad 1 \leq h < m,\ h \geq a$$

Haploid subtree

Referring back to Equation (5), we must determine the prior probability of an alternate allele count σ at a locus given that a point mutation occurred within the sequenced subtree and a haploid event occurred ancestral to the subtree. This would lead to all sampled cells being haploid at this locus, therefore allowing only even allele counts.

$$P(\sigma \mid \text{SNV}_T, H, \neg H_T) = \begin{cases} \frac{2m-1}{2(m-1)} T(m, \frac{\sigma}{2}) & 2 \mid \sigma,\ 0 < \sigma < 2m \\ 0 & \text{else} \end{cases}$$

Clonal and subclonal mutations

We have so far considered the case where sSNVs have been subclonal: they may affect some of our sampled cells and not others. There is some probability however that a sSNV at a given locus may be due to a mutation in a cell ancestral to all sampled cells. The majority of these ancestral mutations will affect all tumor cells: so-called clonal, truncal or public mutations [7, 8, 26]. If the sample of single cells is small enough, however, it could be the case that a subclonal mutation is common to all cells sampled.

$$P(\text{ancestral} \mid \text{sSNV}) = P(\text{clonal}) + P(\text{ancestral} \mid \text{subclonal})(1 - P(\text{clonal})) \tag{6}$$

To find the probability of a subclonal mutation affecting all sampled cells, we assume tumor subclones follow a neutral evolutionary model such that subclonal mutant allele frequencies follow a power law distribution [26]. Using an IID model for tumor cell sampling, the probability that all m cells are from a subclone with cellular frequency $2f$ (allelic frequency $= f$) is $(2f)^m$. Similar to Williams et al. we define a probability density function for the allelic frequency of subclonal mutations proportional to the inverse of the allelic frequency.

$$P(f) = k\left(\frac{1}{f} - 2\right) \tag{7}$$

where k is a normalization constant. We define the support of $P(f)$ for subclones as $[10^{-8}, 0.5]$, as a frequency of 10^{-8} is on the order of affecting single cells and an allelic frequency of 0.5 corresponds to clonal mutations [27]. We find a value of k by integrating $P(f)$ over this support. If a subclonal mutation affects all sampled cells, then all these cells must be from the same subclone.

$$P(\text{ancestral} \mid \text{subclonal}) = \int_f P(\text{all in subclone} \mid f)P(f)\,df = \int_{10^{-8}}^{0.5} (2f)^m k\left(\frac{1}{f} - 2\right)\,df \tag{8}$$

For large enough samples ($m > 30$) the probability that all cells were sampled from a single subclone (Equation 8) becomes negligible. Since it is only a function of m these values are pre-computed for efficiency. The empirically estimated probability that any given mutation is clonal is set at $P(\text{clonal}) = 0.51$ [7, 8, 28]. Having developed the probability that any given somatic mutation is ancestral to all sampled cells, we have also found the probability that a mutation has occurred within the sampled subtree.

$$P(\text{SNV}_T \mid \text{sSNV}) = P(H_T \mid H) = 1 - P(\text{ancestral} \mid \text{sSNV})$$

Ancestral sSNVs

Referring Equation (2) we must also determine the prior for σ in the case that the sSNV is ancestral to all cells. Without LOH this would always result in $\sigma = m$, however we must again consider haploid events both within and ancestral to the sampled subtree.

$$P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T) = P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T, H)P(H) + P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T, \neg H)(1 - P(H)) \tag{9}$$

If there is no haploid event, there will be no LOH and the ancestral sSNV will be heterozygous across all cells.

$$P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T, \neg H) = \begin{cases} 1 & \sigma = m \\ 0 & \text{else} \end{cases}$$

In the case of a haploid event, we must consider the cases where the whole subtree is haploid and when the haploid event happens within the subtree. We also consider the alternate and reference alleles to be dropped with equal probability as before.

$$\begin{aligned} P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T, H) =& (1 - P(H_T \mid H))P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T, H, \neg H_T) \\ &+ P(H_T \mid H)P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T, H_T) \end{aligned}$$

If both the somatic SNV and the haploid event are ancestral to the sequenced subtree, the cells will either all be haploid reference or all be haploid alternate with equal probability. Since we model haploid cells as homozygous diploid this results in only alternate allele counts of 0 or $2m$.

$$P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T, H, \neg H_T) = \begin{cases} \frac{1}{2} & \sigma = 0, 2m \\ 0 & \text{else} \end{cases}$$

If there is an sSNV ancestral to the sampled cells but a haploid event occurred within the sampled subtree we may have any alternate allele count from 1 to $2m-1$ depending on where in the phylogeny heterozygosity was lost and which allele was dropped.

$$P(\sigma \mid \text{sSNV}, \neg\text{SNV}_T, H_T) = \begin{cases} \frac{1}{2}T(m, \sigma - m) & \sigma > m \\ \frac{1}{2}T(m, m - \sigma) & m > \sigma \end{cases}$$

2.2 Welltype site priors

We have so far considered the prior probability of alternate allele counts at a locus given that a somatic SNV has occurred at that locus. The majority of loci, however, will be unaffected by sSNVs, although may still contain germline point mutations. Referring back to Equation (1) we must consider the prior probabilities of σ for sites without sSNVs. Note, however, that such a site may still be affected by aneuploidy.

$$P(\sigma \mid \neg\text{sSNV}) = P(\sigma \mid \neg\text{sSNV}, H)P(H) + P(\sigma \mid \neg\text{sSNV}, \neg H)(1 - P(H)) \tag{10}$$

In the case where there is no sSNV and no aneuploidy, we simply assume Hardy-Weinberg equilibrium, with a germline mutation rate of μ. We set the value of μ relatively high at 0.1 to reduce false positive errors.

$$P(\sigma \mid \neg\text{sSNV}, \neg H) = \begin{cases} \mu^2 & \sigma = 2m \\ 2\mu(1-\mu) & \sigma = m \\ (1-\mu)^2 & \sigma = 0 \\ 0 & \text{else} \end{cases}$$

Continuing to assume HWE for the germline genotype, aneuploidy will only affect the alternate allele count for a heterozygous germline genotype. Here again we assume either allele may be dropped with equal probability.

$$P(\sigma \mid \neg\text{sSNV}, H) = \begin{cases} \mu^2 + \mu(1-\mu)(1 - P(H_T \mid H)) & \sigma = 2m \\ \mu(1-\mu)P(H_T \mid H)\frac{2m-1}{2(m-1)}T(m, \sigma - m) & m < \sigma < 2m \\ 0 & \sigma = m \\ \mu(1-\mu)P(H_T \mid H)\frac{2m-1}{2(m-1)}T(m, m - \sigma) & 0 < \sigma < m \\ (1-\mu)^2 + \mu(1-\mu)(1 - P(H_T \mid H)) & \sigma = 0 \end{cases}$$

2.3 Cell genotype likelihoods

Having found the prior probability distribution for the alternate allele count across all cells at a locus, we now turn to the likelihoods of individual cell genotypes. The (unphased) genotype $g \in \{0, 1, 2\}$ for cell j at locus i is equivalent to the specific alternate allele count for that cell. At this stage of the calculation we consider all sites and cells independently.

Homozygous genotypes

For the cell likelihood $\Lambda(g) = P(d_{ij} \mid g)$ we assume the error on each read to be independent of the other reads. Hence if cell j has l reads covering locus i we have:

$$P(d_{ij} \mid g) = \prod_{k=1}^{l} P(d_{ijk} \mid g)$$

Note $d_{ij} = (\vec{r}, \vec{e})$, where $\vec{r}$ are the l reads for this cell at this locus and $\vec{e}$ are the associated l probabilities of read error. The values of $\vec{e}$ are provided by the sequencing machine in the form of Phred quality scores, which are directly proportional to the log-probability of sequencing error.

For any particular read and associated read error probability, we first marginalize the probability of observing this data on whether or not there has been an error in sequencing:

$$P(d_k \mid g) = P(r_k, se \mid g) + P(r_k, \neg se \mid g)$$

Since errors can occur during amplification or sequencing, we model the result of amplification as a latent "intermediate allele", denoted β. This random variable follows a discrete, symmetric, four-point distribution that assumes a constant probability of amplification error for any base, and equal probability of amplifying either allele in the heterozygous case [1]. The probability of amplification error is an input parameter for SCarborSNV, with a default value of 0.002. If a read is not the result of a sequencing error:

$$P(r_k, \neg se \mid g) = P(r_k \mid \neg se, g)(1 - e_k) = P(\beta_k = r_k \mid g)(1 - e_k)$$

In the case that there is a sequencing error, we similarly see:

$$P(r_k, se \mid g) = P(r_k \mid se, g)e_k$$

Furthermore:

$$P(r_k \mid se, g) = P(\beta_k \neq r_k \mid se, g)P(r_k \mid \beta_k \neq r_k, se, g)$$

We expect an error in sequencing the intermediate allele could produce any of the other three alleles with equal probability. Equivalently, we find $P(r_k \mid \beta_k \neq r_k, se, g) = 1/3$.

Since the amplification of β from the true underlying genotype is unaffected by sequencing:

$$P(\beta_k \neq r_k \mid se, g) = P(\beta_k \neq r_k \mid g) = 1 - P(\beta_k = r_k \mid g)$$

And therefore:

$$P(r_k, se \mid g) = e_k \frac{1}{3} \left[1 - P(\beta_k = r_k \mid g)\right]$$

Finally the likelihood $P(d_{ij} \mid g)$ for cell at a locus for a homozygous genotype is:

$$P(d_{ij} \mid g) = \prod_{k=1}^{l} \left[(1 - e_k) P(\beta_k = r_k \mid g) + e_k \frac{1}{3}(1 - P(\beta_k = r_k \mid g))\right] \tag{11}$$

This equation is used for the homozygous cases $g \in \{0, 2\}$. This is somewhat more simple than the heterozygous case because an allelic dropout at a homozygous locus would not have a detectable effect on the read values and is therefore not considered.

Heterozygous genotypes and allelic dropout

For the case where the underlying genotype is heterozygous, $g = 1$, we must account for the possibility of an allelic dropout (ADO) event [1, 2]. Allelic dropout is when from two homologous chromosomes in a cell only one is amplified, and the other is lost. Since each chromosome is a single molecule of DNA in the cell, this is a very frequent event in single cell DNA sequencing.

