METHOD

Benchmarking Tools for Copy Number Aberration Detection from Single-cell DNA Sequencing Data

Xian Fan[1†], Mohammadamin Edrisi[1†], Nicholas Navin[2,3] and Luay Nakhleh[1*]

*Correspondence:
nakhleh@rice.edu
[1]Department of Computer Science, Rice University, Houston, Texas, USA
Full list of author information is available at the end of the article
[†]Contributed equally

Abstract

Background: Accurate detection of copy number aberrations (CNA) can aid in understanding the genetic causes of diseases. Three methods (CopyNumber, Ginkgo, and HMMcopy) have been applied to single-cell DNA sequencing data for CNA detection.

Results: In this paper, we benchmarked these three methods on simulated as well as biological datasets. We found that HMMcopy has the best accuracy of the three methods in terms of breakpoint detection but that Ginkgo is better in terms of detecting the actual copy numbers. In terms of computational requirements, HMMcopy consumes the least memory, but is in between the two other methods in terms of running time.

Conclusion: While single-cell DNA sequencing is very promising for elucidating and understanding CNAs, even the best existing method does not exceed 80% accuracy. New methods that significantly improve upon the accuracy of these three methods are needed. Furthermore, with the large datasets being generated, the methods must be computationally efficient.

Keywords: Single-cell sequencing; Copy number aberration; Benchmarking; Tumor evolution

Background

Acquired mutations are the main causes of cancer [1–3]. Copy number aberrations (CNAs) are one such type of acquired mutations and have been implicated in over-regulating oncogenes or down-regulating tumor suppressor genes [4]. Consequently, accurate detection of CNAs could hold a great promise to understanding some of the genetic underpinnings of cancer as well as developing targeted drugs. In the past two decades, a wide array of DNA technologies have been used to detect CNAs, among which the three most widely used are array Comparative Genomic Hybridization (aCGH), Next Generation Sequencing (NGS), and single-cell sequencing [5].

Array CGH [6] is a cytogenetic approach that uses fluorescent dyes on the test (tumor) and reference (normal) samples, and applies them to a microarray, which is an array of probes. Each probe is a DNA sequence that represents a region of interest. The size of a probe depends on the DNA sequence being used, and it varies from dozens of base pairs, such as oligonucleotides, to thousands of base pairs, such as bacterial artificial chromosomes. The probes from the paired samples, after being mixed together, hybridize at each target region. The fluorescence intensities can then be measured for both samples, and the ratio of the two is used to inform about CNAs of the test sample relative to the normal one. Array CGH data is advantageous in comprehensively detecting aneuploidies, amplifications and deletions

simultaneously. A few computational methods [7–9] have been developed to detect CNAs using aCGH data. DNAcopy [7] applies a modification of binary segmentation [10] called circular binary segmentation (CBS) to aCGH to overcome data noise, but it suffers from the problem of outliers [8, 9]. HMMcopy [8] was designed to ameliorate the problem of outliers and uses a Hidden Markov Model (HMM) to divide the genome into piecewise fixed segments in order to makes inferences on CNAs. However, since aCGH data is limited in resolution and throughput [11], as well as suffers from a hybridization bias problem, it is not the optimal technology to detect CNAs for cancer samples.

Unlike aCGH, which obtains signal on only a limited number of genomic sites of interest, NGS technology makes it possible to survey the whole genome at a nucleotide-level resolution by sequencing millions of small fragments (reads) of the genome in parallel. By aligning the reads to an assembled reference genome, the reads that cover a position in the genome are counted to obtain the read depth at that position. Read depths at different regions of the genome can then be contrasted to assess hypothesized genomic locations where copy number gains and losses had occurred. NGS technologies suffer from high false positive rate compared with aCGH, due mainly to GC bias and the presence of repetitive regions [12, 13]. Even more challenging in the case of cancer genomes that are often aneuploid, contamination of normal cells may occur in the bulk tissue further complicating the task of estimating the absolute copy number from NGS data. To overcome these challenges, a plethora of computational tools [12, 14–20] have been developed for detecting CNAs from NGS data. SeqCNA [12] filters out potentially false-positive CNAs and corrects GC content for a more accurate CNA detection. CNAseg [14] analyzes flowcell-to-flowcell variability to avoid false-positive CNAs. CNAnorm [15] addresses the normal contamination and aneuploidy of the tumor sample to accurately infer CNAs. Paired-end NGS data provides another modality in addition to the read depth to infer CNA, and a few bioinformatics tools use this, including, for example, CNVer [17], CNVnator [18], ReadDepth [19], and Mseq-CNV [20].