For the heterozygous case we therefore first marginalize the genotype likelihood on whether or not such an ADO event has occurred:

$$P(d_{ij} \mid g = 1) = P(d_{ij}, \text{ADO} \mid g = 1) + P(d_{ij}, \neg\text{ADO} \mid g = 1)$$

Letting P_{ADO} be the prior probability that any given locus in a cell has been affected by an allelic dropout event, this expands to:

$$P(d_{ij} \mid g = 1) = P_{ADO} P(d_{ij} \mid \text{ADO}, g = 1) + (1 - P_{ADO}) P(d_{ij} \mid \neg\text{ADO}, g = 1)$$

P_{ADO} is another initial parameter of SCarborSNV, and like in Monovar, is set by default to 0.2. This value is set quite high to reflect how common these events are expected to be; if $P_{ADO} = 0.2$ we expect that about four or more of the twenty-two homologous pairs of autosomal chromosomes in each cell will be affected by such a dropout.

If an ADO event affects a heterozygous locus, only one allele will remain after the amplification process and hence the likelihood $P(d_{ij} \mid \text{ADO}, g = 1)$ will resemble the homozygous case. We assume allelic dropout

can affect either allele with equal probability and hence:

$$P(d_{ij} \mid \text{ADO}, g = 1) = \frac{1}{2}P(d_{ij} \mid g = 0) + \frac{1}{2}P(d_{ij} \mid g = 2)$$

For the case without allelic dropout, the form of the likelihood is identical to the homozygous case:

$$P(d_{ij} \mid \neg\text{ADO}, g = 1) = \prod_{k=1}^{l} \left[(1 - e_k)P(\beta_k = r_k \mid g = 1) + e_k \frac{1}{3}(1 - P(\beta_k = r_k \mid g = 1)) \right]$$

With this final equation we can calculate the genotype likelihoods $P(d_{ij} \mid g)$ for any $g \in [0, 2]$ for every cell at every locus. These values can be then be used in a dynamic programming algorithm to calculate the likelihood of the total alternate allele count across each locus: $\Lambda(\sigma) = P(D_i \mid \sigma)$.

2.4 Alternate allele count likelihoods

Until now we have considered the genotype likelihoods of individual cells independently. In population genetics, it has been found that pooling low coverage sequencing data from multiple samples at each locus can significantly reduce false positive rates and improve SNP calling accuracy [29,30]. Following in this vein, Monovar and later SciΦ found that similar approaches could be applied to calling somatic SNVs from SCS data: pooling information from across cells instead of individuals [1,2].

To develop this dependence between cells at a locus, we seek to find a posterior probability distribution over the total alternate allele count summed over all cells at a locus. Using Bayes' formula:

$$P(\sigma \mid D_i) = \frac{P(D_i \mid \sigma)P(\sigma)}{\sum_{\sigma'=0}^{2m}[P(D_i \mid \sigma = \sigma')P(\sigma')]} \tag{12}$$

Having already found the prior probability distribution for this variable $P(\sigma)$, we are left to find the likelihoods of total alternate allele count at each locus $P(D_i \mid \sigma)$ from the individual cell genotypes calculated above.

There are various permutations of cell genotypes that may give rise to an alternate allele count of σ, so this is not a straight-forward task. While it may be possible to enumerate and sum over all these permutations, it would require the calculation of 2^m values at each locus and therefore would be impractically complex. Luckily, however, there is a dynamic programming algorithm that can calculate these likelihoods at each locus in quadratic time [1,29,30].

Let the phased genotypes of all m cells at a site be represented by $\vec{G} = (G_1, G_2, \ldots, G_m)$ such that $G_j \in [0, 1] \times [0, 1]$ is the phased genotype for cell j (0 = reference, 1 = alternate). Furthermore let the

unphased genotype vector be $\vec{g} = (g_1, g_2, \ldots, g_m)$ be such that $g_j = \|G_j\|_1 \in [0, 2]$. Our likelihood for σ can therefore be considered

$$P(D_i \mid \sigma_i) = \sum_{\vec{G}} P(D_i \mid \vec{G}) P(\vec{G} \mid \sigma_i) \tag{13}$$

We assume that all phased genotype vectors with a total alternate allele count of σ are equally probable. Across m diploid cells at a locus there are $\binom{2m}{\sigma}$ different phased genotype vectors with total alternate allele count σ, since there are $2m$ distinct positions that may be mutated. Therefore for any such $\vec{G}$:

$$P(\vec{G} \mid \sigma) = \binom{2m}{\sigma}^{-1}$$

Since we do not consider phased sequencing data, we must reproduce Equation (13) in an unphased form. To begin, we see that the likelihood $P(D_i \mid \vec{G}) = P(D_i \mid \vec{g})$ if $\vec{g}$ is the unphased vector that corresponds to $\vec{G}$, since our cell genotype likelihoods do not consider phasing.

Let $\chi(\vec{g})$ is the number of heterozygous cells in the unphased genotype vector $\vec{g}$. We see that there are 2^{χ} phased genotype vectors that correspond to any given unphased genotype vector $\vec{g}$, as there is a choice of which allele is mutated at each heterozygous cell. Using this multiplicity, we can now reproduce Equation (13) without reference to phasing.

$$P(D_i \mid \sigma_i) = \sum_{\{\vec{g} \,:\, \|\vec{g}\|_1 = \sigma\}} \frac{2^{\chi(\vec{g})}}{\binom{2m}{\sigma_i}} P(D_i \mid \vec{g}) = \sum_{\{\vec{g} \,:\, \|\vec{g}\|_1 = \sigma\}} \frac{2^{\chi(\vec{g})}}{\binom{2m}{\sigma_i}} \prod_{j=1}^{m} P(d_{ij} \mid \vec{g}_j)$$

Let the function $\delta(\vec{g}, \sigma) = 1$ if $\|\vec{g}\|_1 = \sigma$ and otherwise it evaluates to 0. We can now write the above in a more suggestive form:

$$P(D_i \mid \sigma_i) = \binom{2m}{\sigma_i}^{-1} \sum_{g_1=0}^{2} \sum_{g_2=0}^{2} \cdots \sum_{g_m=0}^{2} \delta((g_1, \ldots g_m), \sigma_i) \left[\prod_{j=1}^{m} \binom{2}{g_j} P(D_{ij} \mid g_j) \right] \tag{14}$$

Following previous approaches, we now employ a dynamic programming approach to compute these likelihoods for σ from cell genotype likelihoods [1, 2, 29, 30]. The idea behind the dynamic approach is to consider only a subset of the cells and compute the alternate allele likelihoods therefor. We then add one cell at a time to this subset. Each additional cell can only increase the alternate allele count by 0, 1 or 2 and as such each value in the current iteration needs to include at most three values from the previous iteration. Once we have added all the sampled cells to the subset we can quickly calculate the overall alternate allele count likelihoods.

Let $F(k, l)$ be the subproblem objective given by

$$F(k,l) = \begin{cases} \sum_{g_1=0}^{2} \sum_{g_2=0}^{2} \cdots \sum_{g_k=0}^{2} \delta((g_1, \ldots g_k), l) \left[\prod_{j=1}^{k} \binom{2}{g_j} P(D_{ij} \mid g_j) \right] & 0 \leq l \leq 2k \\ 0 & \text{else} \end{cases} \tag{15}$$

We can consider creating a genotype vector of length k from a vector of length $k-1$ by adding one new cell with an alternate allele count of $0, 1$ or 2. Hence our recurrence relation can be given by

$$F(k,l) = F(k-1,l)P(d_{ik} \mid g_k = 0) + 2F(k-1,l-1)P(d_{ik} \mid g_k = 1) + F(k-1,l-2)P(d_{ik} \mid g_k = 2) \tag{16}$$

Note that two possible phased genotypes correspond to the heterozygous case, hence the factor of 2 in the second term. The base case where $k = 1$ corresponds to a single cell

$$F(1,0) = P(d_{i1} \mid g_1 = 0), \;\; F(1,1) = 2P(d_{i1} \mid g_1 = 1), \;\; F(1,2) = P(d_{i1} \mid g_1 = 2)$$

The values for $F(k, l)$ are memoized in an array and the likelihood in Equation 14 can be given by

$$P(D_i \mid \sigma_i) = \frac{F(m, \sigma_i)}{\binom{2m}{\sigma_i}} \tag{17}$$

In this way we can determine the likelihood of all $0 \leq \sigma \leq 2m$. As we see in Equation (12), the normalized entrywise product of these likelihoods with the priors $P(\sigma)$ gives us a posterior probability distribution over σ for each locus.

The dynamic programming algorithm requires m iterations to arrive at the final likelihoods considering all cells. At every iteration up to $2m$ values are calculated, each in constant time. Therefore the likelihoods at each locus are calculated in $O(m^2)$ time, resulting in an overall time complexity of $O(nm^2)$.

2.5 Pairwise distances and candidate loci

The most accurate phylogenetic structure of the sampled tumor cells could be found by searching through the entire tree space and finding a tree that maximizes likelihood or posterior probability. If s candidate sites are called in the previous step there are $(2m-3)!!(2m-1)^s$ trees in the search space, making this approach infeasible and leading a previous phylogeny aware approach to adopt a more efficient Markov chain Monte Carlo (MCMC) algorithm [2]. This more efficient approach results in an overall asymptotic time complexity of $O(nm^3 \log(m))$ [2]. The Monovar algorithm has an overall asymptotic time complexity of $O(nm^3)$ [1].

Seeking a more efficient phylogeny-aware method for single nucleotide variant calling, SCarborSNV uses a

neighbor-joining algorithm to infer an approximate cell phylogeny based on estimated pairwise evolutionary distances between cells. This approach has an asymptotic complexity of $O(nm^2 + m^3)$, and so especially for data sets with many cells we expect it will yield results much faster even than Monovar.

So far at each locus we have calculated a posterior distribution over the alternate allele count: $P(\sigma \mid D_i)$. To compute pairwise distances between cells we need posterior genotype probabilities for each cells genotype, considering the pooled information from all cells at the locus.

Monovar achieves this with a second dynamic programming algorithm that jointly considers the probability of cell j having genotype g and the remaining $m-1$ cells having an alternate allele count of $\sigma - g$. Even with the increased efficiency of Monovar's second dynamic programming algorithm it still has a time complexity of $O(nm^3)$. Trading some accuracy for efficiency we instead use a simple binomial model to compute intermediate posterior cell genotype probabilities. The final SNV calling will be done after inferring phylogeny-aware genotype probabilities.

Using Bayes' formula and marginalizing on σ:

$$P(g_{ij} \mid D_i) = \sum_{\sigma=0}^{2m} \left[\frac{P(d_{ij} \mid g_{ij}) P(g_{ij} \mid \sigma)}{\sum_g P(d_{ij} \mid g) P(g \mid \sigma)} P(\sigma \mid D_i) \right] \tag{18}$$

where $P(d_{ij} \mid g)$ are simple cell genotype likelihoods. For the conditional prior on cell likelihoods, we use a simple binomial approximation that can be pre computed for all sites

$$P(g \mid \sigma) = \binom{2}{g} \left(\frac{\sigma}{2m}\right)^g \left(1 - \frac{\sigma}{2m}\right)^{2-g}$$

Next we define a pairwise value $\overline{p}_{ab}$, the expected frequency with which nucleotides differ between two cells a and b:

$$\overline{p}_{ab} = \frac{1}{2n} \sum_{i=1}^{n} \left[\sum_{|g_{ia} - g_{ib}|=2} 2P(g_{ia})P(g_{ib}) + \sum_{|g_{ia} - g_{ib}|=1} P(g_{ia})P(g_{ib}) \right]$$

Note we assume that if two cells have the same alternate allele count at a locus they have the same phased genotype at that locus. For each pair of cells this value can be computed in linear time in n, giving this step an overall time complexity of $O(nm^2)$.

Next we compute an approximately additive distance inspired by the Jukes-Cantor distance [3, 25]:

$$d_{ab} = -\frac{3}{4} \log \left(1 - \frac{4}{3} \overline{p}_{ab}\right) \tag{19}$$

Included with the biological cells is a false cell with homozygous reference genotype which is included as an outgroup to root the tree. After computing d for all pairs of cells, we implement a simple neighbor-joining

algorithm based thereupon.

At this stage also we threshold loci on $P(\sigma = 0 \mid D)$. By default, if $P(\sigma = 0 \mid D) < 0.5$ we save a locus as a candidate for later genotyping. By only considering these loci with a high probability of mutation we further increase the efficiency of this method.

2.6 Building a cell phylogeny

From the matrix of pairwise evolutionary distances d we can now build a single phylogeny to relate the m cells. We will then use this phylogeny to improve our SNV calls.

Neighbor joining is an agglomerative phylogeny building algorithm that assumes approximately additive evolutionary pairwise distances. The edge lengths of a tree are additive if and only if "the distance between any pair of leaves is the sum of the lengths of the edges on the path connecting them" [3]. Neighbor joining finds pairs of neighboring nodes (initially single cells), connects them and then replaces the two with a new node representing the newly formed cluster. The distance between the new node and all remaining nodes can be computed using the assumption of additivity.

Note that neighboring nodes are not necessarily those with the closest distance between them as this fails to join neighbors where one edge is short and the other is long [3, 4]. Instead, a pair of nodes is selected to be joined in any iteration if they minimize the pairwise metric:

$$D_{ij} = d_{ij} - r_i - r_j \tag{20}$$

where

$$r_i = \frac{1}{|L| - 2} \sum_{k \in L} d_{ik}$$

and L is the set of leaves or active nodes to be joined.

Since our approximate distances in reality are not always additive, it is possible to get negative edge lengths. In this case we set the negative distance to 0, and adjust the adjacent distance so that their sum remains unchanged [31]. The neighbor joining algorithm is therefore as described below, using similar notation as is used in [3].

After the neighbor joining algorithm is complete, the tree can be rooted using the pseudo-cell outgroup that is homozygous reference with a probability of 1.

Algorithm : Neighbor Joining. Modified from Saitou & Nei, 1987

$L \leftarrow$ {Cells, reference outgroup}
$T \leftarrow \{\}$
while $|L| > 2$ **do**
  $D \leftarrow$ Eq. (20) on all pairs in L
  $N_i, N_j \leftarrow$ nodes from L s.t. $(i,j) = \text{argmin } D_{ij}$
  $N_k \leftarrow$ new node
  for all $N_l \in L - \{N_i, N_j\}$ **do**
    $d_{kl} \leftarrow \frac{1}{2}(d_{il} + d_{jl} - d_{ij})$
  end for
  $T \leftarrow T + N_k$ with edge distances $d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$ and $d_{jk} = \frac{1}{2}(d_{ij} + r_j - r_i)$
  if $d_{ik} < 0$ **or** $d_{jk} < 0$ **then**
    Positive $d_+ \leftarrow d_+ + d_-$
    Negative $d_- \leftarrow 0$
  end if
  $L \leftarrow L - N_i - N_j + N_k$
end while
Join last two nodes with edge distance d_{ij}

2.7 Genotyping single cells

Assuming the tree T created above is approximately accurate, we now seek to infer genotypes from this phylogeny so as to overcome errors and noise associated with low coverage SCS data. We first determine weights for attaching point mutations and different types of LOH events to different edges of the tree, and then use these weights to determine genotype probabilities for each cell. The approximate phylogeny was found by combing information from across the genome, however to infer cell genotypes we shall return to considering individual loci.

SNV weights

In the simplest case, we ignore any potential loss of heterozygosity. For each edge of the tree e, we seek to answer the following question: assuming there is a point mutation at this locus, what is the probability it occurred at e?

To begin with, we compute the likelihood of all descendants of an edge in the tree having the same genotype. We say that $\pi_0(e), \pi_1(e)$ and $\pi_2(e)$ are respectively the likelihoods that all descendants of e are homozygous reference, heterozygous and homozygous alternate.

These values are taken to be

$$\pi_g(e) = \prod_{\{j : c_j \succ e\}} P(g_j = g) \tag{21}$$

where $c_j \succ e$ indicates that the j^{th} cell is below e in T. $P(g_j = g)$ here are the probabilities calculated in Equation (18). We also compute one more value, $\pi_\mu(e)$, defined as the likelihood that all descendants of e

have a genotype of either 1 or 2, that is to say they are mutant. These four values can be computed recursively at each locus in $O(m)$ time by multiplying the corresponding values from the two branches directly beneath each branch.

If an SNV occurred at edge e in T, we would expect that all descendants of e would have genotypes of 1 or 2, while all other cells would have genotype 0 (assuming infinite sites). Letting ρ be the edge ancestral to all sampled cells, it is easy to verify that this likelihood simplifies to:

$$\prod_{c_x \succ e} P(g_x = 1, 2) \prod_{c_y \not\succ e} P(g_y = 0) = \pi_\mu(e) \frac{\pi_0(\rho)}{\pi_0(e)}$$

If a mutation occurred at a particular locus, the prior probability that it occurred on branch e can be found by using the edge lengths of T: mutations occur on longer edges more frequently. We must undo the Jukes-Cantor correction that was useful for neighbor joining to find a value proportional to the expected number of SNVs along an edge. We find $\overline{p}_e = \frac{3}{4}\left(1 - e^{-\frac{4}{3}d_e}\right)$ to be such a value and therefore define the prior probability (conditioned on the existence of a mutation at this locus) of attaching a SNV to edge e to be:

$$P(S_e) = \frac{\overline{p}_e}{\sum_{e'} \overline{p}_{e'}}$$

Since $\pi_0(\rho)$ and $\sum_e \overline{p}_e$ are constant for a locus, we define a weight for attaching a SNV with no LOH to edge e as:

$$W(S_e) = \frac{\pi_\mu(e)}{\pi_0(e)} \overline{p}_e$$

which is proportional to the posterior probability of attachment at e up to a normalization constant (assuming a SNV occurred at this locus). This is because the ratio π_μ/π_0 is as a likelihood, where the 'observations' are the posterior probabilities from the last stage of the algorithm, and $\overline{p_e}$ is proportional to a prior for e, conditioned on the site being mutant and T.

When we recurse up the tree computing π_g values, we can also compute and store these weights $W(S_e)$. If we recursively sum the weights from the two children of each node, at the root this sum value will equal the total sum of weights and we can thereafter derive the probabilities:

$$P(S_e \mid SNV, T) = \frac{W(S_e)}{\sum_{e'} W(S_e)} \tag{22}$$

Weights for loss of heterozygosity

(a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

Figure 2: Different ways loss of heterozygosity can occur in a tree. The LOH in case 4 is undetectable from the SCS reads and thus ignored.

The more complicated cases involve both a SNV and a ploidy change occurring at the same locus. The four possible ways this can occur is shown in Figure 2.

For case 1, a point mutation occurred at edge e_1 before a ploidy change dropped the alternate allele at e_2. We expect that all cells $e \succ e_2$ will have genotype 0, since the alternate allele has been dropped for this subclone. Cells for which $e \succ e_1$ but $e \not\succ e_2$ will have genotype 1: these are the cells from the SNV subclone that did not experience the LOH. All other cells will have genotype 0, as they are not in the subclone of the SNV.

Similar to the simplest case, the likelihood of a SNV at e_1 followed by a dropped alternate allele at e_2 is therefore given by:

$$\prod_{c_x \succ e_2} P(g_x = 0) \times \prod_{c_y \succ e_1, \not\succ e_2} P(g_y = 1) \times \prod_{c_z \not\succ e_1, \not\succ e_2} P(g_z = 0) = \pi_0(e_2) \cdot \frac{\pi_1(e_1)}{\pi_1(e_2)} \cdot \frac{\pi_0(\rho)}{\pi_0(e_1)}$$

The prior probability of attaching the SNV at e_1 is the same as in the simple case: normalized $\overline{p}_{e_1}$. The prior probability of attaching a loss of heterozygosity at e_2, however, is subtly different. We assume that a loss of heterozygosity affecting just a single cell is far less probable than an allelic dropout event during the amplification process. Therefore we define

$$P(L_e) = \frac{\overline{p}_e}{\sum_{e' \in E - E_L} \overline{p}_{e'}}$$

where E_L is the set of all edges directly above a leaf node. This is the same as setting the prior probability of a loss of heterozygosity affecting individual cells to 0. The weight of attaching the SNV to e_1 and the LOH to $e_2 \succeq e_1$ for case 1 is:

$$W^{(1)}(S_{e_1}, L_{e_2}) = \frac{\pi_0(e_2)\pi_1(e_1)}{\pi_1(e_2)\pi_0(e_1)} \overline{p}_{e_1} \overline{p}_{e_2} \tag{23}$$

This is proportional to a joint posterior probability, conditioned on several assumptions: both a SNV and LOH have occurred at the locus; the SNV was ancestral to the ploidy change and the loss of heterozygosity

dropped the alternate allele. This weight can be converted into a joint attachment probability by normalizing over the pairs of edges for which $\{e_1, e_2\} \subset E - E_L$ and $e_1 \preceq e_2$.