Although both aCGH and NGS data can be used to detect CNAs, they do not provide high-throughput data at the single-cell resolution that is ideal for understanding tumor growth. In particular, intratumor heterogeneity [21] is best understood by using data obtained from individual cells within the tumor tissue. Indeed, in the last 10 years, the field has made great strides towards developing technologies for single-cell DNA sequencing. Data generated by these technologies can be analyzed to detect CNAs and other types of mutations in individual cells and individual clones within the heterogeneous tumor [22]. For example, DOP-PCR is a PCR amplification method that generates low-coverage data suitable for CNA detection in single-cell data [23–26]. However, it also suffers from uneven coverage and allelic dropout [22] that could lead to false-positive calls of CNAs. Beyond this method, three tools have been extensively applied to single-cell sequencing data for CNA detection: AneuFinder [27, 28], CopyNumber [29], and Ginkgo [30]. Like HMMcopy, AneuFinder uses a Hidden Markov Model (HMM) to determine the segmentation of the genome and the absolute copy number of each segment. CopyNumber [29] pools all cells together for joint segmentation to improve boundary detection accuracy. Since cancer cells in the same subclone mostly share the same CNA boundaries,

such strategy can improve the nucleotide resolution of the boundary by implicitly amplifying the signal in the data. Ginkgo [30] uses Circular Binary Segmentation (CBS) [7] to segment the genome, followed by inferring the integer value of the absolute copy number. It is worth noting that some methods designed for aCGH and NGS data have also been extensively used on single-cell data [31–34]. One such method is HMMcopy [35]. As both AneuFinder and HMMcopy are HMM-based methods, we focus on HMMcopy as a representative of the HMM-based approach. A more recent method is SCNV [36], which uses a bin-free segmentation method to do segmentation and copy number profiling. However, the method has not been widely applied to single-cell DNA studies.

Among CopyNumber, Ginkgo and HMMcopy, only CopyNumber utilized the pooled information from single-cell data. The other two methods can be equally well applied to bulk samples. Moreover, HMMcopy was designed for aCGH data originally, and thus does not take into account the specific error profiles that characterize single-cell sequencing data, such as low and uneven coverage, or the computational challenges that arise due to biological processes such as aneuploidy in a tumor single cell. While these three methods have been widely applied to analyze single-cell data [27, 31–34, 37–48], a comprehensive study of their performance is currently lacking. While Knouse *et al.* [32] assessed the performance of CBS and HMM-based methods on single-cell DNA sequencing data, their evaluation is limited to CNVs in brain and skin cells. Moreover, they did not investigate the effect of the ploidy on the accuracy of the methods.

The contribution of this paper is two-fold. First, we developed a single-cell CNA simulator that mimics realistic cell genealogy scenarios and is flexible enough to allow for controlling processes that affect CNA detection, such as ploidy and read count variability. Second, we used the simulator to generate synthetic datasets and benchmarked the three methods that have been used for CNA detection from single-cell DNA data: CopyNumber, Ginkgo, and HMMcopy. Moreover, we evaluated the methods on real data. In addition to assessing the accuracy of the methods in terms of precision and recall, we introduced the use of parsimony-based counting of copy number changes as a potential indicator of accuracy, especially when the ground truth is unknown. We found that in terms of accuracy of the detected breakpoints and memory consumption, HMMcopy is the best of the three methods, and in terms of running time, it is slower than CopyNumber but faster than Ginkgo. However, when evaluated the methods in terms of the actual copy number profiles they detect, we found that Ginkgo is more accurate than HMMcopy; in fact, we found that HMMcopy is not stable at this task (paradoxically, CopyNumber does not detect actual copy numbers). In terms of accuracy, CopyNumber has a very poor performance. Our results highlight the need for developing new accurate and efficient methods for CNA detection from single-cell DNA data.

Results and discussion

Simulations

We designed three experiments to evaluate the performance of the three methods under different conditions. The first experiment was designed to evaluate the recall and precision of the CNA detection methods. We designed the simulation study in

this experiment to produce single cells that have ideal read count variability and normal ploidy levels ranging between 2 and 3, so that we can learn how the methods perform when the data is relatively not challenging. The second experiment was designed to evaluate the performance of each method under a variety of ploidy levels. Specifically, we simulated single cells whose ploidies range from 1.5 to 5.26 (the ploidy of a cell is defined as the average copy number across the cell's genome). We then compared the recall and precision of the three methods on the simulated data at different ploidies. The third experiment was designed to assess the performance of each method under different coverage variabilities. In particular, we simulated single cells whose coverage variabilities mimic those produced by three single-cell sequencing technologies (MALBAC, DOP-PCR and TnBC) [49] that have been used for CNA detection.

Precision and recall of the methods

In the first experiment, we simulated the evolution of 10,000 cells from which we generated, through sampling without replacement, three 1000-cell datasets. For each cell, we simulated read data using a simulator that we developed (see the "Methods" section). We then aligned the reads back to the reference genome using BWA with default parameters [50, 51]. Finally, we ran each of three methods on the resulting BAM files, and computed the recall and precision of each method based on the ground truth generated by the simulator.

We assessed the methods' performances in coarse- and fine-grained analyses. For the coarse-grained evaluation, we inspected the breakpoint positions as well as whether they were consistent with the ground truth in terms of estimated gain/loss state (rather than the actual value) in the copy number. The predicted breakpoint is counted as consistent with the ground truth whenever it has the same status (i.e., the copy number increases or decreases) and its genomic location is within a certain distance of its counterpart in the ground truth. We varied the value of this distance to study the methods' accuracies. Each ground truth breakpoint was matched by at most one predicted breakpoint to avoid double counting of the true positive calls. For each method, we varied a parameter to obtain the receiver operator characteristic (ROC) curve, the details of which are described in the caption of Fig. 1.

A preliminary analysis of CopyNumber on the data revealed that the method achieves extremely low recall and precision. Since CopyNumber pools all cells together for breakpoint detection, we suspected that this poor performance owes mainly to the method's lack of sensitivity in detecting breakpoints shared by only a small number of cells. Therefore, to allow for more meaningful comparison of CopyNumber to the other two methods, we eliminated breakpoint pairs shared by fewer than five cells in the ground truth and used the resulting new ground truth to evaluate CopyNumber's recall and precision (but we did not filter the breakpoints for the other two methods). As Fig. 1a shows, CopyNumber still has, by far, the poorest performance. We hypothesize that for a breakpoint to be detectable by CopyNumber, it needs to be shared by a large number of cells. We further checked this by calculating the number of cells sharing a breakpoint that is called or missed by CopyNumber, and found that there is a significant difference between the two sets

(p-value $<$ 9.019e-09 for Student's t-test with mean 9.27 versus 5.35). We also observe that as the tolerance threshold for the detected breakpoint position increases, improvement in CopyNumber's performance is much larger than the improvement in the performance of the other two methods. However, even with the most forgiving threshold (two breakpoints are considered to be the same if their positions are within 1 million basepairs of each other), CopyNumber still has poorer performance than the other two methods even under the most stringent threshold. Overall, the results in Fig. 1a show that (1) HMMcopy generally outperforms the other two methods, with Ginkgo being a close second, and (2) that even HMMcopy's best recall and precision are around 0.8 and 0.7, respectively.

In the fine-grained analysis, we focused on the agreement of the absolute copy numbers on both 5' and 3' of an inferred breakpoint with those of the ground truth, in addition to the requirements on gain/loss consistency and distance. Since CopyNumber does not predict the absolute copy numbers for an individual cell, it is not considered in this analysis. Surprisingly, HMMcopy's prediction of the absolute copy number is not stable, leading to a bimodal distribution of both recall and precision (Additional file 1: Figure S1). We found that cells with low recall and precision correspond mainly to cases where HMMcopy made inaccurate estimates of the cells' ploidies (Additional file 1: Figure S2). We then selected only those cells for which the ploidy was correctly predicted (i.e., 2 or 3), and plotted the ROC curve of HMMcopy on them. We found that HMMcopy performed generally better than Ginkgo (Fig. 1b), which is in agreement with the coarse-grained analysis. The recall and precision for the two methods dropped, which is expected since the true positives and negatives are now measured most stringently. However, we observed that the difference in results between the coarse- and fine-grained analyses is not large, suggesting that once the breakpoint is found by these methods, predicting the absolute copy number can be attained quite accurately. This is especially true for Ginkgo whose ploidy prediction is stable.

Similar results were observed on the other two datasets (Additional file 1: Figures S3 and S4).