The weights for cases 2 and 3 can be found using similar logic to be:

$$W^{(2)}(S_{e_1},\, L_{e_2}) = \frac{\pi_2(e_2)\pi_1(e_1)}{\pi_1(e_2)\pi_0(e_1)}\overline{p}_{e_1}\overline{p}_{e_2} \tag{24}$$

$$W^{(3)}(S_{e_1},\, L_{e_2}) = \frac{\pi_2(e_1)}{\pi_0(e_1)}\overline{p}_{e_1}\overline{p}_{e_2} \tag{25}$$

although $W^{(3)}$ has the slightly different domain for normalization: $e_2 \in E - L$, $e_1 \in E$ and of course $e_1 \succeq e_2$.

Marginal probabilities of ploidy change

For cases 1 and 2, it will be useful to answer the following question at each edge: assuming there is a case i event at this locus and a SNV has already been observed ancestral to this edge, what is the probability that a LOH event will happen at this edge? Let us assume that an SNV occurs at this locus and that a case 1 loss of heterozygosity also definitely occurs. This is to say that if an edge experiences a loss of heterozygosity the alternate allele must be dropped, and an ancestral SNV at this locus is implicit. We first see that for any $e' \preceq e$:

$$\begin{aligned} P^{(1)}(L_e \mid S_{\preceq e}) &= P(L_e \cap S_{\preceq e}) \\ &= P(L_e \cap (S_\rho \cup S_{e_i} \cup \dots) \\ &= P((L_e \cap S_\rho) \cup (L_e \cap S_{e_i}) \cup \dots) \end{aligned}$$

Due to infinite sites, mutations on different branches at a locus are mutually exclusive, hence we can sum

$$P^{(1)}(L_e \mid S_{\preceq e}) \propto \sum_{e' \prec e} W^{(1)}(L_e, S_{e'}) + \frac{1}{2}W^{(1)}(L_e, S_e)$$

If the SNV and LOH happen on the same branch, on average we expect the SNV to occur halfway down the branch leaving only half the distance for a ploidy change to occur afterwards, hence the factor of $\frac{1}{2}$ for the second term. The conditional probability is proportional to the term on the right up to a normalization factor.

This has all been shown assuming case 1 to be true for the locus, but the values for case 2 are exactly similar and can be computed at the same time. Case 3, however, is somewhat different, as we need to find

the probability of an LOH given that a SNV occurs in a descendant branch:

$$P^{(3)}(L_e \mid \neg S_{\prec e}) = P(L_e \cap S_{\succeq e}) \propto \sum_{e' \succ e} W^{(3)}(S_{e'}, L_e) + \frac{1}{2} W^{(3)}(S_e, L_e)$$

Again to return the true marginal probability these sums must be normalized sum over all valid pairs.

The marginal probabilities here have so far assumed not only that there is a SNV and loss of heterozygosity at the locus, but also that their particular case is true. We established when determining prior probabilities for σ that $P(\text{case } i \mid LOH, SNV) = 1/3$ for $i \in [0, 3]$. We take $P(LOH) = P(H)$ as an input parameter, and $P(SNV)$ for a locus can be given by the posterior distribution of σ for the locus, where $P(SNV) = 1 - P(\sigma = 0)$.

Upwards-downwards algorithm

The naïve approach for calculating these values would be to visit each edge in a tree traversal and from each edge retrieve all the values required from its ancestors or descendants. If we consider $P^{(3)}(S_{L_e \mid \neg S_{\prec e}})$, however, at each edge e we require the joint probability $P^{(3)}(S_{e'}, L_e)$ using values from all the descendants of e as well as a product of probabilities from all the ancestors of e. The naïve approach has a total time complexity of $O(nm^2)$ which does not asymptotically increase the complexity of SCarborSNV, but nonetheless there is a more efficient algorithm using a dynamic programming approach.

We saw that $\pi_g(e)$ were useful values that could be found recursively in a single pass up the tree. Similarly there are other sub-problem values we can store at each node so that we only need to make two passes of the tree to find our final phylogeny aware genotype probabilities.

In the first pass (upwards step) we store three values:

- π_g and π_m
  - The value at e is the sum of values from both children.
  - The base case at the leaves is the original genotype posteriors.
- $\sum_e W(S_e)$
  - The value at e is the sum from both children added to $W(S_e)$.
- $\sum_{e' \succ e} \frac{\pi_2(e')}{\pi_0(e')} \overline{p}_{e'}$
  - The value at e is the sum from both children added to the term evaluated at e.

Once the root has been reached, the value of $\sum W(S_e)$ can be used to normalize these W values to $P(S_e)$

at each edge in the following steps. In the second pass (downwards step) we compute the following partial sum by adding the value at e to the partial sum of the parent:

- $\sum_{e' \prec e} \frac{\pi_1(e')}{\pi_0(e')} \overline{p}_{e'}$

From these values we can calculate:

$$P^{*(1)}(S_{e'}, L_e) = \left(\sum_{e' \prec e} \frac{\pi_1(e')}{\pi_0(e')} \overline{p}_{e'} + \frac{1}{2} \frac{\pi_1(e)}{\pi_0(e)} \overline{p}_e \right) \overline{p}_e \frac{\pi_0(e)}{\pi_1(e)}$$

Each time one of these sums is calculated in the downward pass we can add it to a global total. The same is done for cases 2 and 3. These are the last remaining value required to calculate all edge probabilities: $P(S_e)$, $P^{(1)}(L_e \mid S_{\preceq e})$, $P^{(2)}(L_e \mid S_{\preceq e})$ and $P^{(3)}(L_e \mid \neg S_{\preceq e})$.

Algorithm : Upwards-Downwards

```
    procedure FillUpwards(Node)                                   ▷ Recursive upwards pass
        if Node is leaf then
            Node.π_g ← P(g_j = g, | D_i)
            Node.π_μ ← P(g_j = 1, | D_i) + P(g_j = 2, | D_i)
 5:         Node.∑W(S_e) ← Node.W(S_e) ← Node.π_μ ÷ Node.π_0 × Node.p̄
            Node.aux0 ← Node.π_2 ÷ Node.π_0 × Node.p̄
            Node.∑aux0 ← 0                  ▷ Auxiliary partial sum for calculating probabilities
        else
            FillUpwards(Child 1)                                  ▷ Recursive calls first
10:         FillUpwards(Child 2)
            Node.π ← Child 1.π× Child 2.π                         ▷ For all four π values
            Node.∑W(S_e) ← Node.W(S_e) + Child 1.∑W(S_e) + Child 2.∑W(S_e)
            Node.aux0 ← Node.π_2 ÷ Node.π_0 × Node.p̄
            Node.∑aux0 ← Child 1.∑aux0 + Child 1.aux0 + Child 2.∑aux0 + Child 2.aux0
15:     end if
    end procedure
    procedure FillDownwards(Node)                                 ▷ Recursive downwards pass
        if Node is root then
            Node.∑ π_1/π_0 p̄ ← π_1/π_0 p̄                          ▷ Initialize sum at root
20:         global ∑W(S_e) ← Node.∑W(S_e)                         ▷ Used to normalize weights
            Node.P(S_e) ← Node.W(S_e) ÷ global ∑W(S_e)
            FillDownwards(Child 1)
            FillDownwards(Child 2)
        else
25:         Node.∑ π_1/π_0 p̄ ← Parent.∑ π_1/π_0 p̄ + π_1/π_0 p̄
            Node.P(S_e) ← Node.W(S_e) ÷ global ∑W(S_e)
            if Node is not leaf then
                FillDownwards(Child 1)                            ▷ Recursive calls last
                FillDownwards(Child 2)
30:         end if
        end if
    end procedure
    FillUpwards(Root)                          ▷ Call both traversals to fill needed values
    FillDownwards(Root)
```

The upwards-downwards algorithm prepares the tree at each candidate locus for a final dynamic programming approach to call the final genotypes.

Genotyping cells

With all of these values computed for each of the edges in the tree, we can use a dynamic programming algorithm to genotype the cells. We complete a depth first traversal of the tree keeping track of genotype probabilities at every node. We also keep track of the probability $P(L^{(3)}_{\preceq e})$ that a silent case 3 haploid event has occured ancestral to each node. The initial conditions for the root node are the genotype probabilities $P(g = 0) = 1$ and $P(g = 1) = P(g = 2)$ and $P(L^{(3)}_{\preceq \rho}) = 0$. Let n_1 be the direct ancestor of n_2 such that n_1 and n_2 are joined by e and have genotypes g_1 and g_2 respectively. Therefore we define the relations:

$$P(L^{(3)}_{\preceq n_2}) = P(L^{(3)}_{\preceq n_1}) + P(L^{(3)}_e)$$

$$\begin{aligned} P(g_2 = 0) =& P(g_1 = 0)\left[(1 - P(S_e)) + P(S_e)P(L^{(1)}_e)\right] \\ &+ P(g_1 = 1)\left[P(L^{(1)}_e)\right] \end{aligned}$$

$$\begin{aligned} P(g_2 = 1) =& P(g_1 = 0)\left[P(S_e)(1 - P(L^{(1)}_e) - P(L^{(2)}_e))\right] \\ &+ P(g_1 = 1)\left[1 - P(L^{(1)}_e) - P(L^{(2)}_e)\right] \end{aligned}$$

$$\begin{aligned} P(g_2 = 2) =& P(g_1 = 0)\left[P(S_e)(P(L^{(3)}_{\preceq n_2}) + P(L^{(2)}_e))\right] \\ &+ P(g_1 = 1)\left[P(L^{(2)}_e)\right] \\ &+ P(g_1 = 2) \end{aligned}$$

When the tree traversal reaches the leaf nodes, these are taken to be the cell genotype probabilities. The cell will be genotyped in accordance with the highest probability.

2.8 Additional computational methods

Some likelihoods used in SCarborSNV may be the product of many small probabilities, especially if read depth is high. Therefore, despite the use of extended precision `long double` types, there is a possibility of floating point underflow occurring when manipulating and storing these values. Therefore all of SCar-

borSNV's calculations are done in log-space.

Sums of probabilities or likelihoods would require converting back from log-space and potential underflow. Therefore SCarborSNV has implemented LogSumExp functions to minimize potential underflow in these situations.

3 Results

Since there is no way of establishing a ground truth of which cells contain which mutations from real SCS data, most of the results are taken from simulated datasets. SCarborSNV uses phylogenetic inference to overcome some of the errors of low coverage SCS data, and therefore the simulated data must reflect a phylogenetic relationship, with mutations being attached to random points of the phylogeny.

We therefore have created a simulator to generate a phylogeny of cells with different mutations, and then introduce the error profile characteristic of single-cell DNA sequencing by sampling amplification errors, sequencing errors and allelic dropout events. To ensure that SCarborSNV also works on real data, we have also compared SCarborSNV to Monovar and SciΦ on a clinical cancer SCS dataset, however since there is no ground truth for these results they are somewhat more limited.

We analyze SCarborSNV both on its own and in comparison to similar tools on a number of metrics including accuracy of phylogeny, accuracy of site calls, accuracy of cell calls and overall time taken. We find that SCarborSNV is drastically faster than similar tools with high recall and reasonable precision in calling simulated mutations.

3.1 Simulating datasets

To simulate datasets to benchmark SCarborSNV, we have modelled various aspects of the random process of tumor evolution, amplification and sequencing. We first define an overall genome length, as well as chromosome length. In the first stage we have a single node that is allowed to acquire mutations over a set number of generations. At each generation the node has a fixed probability of acquiring a new point mutation anywhere within the genome range. At each generation there is also a small probability that a chromosome will experience a LOH, in which case that chromosome will be marked as homozygous with one chromosome randomly chosen dropped from the homologous pair. This first stage is intended to model germline and clonal mutations.

After a fixed number of generations, the node can still acquire SNVs and chromosomal aberrations, but also has some chance of splitting into two nodes. In this eventuality, the first node is removed from the list active nodes, and the two new nodes are added. Hereafter at each generation each active node has a random

chance of acquiring a SNV, acquiring a ploidy change or branching. The descendants of a node both gain a copy of the parent's mutational history.

When the number of active cells increases to the desired number of cells, the active nodes are then 'sequenced.' First each diploid chromosome has a random chance of experiencing allelic dropout, and a note is made of which chromosomes are lost. Then for each cell, at each locus, a non-negative read depth is sampled from a Gaussian distribution centered about an average read depth. Then for each 'read' an intermediate allele is 'amplified' with some probability of error from a random selection of the alleles present in that cell at that locus. Finally a read error is sampled from a Gaussian distribution about an average read error, and the intermediate allele is 'sequenced' with a chance of error equal to the read error.

The final sequenced read and associated quality score is then written to a pileup file. The true genotypes before sequencing (or allelic dropout simulations) are written to a VCF file for reference.

3.2 Simulated results

Since there is no a priori way of knowing which cells contain what mutations in real data, all accuracy benchmarking results must be taken from simulated data. SCarborSNV is compared against the two similar tools Monovar and SciΦ on time, precision, recall and F1 score on simulated datasets with short (4000bp) genomes.

Time comparison

The goal of SCarborSNV is to call SNVs efficiently from SCS data, and therefore the first important measure of success is the time taken by the program. We generated datasets from simulated phylogenies with three cells up to 100 cells. Next we compared the time taken for SCarborSNV to analyze these datasets, as well as its two competitors: Monovar and SciΦ. We generated 33 datasets with three cells, but decreased the number of tests as the number of cells increased for practical time reasons. Each tool was run on the exact same data; all the tools were run using default settings, on a single CPU core.

As we can see in Figure 3, SCarborSNV far outperformed its competition at all numbers of cells tested. For 100 cells sequenced from a simulated genome of 4000 sites, SCarborSNV completed in just over a minute and a half while Monovar took over 48 minutes. Even for small numbers of cells SciΦ ran much slower than the other two tools.

In these tests SCarborSNV was much faster than Monovar, which was much faster again then SciΦ. Since SCarborSNV has an asymptotically quadratic time complexity we expect that SCarborSNV will outperform its competitors on speed by even larger margins when yet larger datasets are analyzed.

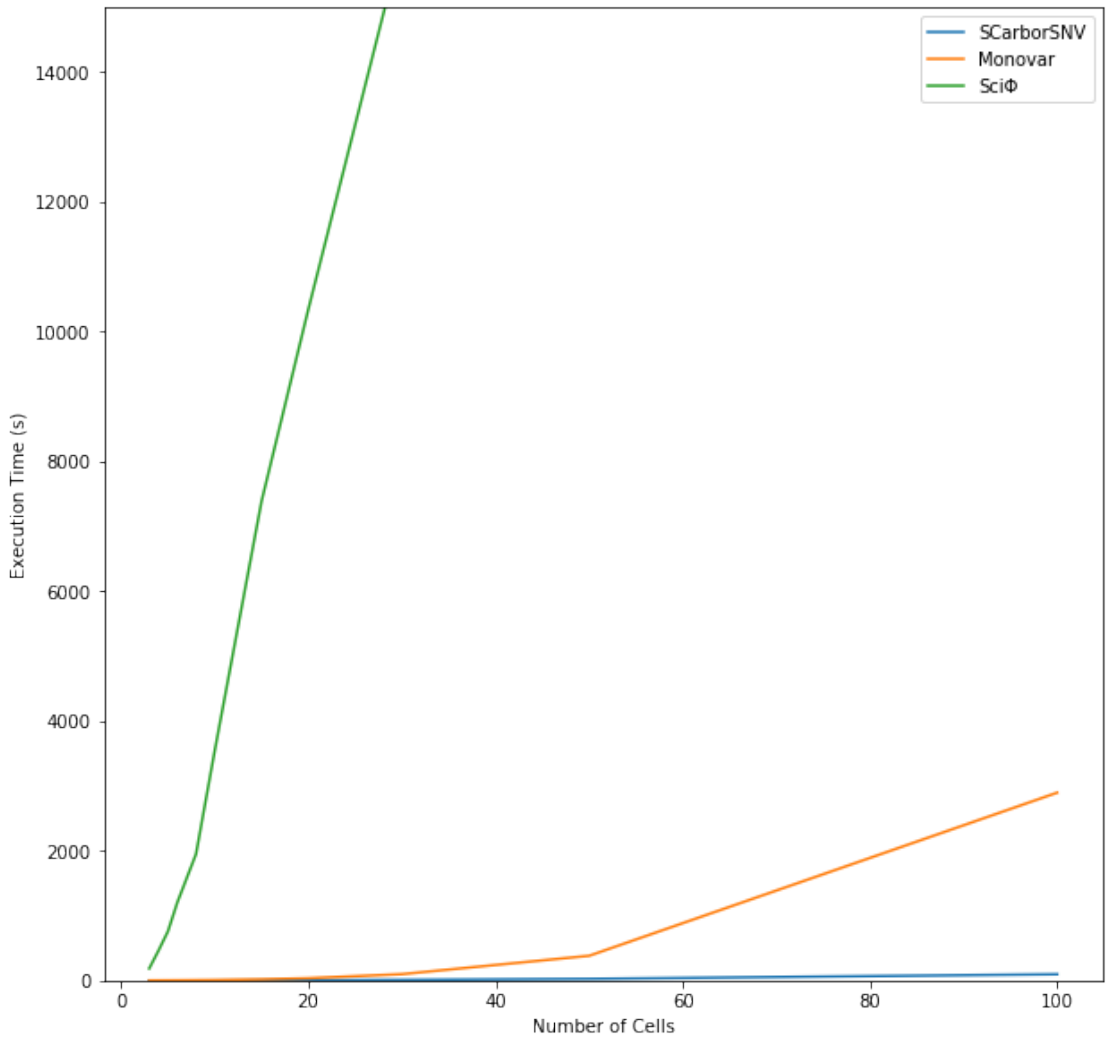


Figure 3: Average time taken by three SCS tools for 4000 simulated sites covering different numbers of related cells. The times are the averaged CPU times when run on an Intel® i7-4700MQ processor running at 2.40GHz.

Cell calling accuracy

As well as the time benchmarking with a variable number of cells, we benchmarked the accuracy of SCarborSNV and other tools on 100 simulated datasets with 10 cells with genomes of 4000 sites each. We compared the results of SCarborSNV, Monovar and SciΦ on their abilities to correctly call sites as variant, as well as calling individual cells as variant. Figure 4 shows that on these simulations SCarborSNV had a distinctly higher recall than Monovar, however this was at the cost of generally reduced precision. On these datasets, SciΦ had significantly lower recall than the other two tools, with an average precision between Monovar and SCarborSNV. In other words SCarborSNV on average called more cells as variant, including more true variants but also more false positives. Overall, when compared with its competitors SCarborSNV

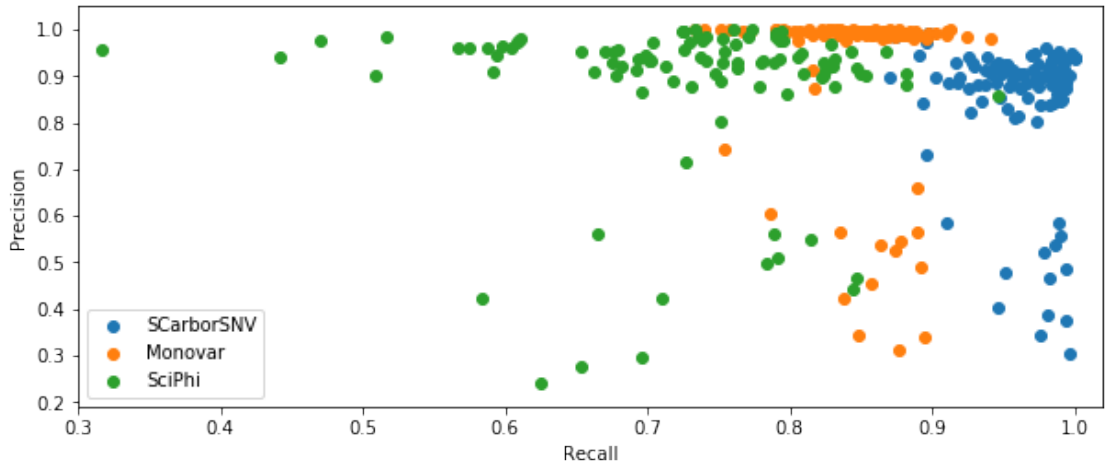


Figure 4: Variant cell calling from simulated datasets of 10 cells each.

did have a better F1 score for calling cells as variant from these data, as shown in Table 1.