The results in Fig. 1 were obtained under default parameters except for the parameters that were tuned to generate the ROC curves (*alpha*, *gamma*, and *nu*). However, we found that the value of parameter *strength* in HMMcopy has to be much larger than the default value in order to make the results more expected, i.e., increasing recall is accompanied with decreasing precision, and vice versa. We therefore set *strength* to be 10 million. According to HMMcopy's users' guide, *strength* is the parameter that controls how much the initial values of e, which controls the probability of extending a segment, remains the same throughout the iterations. We found that setting *strength* would help to have a good quality control of the result by making the initial setting of e last throughout all the iterations. Apart from parameters *nu* and *strength*, we found that e is also an important parameter in HMMcopy. The larger the value of e, the smaller the chance that the breakpoint is detected. To explore which combination can yield the best performance for HMMcopy, we varied both e and *nu* and calculated the F1 score. The performance of HMMcopy is the best when *nu* is 4 and $e \geq 0.999999$ (Additional file 1: Figure S5).

We also analyzed the computational requirements in terms of running time and memory consumption for the three methods on a 1000-cell dataset (Fig. 2). Ginkgo

is the slowest among the three and CopyNumber is the fastest. HMMcopy is in between Ginkgo and CopyNumber in terms of running time. For memory consumption, Ginkgo is the most demanding of the three, followed by CopyNumber. HMMcopy is the lightest in terms of memory consumption. Note that in running Ginkgo, we eliminated the steps of generating figures such as heatmaps and copy number profile, so that these do not affect the running time and memory in comparison with the other two methods. For CopyNumber, an extra step is required to generate its input file. We used the intermediate result of HMMcopy, i.e., the GC corrected read count on each bin, as the input to CopyNumber. We take the time for calculating this intermediate file into account for CopyNumber for a fair comparison. Since CopyNumber processes all cells together, we suspect that more cells will require more memory, whereas Ginkgo and HMMcopy's memory requirements are not affected by the number of cells involved.

In summary, we found that HMMcopy is the most accurate in predicting breakpoints among the three. When HMMcopy's prediction of ploidy is accurate, its recall and precision of predicting the absolute copy number are the best among the three methods. However, it is not as stable as Ginkgo in terms of absolute copy number detection since its prediction of ploidy is wrong for many cells (49.4% for default values of *nu* and *e*). CopyNumber's recall and precision are the worst of the three methods. Moreover, it cannot predict the absolute copy number for each individual cell, and thus is not as applicable as the other two methods.

The effect of ploidy on performance

To test the robustness of the methods to different ploidies, we varied the ploidy by tuning the parameters that control whole chromosomal amplifications and the rate of deletion (see the "Methods" section). We varied the ploidy from 1.5 to 5. Specifically, we used 1.5, 2.1, 3, 3.8, and 5.26 (we refer to them as 1, 2, 3, 4, and 5, respectively, hereafter), and generated three datasets for each ploidy. We tuned the coverage parameter for each ploidy so that the total number of reads for different ploidies are approximately the same to avoid adding reads for larger genomes resulting from higher ploidies.

We ran each method using their default parameters (except the *strength* parameter in HMMcopy). Finding CopyNumber's recall to be zero using the default *gamma*, we tuned *gamma* using the optimal value, i.e., 5, shown in Fig. 1). We then found the recall greatly increased with this setting. Similar to the previous experiment, we again removed those breakpoint pairs shared by ≤ 5 cells from the ground truth for evaluating CopyNumber's performance.

We used different combinations of the parameters to simulate high-ploidy cells (details are in the "Methods" section), i.e., 4 and 5, and found that in the absence of odd and intermediate copy numbers, HMMcopy's inference of the ploidy and absolute copy numbers were inaccurate (Additional file 1: Figure S6). This is also the case for Ginkgo in the absence of the odd copy numbers. However, despite the lack of intermediate copy numbers, Ginkgo correctly predicted absolute copy numbers for the case of ploidy=5, showing that Ginkgo is more robust to changes in ploidy than HMMcopy in terms of predicting absolute copy numbers. In summary, the lack of odd or intermediate copy numbers in the data led to wrong predictions of

absolute copy numbers. We then tuned the simulator's parameters so that in higher ploidies there are odd and intermediate copy numbers to avoid the extremely hard cases for CNA detection (details are in the "Methods" and "Conclusions" sections). Fig. 3 shows the precision and recall for the three methods for coarse- and fine-grained analyses, respectively.