Algorithm	Recall	Precision	F1 Score
SCarborSNV	0.962	0.841	0.887
Monovar	0.849	0.923	0.873
SciΦ	0.720	0.877	0.772

Table 1: Average recall, precision and F1 score of variant cell calling for three different algorithms benchmarked on 100 simulated datasets of 10 cells each.

Site calling accuracy

As well as their abilities to call individual cells as variant at different loci, we also compared the three algorithms on their ability to call one or more cells as variant at a variant locus. In some applications it may be important to know whether certain mutations occur at all in any subpopulation of cells, so this is also an important metric.

Figure 5 shows the precision and recall of calling sites as variant for the three algorithms SCarborSNV, Monovar and SciΦ. In this measurement, SCarborSNV on average had better precision, recall and F1 score than Monovar, as shown in Table 2. SciΦ performed poorly on these simulations, with similar precision to the other tools but on average significantly lower recall.

Tree accuracy

A common way to compare unrooted labeled phylogenies is using the Robinson-Foulds metric. This distance measure between two phylogenies is based on the fact that a phylogenetic tree implies certain partitions of the set of terminal clades. The RF metric is equal to the sum of both the number of partitions implied by

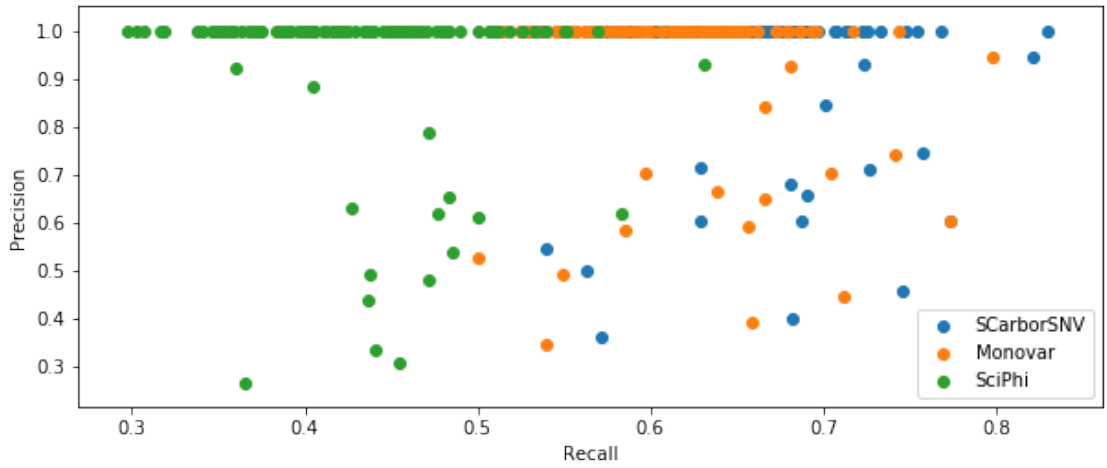


Figure 5: Variant site calling from simulated datasets of 10 cells each.

Algorithm	Recall	Precision	F1 Score
SCarborSNV	0.652	0.943	0.763
Monovar	0.617	0.941	0.737
SciΦ	0.431	0.935	0.579

Table 2: Average values for site calling for three different algorithms for the 100 simulated datasets of 10 cells each. Generally the accuracy for site calling is lower than that for cell calling since there are many sites with only one or two mutant cells each, and these sites are challenging to call accurately.

tree 1 that are not implied by tree 2, as well as the number of partitions implied by tree 2 that are not implied by tree 1 [32].

There is a similar value that can be computed for rooted trees that is based on clusters rather than partitions. This rooted Robinson-Foulds metric can be normalized empirically by comparing the metric to the average distance between randomly generated trees. In this way a normalized metric of 0 implies the exact same phylogeny, and a metric of 1 implies that the two phylogenies are only as similar as two random trees on average. We use the tool TreeCmp to compute this metric between the simulated tree and inferred tree for both the trees created my SCarborSNV and SciΦ. TreeCmp has empirically calculated the normalization values for trees with different numbers of leaves beforehand [33].

On the 100 10-cell datasets, SciΦ recovered trees more accurately than SCarborSNV, with an average normalized rooted Robinson-Foulds metric of 0.752 compared with 0.798 for SCarborSNV. The normalized Robinson-Foulds metric for these 100 datasets is summarized in Figure 6.

3.3 Results from real data

While simulated data are very useful since a ground truth can be used to determine accuracy, it is important to see also how well SCarborSNV works on real data. This however requires a significant amount of prepro-

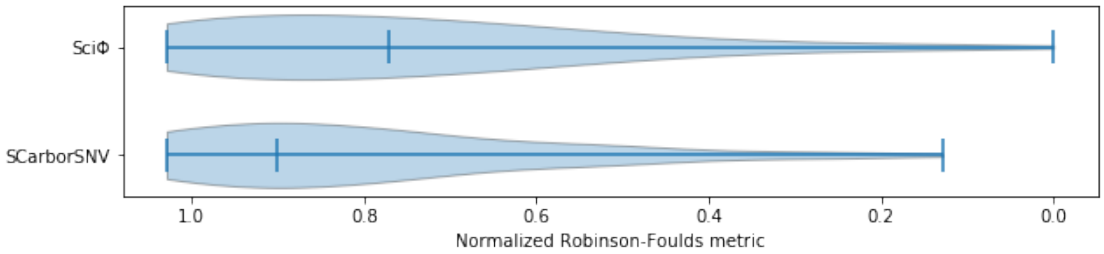


Figure 6: Violin plots of the normalized Robinson-Foulds metric between the ground truth phylogenies and those inferred by each tool. A lower value of the RF metric indicates more accurate recovery of the simulated phylogeny.

cessing to get the real data from a sequencing machine to be ready for SCarborSNV and its similar tools to use. Furthermore, real datasets have significantly longer genomes than the simulated examples above. Since SciΦ in particular takes so long to run, for practical reasons we only compare SCarborSNV on a portion of the genome by truncating the input pileup file. While this disadvantages both phylogeny aware algorithms by reducing the available kinship information, it should disadvantage them equally. It should be noted, however, that this restriction may make Monovar perform comparatively better than if this test were to be done on the full genome.

Data and preprocessing

We here use the same dataset used by [2] (accession: SRA053195), which includes DNA sequencing data from 16 single tumor cells from a triple negative breast cancer patient [39]. The raw FASTQ files were downloaded from the NCBI Sequence Read Archive using the SRAToolkit's fastq-dump program.

Next we indexed the hg19 human genome with the BWA index command, and then used BWA-mem to perform the Burrows-Wheeler alignment, aligning the reads from each of the cell samples to the now indexed hg19 [34]. We then added unique read-group identifiers to each aligned BAM file created by BWA using the Picard tool 'AddOrReplaceReadGroups'. We also used Picard to remove spurious duplicate reads, and sorted the BAM files with Samtools sort [36].

Using the Genome Anlaysis Toolkit (GATK) version 3, we then locally realigned the reads around indels using the RealignerTargetCreator and IndelRealigner tools [37]. These tools have been depracated from GATK version 4 and removed from the GATK data preparation best practices as GATK now recommends variant callers that perform haplotype assembly and do not require realignment [38]. Since SCarborSNV and other SCS callers do not perform this operation we have chosen to do the local realignment.

Finally we perform base quality score recalibration using GATK version 4. This process uses machine learning to adjust the read quality scores from the original sequencer output by detecting non-biological

artifacts ifrom the sequencing machine that may have systematically affected the read quality scores [37].

Cancer cell results

We analyzed 13 of the 16 tumor cells in SRA053195 using Monovar, SciΦ and SCarborSNV. For practical time considerations, we only analyzed the first 33.6 million (2^{25}) sites of each cell. Since there is no a priori way to know what mutations were actually present in this patient, we can only compare the results of the three tools.

In terms of mutations found, Figure 7 shows unique and common mutations identified by by the three different tools. While there were some mutations that Monovar called that SCarborSNV did not, there was only 1 (???) site that SciΦ called as variant that SCarborSNV did not. ScarborSNV did however call many more sites as variant than the other two tools, suggesting perhaps a high rate of false positive errors and therefore low precision. This discrepancy could hopefully be remedied by a different choice of threshold values.

Since SCarborSNV and SciΦ both infer phylogenies, the dendrograms of the two final inferred phylogenies are shown in Figure 8. Both show a similar genetic structure, with about half the cells simply branching off the main trunk, with the other half together in a more heirarchical structure.

4 Discussion

This thesis is an exploratory work to prove the concept of a phylogeny-aware SNV detection algorithm for SCS data that can have quadratic time complexity in the number of cells. As such, the massive speedup of using SCarborSNV over similar tools is very encouraging for the use of SCarborSNV as a medical and research tool and for the development of a more mature and robust tool derived therefrom.

State-of-the-art tools such as Monovar and SciΦ are slow enough to put a major restriction on the size of SCS datasets that can be analyzed. SCarborSNV therefore paves the way for a new era of massively multicellular single cell DNA sequencing analysis. In the context of tumor research and individualized therapies, this could allow the improved analysis of the intra-tumor heterogeneity of cancer from an individual patient, or allow more practical research into the nature of and mechanisms behind ITH itself.

There are many improvements to SCarborSNV as well that were beyond the scope of this initial project. These include parallelization, parameter re-estimation using EM and more robust neighbor-joining-like tree building. The upwards-downwards algorithm for inferring genotypes from a phylogeny also has potential for revision and improvement. Even as it currently exists, SCarborSNV could be very useful for an initial examination of SCS datasets with many cells: the high recall rate makes it useful for identifying potential

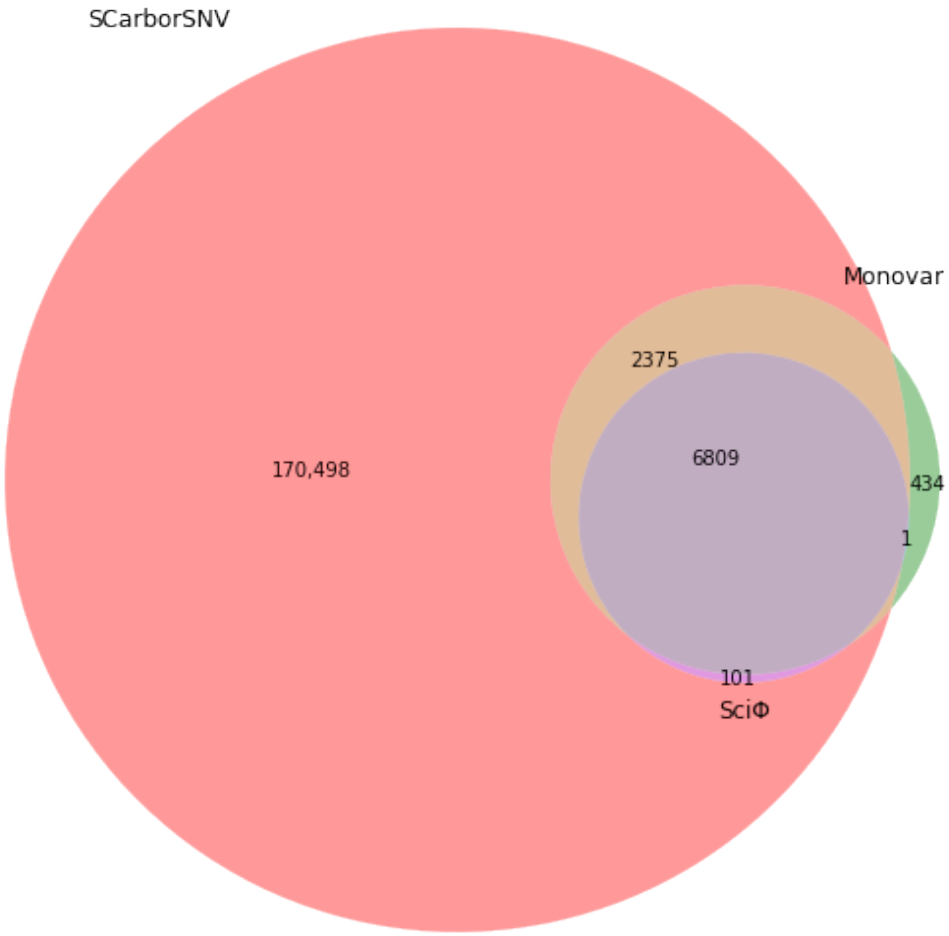
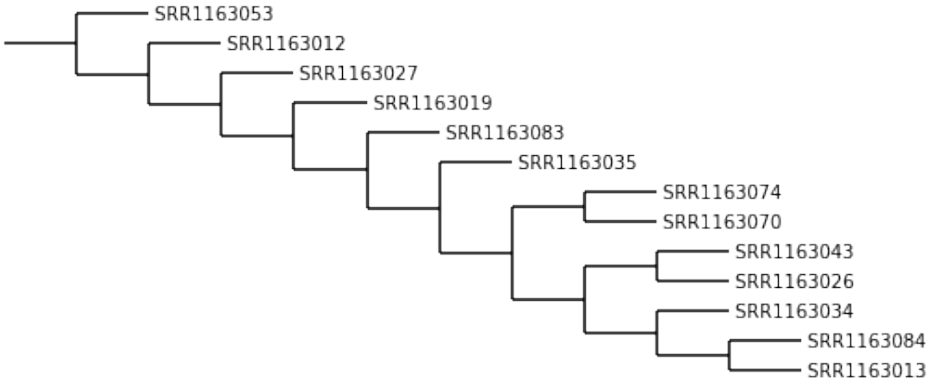


Figure 7: Venn diagram of variant sites called from the first 33.6 million sites of 13 cells in SRA053195. The radius of the SCarborSNV circle size has been reducedi by half to show detail in the overlapping regions.

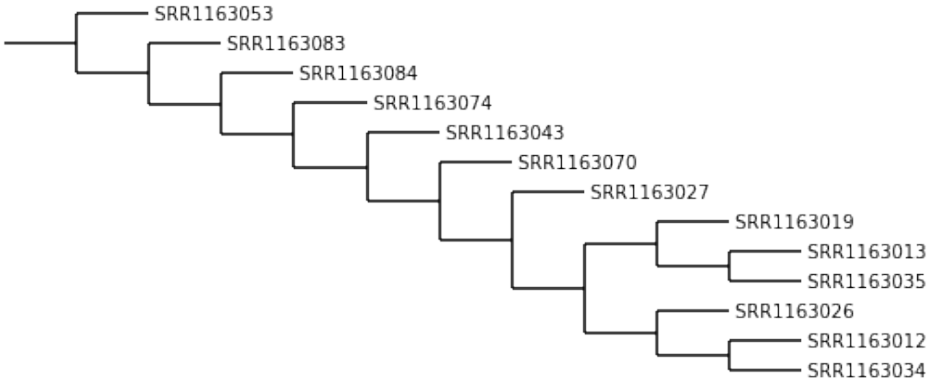
mutations that can be verified by other tools, especially because running SCarborSNV requires a minimal time investment.

4.1 Analysis of results

The simulated results for SCarborSNV show lots of promise for both high precision and recall of both mutant sites and individual cells. For the simulated data sets of 10 related cells each with a 4000bp genome, SCarborSNV performed similarly and even better than the state-of-the-art. These results somewhat contradict those in [2] however, showing both higher precision and recall for Monovar over SciΦ. Since the simulator design for our tests was much more rudimentary than that used in [2], further benchmarking will have to be



(a) Dendrogram of tree inferred by SCarborSNV



(b) Dendrogram of tree inferred by SciΦ

Figure 8: Comparison of the phylogenies inferred by SCarborSNV and SciΦ on cells from SRA053195. For better comparison all edge lengths have been set to 1.

done to compare the two tools.

Based on the 10-cell results, the tree reconstruction accuracy of SCarborSNV was similar to that of SciΦ. This suggests that even if the method of inferring genotypes from a reconstructed tree could be improved, the tree building itself is likely to be sufficiently accurate to infer unread genotypes despite the massively reduced time to infer the tree when compared to SciΦ.

The results from real data were somewhat more disappointing. Although SCarborSNV called almost all the sites called by the other two tools, it also called very many sites not called by the other two tools. It seems that while SCarborSNV has just as good if not better recall than the other two tools, the precision may be significantly lower on real data. There is no way to discern how many of these extra reads were real mutations and how many were false positives, and in simulation SCarborSNV did regularly find bona

fide mutations that the others did not. More work is needed to examine the effect of initial settings and parameters of SCarborSNV to reduce sensitivity while maintaining high recall.

On the whole, given the massively improved complexity, SCarborSNV has proved to be a very promising tool for fast discovery of mutations from SCS data. At the least, SCarborSNV has shown that much faster SCS inference is possible than by current tools.

4.2 Parallelization

While not yet implemented, SCarborSNV can be optimally parallelized for most of the stages of the algorithm. For the determination of cell genotype likelihoods all sites are considered completely independently, and therefore all n of these computations could easily be done on multiple processors. After the quick pre-computation of σ priors, the stages of SCarborSNV before tree building could be optimally parallelized on up to n CREW PRAM processors.

The neighbor joining stage of SCarborSNV may not be as straightforward to parallelize nor optimal, but since the tree reconstruction only needs to occur once it is unlikely that continuing to execute this part sequentially will impose a major slowdown for the overall program.

For the last stage of upwards-downwards genotype inference we again treat sites independently, and again we could optimally parallelize this section for up to n' processors, where n' is the number of candidate loci.

Furthermore, SCarborSNV is implemented in C with only standard libraries, and therefore parallel implementation using POSIX threads is possible without too much modification to the existing program structure.

4.3 Potential improvements: EM, GCNs and maximum likelihood

A potential improvement to the SCarborSNV algorithm would be to include phylogeny building and genotyping in an EM-like loop. At the stage of the upwards-downwards genotype inference algorithm, the probability of an LOH event, clonal mutation, allelic dropout and other events could be stored. These calls of various data artifacts or biological events could be used to re-estimate initial parameters for successively more accurate calls.

Different approaches for such parameter re-estimation would need to be explored. Since SCarborSNV is reliant on several initial global parameters such as somatic mutation frequency, these initial parameters could be re-estimated and the whole program could be run again. Another method may be to store the phylogenetically inferred posterior genotypes for each cell at each locus and then use these values to recompute pairwise distances to rebuild the approximate phylogeny. Some combination of these iterative methods could

also be used. If one or a fixed number of iterations of this step were applied, SCarborSNV would maintain quadratic time complexity in the number of cells and at worst experience linear slowdown. Since SCarborSNV is so much faster practically than similar tools it could very likely perform with competitive speed even with multiple iterations of the main program.

Another potential avenue for improvement to SCarborSNV would be the use of a graph convolutional neural network to infer genotypes from the approximate phylogeny. Because this stage of the algorithm receives a tree structure and normalized posterior probabilities as input, it is possible that a GCN could be pre trained via simulation once as part of the development of the tool and only optionally retrained by users.

One final area of future exploration is phylogeny aware genotype calling via maximum likelihood attachment. Using the π values computed in the upwards-downwards algorithm, a likelihood can be assigned to each attachment position of a SNV in the inferred tree. By including a pseudo-branch not in the tree, a mutation attachment to which represents a welltype locus, the cell genotypes could be called based on this maximum likelihood position. Even if not ultimately used to call genotypes, such maximum likelihood estimates could be used in an iterative EM version of the algorithm.