In the coarse-grained analysis, Ginkgo's recall is > 0.8 in general, but its precision is relatively low (i.e., < 0.4) for ploidy 2 and 3. This is probably because it was affected by the variability of the read counts and over-segmented the genome. With similar recall, HMMcopy has higher precision at all ploidies. CopyNumber's recall and precision are low (< 0.4) for all ploidies, with low recall and precision for ploidy 5 and low precision for ploidy 2.

In the fine-grained analysis, Ginkgo's recall and precision dropped by about 10% as compared with the coarse-grained results. Its recall dropped the most for ploidy 5, indicating the challenge in accurately predicting the absolute copy number for high-ploidy cells. Although the odd and intermediate copy number states are present in this simulated data, HMMcopy's precision and recall are still bimodally distributed for all ploidies. As we observed cells whose incorrect ploidy prediction led to wrong prediction of absolute copy numbers in the previous experiment, these bimodal distributions further showed that such wrong prediction can widely occur to cells with different ploidies.

Similar results were observed on the other two datasets (Additional file 1: Figures S7-S10).

The effect of coverage on performance

To evaluate the performance of each method under different single-cell sequencing technologies, we sampled 20 cells from the simulated dataset and simulated their sequencing at four levels of coverage variabilities, corresponding to MALBAC, DOP-PCR, TnBC and Bulk sequencing (see details in the "Methods" section) and ran the three methods on each of them. We generated three datasets for each level of variability. Fig. 4 show the performance on one of the datasets. With decreasing variability, all three methods' recall increased under both the coarse- and fine-grained analyses. Ginkgo's and HMMcopy's precisions increased with decreasing variability. CopyNumber's precision, on the other hand, stays the same regardless of the coverage variability level, whereas its recall generally increases. In summary, better sequencing technology leads to better performance. The best that can be ever obtained (Bulk sequencing) is about 15% higher than the worst (MALBAC) for recall, and slightly higher for precision.

We looked into the copy number profiles in cases where HMMcopy's precision and recall were effectively 0 (one such case is illustrated in Additional file 1: Figure S11). We found that choosing a wrong ploidy from the set of candidate ploidies by HMMcopy may result in a copy number profile in which all the segments are predicted to have the same absolute copy number, whereas the closest profile to the ground truth is among the reported non-optimal results. We observed that in such cases, the wrong choice of ploidy may affect both the segmentation and inference of the absolute copy number of those segments.

Similar results were observed on the other two repetitions (Additional file 1: Figures S12-S15).

Performance on a real dataset

In real data analysis, due to the lack of ground truth, we evaluated the performance of the three methods in two ways. First, we evaluated the consistency among the three methods. The more overlap among the methods, the more consistent they are. Second, we inferred a maximum parsimony tree using PAUP [52] from the inferred copy number profiles and calculated the smallest number of copy number changes for each bin across all branches of the tree, where the genome at the root of the tree is assumed to be diploid. The rationale for the latter way of assessing performance is that if the CNA's called by a method result (under a parsimony analysis) in a very large number of changes of the copy number at any bin, then one plausible explanation is error in the calls (another plausible prediction is that, for some reason, the locus corresponding to that bin has repeatedly gained and lost copies during the evolution of the cells which could be indicative of some interesting biological processes at play).

We downloaded single-cell DNA sequencing data of seven samples (the median number of cells in the seven samples is 47) from [37] and selected those pre-treatment samples whose CNA profiles had not changed due to treatment compared with mid-treatment and post-treatment ones. We then ran the three methods with default parameters (except for the *strength* parameter in HMMcopy, as discussed above) on the single cells in each sample.

For assessing accuracy, we generated for each sample a Venn diagram of the predictions based on all three methods, where predictions by two methods were deemed consistent according to the same rule we used in the simulation study (in assessing consistency between predictions and the ground truth). Fig. 5a shows the results for Sample 102 (Additional file 1: Figures S16-S18 show the results for the other six samples). It can be seen that 47% of Ginkgo's calls overlapped with the other two methods, leaving a large portion as unique calls. HMMcopy overlapped well with the other two methods, with 22% of unique calls. In particular, HMMcopy overlapped well with Ginkgo: 76% of HMMcopy's calls overlapped with Ginkgo. CopyNumber's overlap with Ginkgo was larger than that of HMMcopy (65% versus 43%). The overlap among the three methods is a very small portion of the union of all calls (8%), indicating a very large inconsistency among the three methods. From these results, we observe that HMMcopy performed the best among the three in breakpoint calling, if we consider consistency with other methods as a metric of quality, which is consistent with what we observed on the simulated data.

We then investigated the smallest number of changes required to explain the copy numbers detected by Ginkgo and HMMcopy (again, CopyNumber does not detect absolute copy numbers, which is why it is excluded in this analysis). Fig. 5b and Fig. 5c show the distributions of copy number changes based on HMMcopy and Ginkgo's results, respectively (Additional file 1: Figures S17-S18 show results for the other six samples). Interestingly, four out of seven samples (samples 102, 132, 152, and 302) showed a higher number of bins that have one copy number change than ones with no copy number changes. The other three samples (samples 126, 129, 615) have the most bins that had no copy number changes at all. Generally the number of bins that had copy number changes decreased with the increasing number of changes. On the other hand, based on the HMMcopy results, all samples

showed much higher percentage of no copy number change than those with some copy number change.

Conclusions

Computational methods for analyzing single-cell DNA data are currently being developed, yet their performance is not yet well understood. This is the case especially for methods for CNA detection. In this work, we benchmarked three methods that have been widely applied to single-cell data, namely CopyNumber, Ginkgo, and HMMcopy. We compared the three methods on simulated data generated under different settings that reflect varying degrees of complexity in the data. To accomplish this task, we developed a simulator that is flexible to simulate different scenarios and also mimic realistic data. We found that HMMcopy performs the best for breakpoint detection. However, HMMcopy is not stable in inferring the absolute copy number. Ginkgo performs well for both breakpoint detection and inference of the absolute copy number. CopyNumber is not as sensitive as the other two methods. We also looked into the performance of the three methods when ploidies were varied. We found that data with higher ploidies presented challenges for Ginkgo. HMMcopy is the most robust in terms of breakpoint detection among the three methods regardless of the ploidy, but its inference of the absolute copy number is not accurate for all ploidies. Both recall and precision of CopyNumber are the worst among the three methods. To explore the effect of technology artifacts on the accuracy of the methods, we simulated data that mimics the variability in coverage corresponding to MALBAC, DOP-PCR, TnBC, and Bulk. We found that all three methods' recall generally increases with the improvement in the technology, with smaller observed change in their precision.

We then applied the three methods to real data and evaluated their performance by analyzing the shared and unique detections they made as well as counting the total number of copy number changes must be invoked based on their detections. We found a good amount of overlap in detections between Ginkgo and HMMcopy. We also found that HMMcopy's detections result in fewer copy number changes than Ginkgo's.

In our simulations, we found that in the absence of odd or intermediate copy numbers, neither Ginkgo nor HMMcopy can correctly predict the absolute copy number. For example, in the absence of copy numbers 1, 3, and 5, copy numbers 2, 4 and 6 can be incorrectly inferred as 1, 3, and 5, respectively. Similarly, in the absence of copy numbers 1, 3, 4, 6, and 7, copy numbers 2, 5, and 8 can be incorrectly inferred as 1, 2, and 3, respectively. It is important to emphasize that while such issues may be present in real datasets, the computational tools we considered are limited in handling them. We also observed that HMMcopy may incorrectly infer the absolute copy number even in the presence of odd and intermediate copy numbers. This is due to the fact that HMMcopy incorrectly inferred the ploidy. We suggest that the users also take into account the intermediate results inferred from ploidies other than the chosen one when applying HMMcopy to their real data.

Another issue is that among the three methods, HMMcopy has the largest number of parameters for the user to set (or use the default values thereof). Following the command listed in [35], where the authors applied HMMcopy to their single-cell

data, while tuning parameters such as e and nu, resulted in odd behavior of the method (in terms of the shape of the ROC curve). We found out that the parameter *strength* needs to be set very high to result in a "standard" ROC curve.

For Ginkgo, we found that it may generate a few outlier bins with extremely high absolute copy number on the real dataset. We suspect this might be due to its read count correction step that takes into account the GC content and mappability of the region that coincides with the bin. In fact, in our real data analyses, we capped these extremely high copy numbers at 15 since this is the maximum number that PAUP can handle in such a parsimony analysis.