4.4 Robustness

Currently SCarborSNV is quite robust and can handle unknown nucleotides, indels and some missing data. If there is some pair of cells in a dataset that share no overlapping sequenced loci with each other, then a pairwise distance between these cells cannot be computed. Since all pairwise distances are needed for the neighbor joining algorithm this causes SCarborSNV to fail. For normal sequencing data of whole genomes or whole exomes this should be a rare occurrence, however there are potential ways to handle this case. For example, there are iterative methods to approximately impute missing pairwise distances from a partial distance matrix [35].

4.5 Conclusion

Single cell DNA sequencing is in its infancy, but it shows great promise to accurately resolve somatic genetic heterogeneity, including intra-tumor heterogeneity. Since the genetic diversity of competing subclones in a tumor allow it to evolve, ITH is a direct cause of treatment resistance in cancer [7, 10, 11]. As SCS data become cheaper and more available, the ability to analyze more of these data is increasingly important for research and individualized therapy. While tools such as SciΦ and Monovar have proved invaluable for SCS analysis, their complexity ultimately will limit the size of SCS datasets that can be analyzed.

Despite some room for improvement and more rigorous testing with more realistic and variable simulated

data, SCarborSNV has pioneered a new direction in efficient phylogeny-aware SCS data analysis that may open up massively multicellular SCS studies, with thousands of cells or more, rather than 10s.

5 Software availability

SCarborSNV has been implemented in C and is available at `https://github.com/coldham10/SCarborSNV`. The SCarborSNV source has been licensed under the GNU General Public License, Version 3.

References

[1] Hamim Zafar, Yong Wang, Luay Nakhleh, Nicholas Navin, and Ken Chen. Monovar: single-nucleotide variant detection in single cells. *Nature methods*, 13(6):505, 2016.

[2] Jochen Singer, Jack Kuipers, Katharina Jahn, and Niko Beerenwinkel. Single-cell mutation identification via phylogenetic inference. *Nature communications*, 9(1):5144, 2018.

[3] Richard Durbin, Sean R Eddy, Anders Krogh, and Graeme Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.

[4] Naruya Saitou and Masatoshi Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4(4):406–425, 1987.

[5] Siddhartha Mukherjee. *The emperor of all maladies: a biography of cancer*. Simon and Schuster, 2010.

[6] Andrea Sottoriva, Haeyoun Kang, Zhicheng Ma, Trevor A Graham, Matthew P Salomon, Junsong Zhao, Paul Marjoram, Kimberly Siegmund, Michael F Press, Darryl Shibata, et al. A big bang model of human colorectal tumor growth. *Nature genetics*, 47(3):209, 2015.

[7] Gunes Gundem, Peter Van Loo, Barbara Kremeyer, Ludmil B Alexandrov, Jose MC Tubio, Elli Papaemmanuil, Daniel S Brewer, Heini ML Kallio, Gunilla Högnäs, Matti Annala, et al. The evolutionary history of lethal metastatic prostate cancer. *Nature*, 520(7547):353, 2015.

[8] Serena Nik-Zainal, Peter Van Loo, David C Wedge, Ludmil B Alexandrov, Christopher D Greenman, King Wai Lau, Keiran Raine, David Jones, John Marshall, Manasa Ramakrishna, et al. The life history of 21 breast cancers. *Cell*, 149(5):994–1007, 2012.

[9] Serena Nik-Zainal, Ludmil B Alexandrov, David C Wedge, Peter Van Loo, Christopher D Greenman, Keiran Raine, David Jones, Jonathan Hinton, John Marshall, Lucy A Stebbings, et al. Mutational processes molding the genomes of 21 breast cancers. *Cell*, 149(5):979–993, 2012.

[10] Mel Greaves and Carlo C Maley. Clonal evolution in cancer. *Nature*, 481(7381):306, 2012.

[11] Lauren MF Merlo, John W Pepper, Brian J Reid, and Carlo C Maley. Cancer as an evolutionary and ecological process. *Nature reviews cancer*, 6(12):924, 2006.

[12] Harith Rajagopalan and Christoph Lengauer. Aneuploidy and cancer. *Nature*, 432(7015):338, 2004.

[13] James M Heather and Benjamin Chain. The sequence of sequencers: The history of sequencing dna. *Genomics*, 107(1):1–8, 2016.

[14] Tracy Tucker, Marco Marra, and Jan M Friedman. Massively parallel sequencing: the next big thing in genetic medicine. *The American Journal of Human Genetics*, 85(2):142–154, 2009.

[15] Yong Wang and Nicholas E Navin. Advances and applications of single-cell sequencing technologies. *Molecular cell*, 58(4):598–609, 2015.

[16] Tal Nawy. Single-cell sequencing. *Nature methods*, 11(1):18, 2013.

[17] HX Dang, BS White, SM Foltz, CA Miller, Jingqin Luo, RC Fields, and CA Maher. Clonevol: clonal ordering and visualization in cancer sequencing. *Annals of oncology*, 28(12):3076–3082, 2017.

[18] Nicholas E Navin. Cancer genomics: one cell at a time. *Genome biology*, 15(8):452, 2014.

[19] Zhong Wang, Mark Gerstein, and Michael Snyder. Rna-seq: a revolutionary tool for transcriptomics. *Nature reviews genetics*, 10(1):57, 2009.

[20] Tomislav Ilicic, Jong Kyoung Kim, Aleksandra A Kolodziejczyk, Frederik Otzen Bagger, Davis James McCarthy, John C Marioni, and Sarah A Teichmann. Classification of low quality cells from single-cell rna-seq data. *Genome biology*, 17(1):29, 2016.

[21] J.M. Berg, J.L. Tymoczko, and L. Stryer. *Biochemistry*. Biochemistry (Berg). W. H. Freeman, 2007.

[22] Ruli Gao, Alexander Davis, Thomas O McDonald, Emi Sei, Xiuqing Shi, Yong Wang, Pei-Ching Tsai, Anna Casasent, Jill Waters, Hong Zhang, et al. Punctuated copy number evolution and clonal stasis in triple-negative breast cancer. *Nature genetics*, 48(10):1119, 2016.

[23] Christopher Greenman, Philip Stephens, Raffaella Smith, Gillian L Dalgliesh, Christopher Hunter, Graham Bignell, Helen Davies, Jon Teague, Adam Butler, Claire Stevens, et al. Patterns of somatic mutation in human cancer genomes. *Nature*, 446(7132):153, 2007.

[24] Ludmil B Alexandrov, Serena Nik-Zainal, David C Wedge, Samuel AJR Aparicio, Sam Behjati, Andrew V Biankin, Graham R Bignell, Niccolo Bolli, Ake Borg, Anne-Lise Børresen-Dale, et al. Signatures of mutational processes in human cancer. *Nature*, 500(7463):415, 2013.

[25] Thomas H Jukes, Charles R Cantor, et al. Evolution of protein molecules. *Mammalian protein metabolism*, 3(21):132, 1969.

[26] Marc J Williams, Benjamin Werner, Chris P Barnes, Trevor A Graham, and Andrea Sottoriva. Identification of neutral tumor evolution across cancer types. *Nature genetics*, 48(3):238, 2016.

[27] Ugo Del Monte. Does the cell number 10^9 still really fit one gram of tumor tissue? *Cell Cycle*, 8(3):505–506, 2009.

[28] Shinichi Yachida, Siân Jones, Ivana Bozic, Tibor Antal, Rebecca Leary, Baojin Fu, Mihoko Kamiyama, Ralph H Hruban, James R Eshleman, Martin A Nowak, et al. Distant metastasis occurs late during the genetic evolution of pancreatic cancer. *Nature*, 467(7319):1114, 2010.

[29] Heng Li. A statistical framework for snp calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics*, 27(21):2987–2993, 2011.

[30] Si Quang Le and Richard Durbin. SNP detection and genotyping from low-coverage sequencing data on multiple diploid samples. *Genome research*, 21(6):952–960, 2011.

[31] Mary K Kuhner and Joseph Felsenstein. A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Molecular biology and evolution*, 11(3):459–468, 1994.

[32] David F Robinson and Leslie R Foulds. Comparison of phylogenetic trees. *Mathematical biosciences*, 53(1-2):131–147, 1981.

[33] Damian Bogdanowicz, Krzysztof Giaro, and Borys Wróbel. Treecmp: comparison of trees in polynomial time. *Evolutionary Bioinformatics*, 8:EBO–S9657, 2012.

[34] Heng Li. Aligning sequence reads, clone sequences and assembly contigs with bwa-mem. *arXiv preprint arXiv:1303.3997*, 2013.

[35] Vladimir Makarenkov and François-Joseph Lapointe. A weighted least-squares approach for inferring phylogenies from incomplete distance matrices. *Bioinformatics*, 20(13):2113–2121, 2004.

[36] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, and Richard Durbin. The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079, 2009.

[37] Mark A DePristo, Eric Banks, Ryan Poplin, Kiran V Garimella, Jared R Maguire, Christopher Hartl, Anthony A Philippakis, Guillermo Del Angel, Manuel A Rivas, Matt Hanna, et al. A framework for variation discovery and genotyping using next-generation dna sequencing data. *Nature genetics*, 43(5):491, 2011.

[38] Geraldine A Van der Auwera, Mauricio O Carneiro, Christopher Hartl, Ryan Poplin, Guillermo Del Angel, Ami Levy-Moonshine, Tadeusz Jordan, Khalid Shakir, David Roazen, Joel Thibault, et al. From fastq data to high-confidence variant calls: the genome analysis toolkit best practices pipeline. *Current protocols in bioinformatics*, 43(1):11–10, 2013.

[39] Yong Wang, Jill Waters, Marco L Leung, Anna Unruh, Whijae Roh, Xiuqing Shi, Ken Chen, Paul Scheet, Selina Vattathil, Han Liang, et al. Clonal evolution in breast cancer revealed by single nucleus genome sequencing. *Nature*, 512(7513):155, 2014.

[40] Christopher A Miller, Brian S White, Nathan D Dees, Malachi Griffith, John S Welch, Obi L Griffith, Ravi Vij, Michael H Tomasson, Timothy A Graubert, Matthew J Walter, et al. SciClone: inferring clonal

architecture and tracking the spatial and temporal patterns of tumor evolution. *PLoS computational biology*, 10(8):e1003665, 2014.