Methods

Simulation

Two steps are involved in simulating reads for single-cell sequencing. First, the cell tree is generated, where the nodes are the cells, and the edges represent the parent-daughter cell relation. The leaf nodes represent the single cells that are sampled from the patient; the internal nodes represent the cells that existed in the past and were not sampled. We set the root node as a normal genome without any CNA, assuming that all CNAs are somatic. The tree is simulated by the Beta-Splitting model (see below), which allows producing imbalanced trees, consistent with what was observed in the real data [38].

On each edge (except for those attached to the root of the tree; see below), we simulate a number of CNAs, the number of which corresponds to a Poisson distribution (by default, $\lambda = 2$). λ relates with the mutation rate which has been studied for two decades [53,54]. There has not yet been a comprehensive knowledge of the mutation rate of CNAs, but according to the data from [37], we found there are about several dozens of CNAs in this data set. The same can be found in a pan-cancer study [55]. Setting default λ to be 2 will lead to the similar number of CNAs at the leaves for a tree. The daughter cell of the edge inherits all CNAs in the parent node, in addition to its unique CNAs. To simulate a CNA, we randomly choose the allele, and the chromosome and position on the allele that CNA is going to occur. First, we sample the allele on which the CNA is going to occur from the paternal and maternal alleles according to a binomial distribution (default $p = 0.5$). We designed the simulator in a framework which keeps track of the allele at which the CNA occurs so that in our future work of simulating single nucleotide variations (SNV) simultaneously, the allele that is dropped due to the high allelic dropout rate can be traced. For the CNA size, we sample from an exponential distribution (default mean=5Mbp), plus a minimum CNA size (default 2Mbp). We set the minimum CNA size by default to be 2Mbp because these CNAs are rare and commonly associated with disease [56], and also because of the limited resolution of of single-cell data. The exponential distribution with mean 5Mbp is to render a wide range of CNA size. According to [32,57], the larger the CNA size, the smaller the CNA's possibility. We choose copy number gain versus loss by a binomial distribution (default $p = 0.5$). We set the default parameter to be 0.5 so that copy number gain and loss are equally distributed. If a copy number gain is sampled, we sample from a geometric distribution (default $p = 0.5$) to determine the number of copies to be gained (mean=$1/p$). This choice of a distribution is motivated by the observation

that extremely high copy number gains are very rare and are often observed by double minutes amplification [58], which we do not take into account currently. Once a whole-genome DNA sequence is simulated with the CNAs, the gained copies are placed in tandem with the original copy. If a copy number loss is sampled, the whole sequence on that region of the allele is deleted.

The CNAs on the edges attached to the root node are simulated differently. In particular, clonal whole chromosomal amplifications can occur on these edge, as indicated in the punctuated evolution model observed in [38]. We simulate the chromosomal amplifications in addition to the focal CNAs. We set the probability of a chromosome to be amplified to be according to a binomial distribution (default $p = 0.2$). This default value is used so that while the whole chromosome amplification is introduced, 20% chromosomes in the genome will be changed. The number of the amplified copy is sampled from a geometric distribution (default mean is $p = 1$) multiplied by a value (default is 1) to amplify the copy numbers simulated without changing the distribution. The distribution of the whole chromosomal amplification can be turned off as an option.

At the edge to the root, we also add an option to allow more CNAs than the other edges. This is again to mimic a scenario of punctuated evolution [38]. To do that, we sample a value from a Poisson distribution (by default, $\lambda = 4$) which is the multiplier of the average number of the CNAs that occur to the edges other than the root. Thus the edge to the root has on average 4 times (default parameter) more focal CNAs than those of other edges. The higher this number, the more focal CNAs the edge to the root carries. This parameter is introduced to allow the user to simulate data that mimics the punctuated evolution model. However, due to the diversity of models that have been summarized for cancer evolution [59], users can turn off this option or tune the parameter λ so that the simulated data corresponds to their observation and experience. In our study, the value of λ was chosen according to the length of the trunk observed in Fig. 6 in [38].

Once we have the tree and the DNA sequences for all leaf nodes, we simulate the generation of read data from the genomes. Given the coverage of the genome (by default 0.04X), the simulator divides the genome into non-overlapping bins each of which has a default size of 200,000bp. To simulate the coverage variability observed in single-cell data, we use a Markov Chain Monte Carlo (MCMC) Metropolis-Hastings algorithm to determine a sequence of numbers of read pairs to be sampled for each bin.

An input of the variability information is a point on the Lorenz curve, whose X axis represents the percentage of the reads, and Y axis represents the percentage of the coverage. We transform it to a Beta distribution by Equations (1) and (2) in [60]. Through this transformation, we can sample read counts from a Beta distribution that corresponds to the given Lorenz curve. The followings are mathematical equations in [60] that are used to calculate the parameters (α and β) for the Beta distribution. In more detail, suppose X is a random variable whose cumulative distribution function F corresponds to a Beta distribution with parameters α and β. A point x sampled from this distribution has its corresponding X and Y positions on the Lorenz curve as $F(x)$ and $\phi(x)$, where

$$F(x) = I_x(\alpha, \beta) \tag{1}$$

and

$$\phi(x) = I_x(\alpha + 1, \beta) \tag{2}$$

Given a point $(F(x), \phi(x))$ on the Lorenz curve, we can calculate α and β for the Beta distribution.

Given the Beta distribution's parameters, we can then sample read count for each bin by MCMC Metropolis-Hastings algorithm. Starting from the first bin whose read count is assigned as the expected coverage x_0, we sample the next bin's proposed read count x' by a Gaussian distribution, and accept it if compared with the previous bin's read count x_0,

$$\frac{I_{x'}(\alpha, \beta) \times \mathbf{Gaus}(x'|x_0)}{I_{x_0}(\alpha, \beta) \times \mathbf{Gaus}(x_0|x')} \leq u \tag{3}$$

where $\mathbf{Gaus}(x'|x_0)$ is the proposal probability of proposing x' given x_0, and u is the acceptance ratio. We set u to be 0.5 by default. We set the same standard deviations for $\mathbf{Gaus}(x'|x_0)$ and $\mathbf{Gaus}(x_0|x')$, centered at x_0 and x', respectively. Thus the two Gaussian distribution canceled out. The rest term, $I_{x'}/I_{x_0} \leq u$, controls how much the next bin's read count differs from the current one. The read counts drawn are thus corresponding to a Beta distribution, and are simultaneously constrained by the acceptance ratio of the Metropolis-Hastings algorithm. This is to mimic the realistic data whose read coverage fluctuates, but the read count changes smoothly without sharp changes between neighboring bins.

Running the programs

In all experiments, we eliminated reads that have mapping quality score < 40. We eliminated the cells that HMMcopy predicted as normal cells (predicted to be diploid and found no copy number aberration) in all experiments, the percentage of which was very small ($< 0.2\%$).

Parameters of simulator

The simulator is designed to be flexible, with user-specified parameters, as now describe.

Parameters for varying the ploidy level

To generate data with different ploidies, parameters associated with whole chromosomal amplification can be set for that purpose.

- **-W (–whole-amp)** Controls whether there are whole chromosomal amplifications or not.
- **-d (–del-rate)** The rate of copy number loss versus copy number gain.
- **-C (–whole-amp-rate)** The possibility that a chromosome is selected to have whole chromosomal amplification.
- **-E (–whole-amp-num)** For those chromosomes that are selected to be amplified, multiplying this number with the sampled value from a geometric distribution, whose p is "-J" described below, renders the final number of copies to be amplified.

- **-J (–amp-num-geo-par)** The parameter p in the geometric distribution from which the number of copy of the chromosome to be amplified is sampled. Combination of **-J** and **-E** can make a variety of copy number distributions and make it convenient to attain higher copy number gains when necessary.

In our experiment above where we varied the ploidy level, we use a combination of these five parameters to generate data whose ploidies range from 1.5 to 5 as follows.

- **Ploidy = 1.5**: **-W 0 -d 1** No amplifications are allowed, and all copy number aberrations come from deletion.
- **Ploidy = 3**: **-W 1 -d 0.5 -C 0.5 -E 1 -J 1** Amplification is allowed, and the average number of amplification for the whole genome is 0.5 for one allele. The final ploidy is 3.
- **Ploidy = 4**, the case that lacks odd copy numbers: **-W 1 -d 0.5 -C 0.5 -E 2 -J 1** Amplification is allowed, and the average number of amplification for the whole genome is 1 for one allele. The final ploidy is 4. Note that since the parameter p in the geometric distribution (-J) is set to be one, the copy number is amplified by two for the allele that is selected for amplification. This causes the lack of intermediate copy numbers such as three, five, etc.
- **Ploidy = 3.8**, the case that has odd copy numbers: **-W 1 -d 0.5 -C 0.9 -E 1 -J 1** Amplification is allowed, and the average number of amplification for the whole genome is 0.9 for one allele. The final ploidy is 3.8. Compared with the previous case which lacks odd copy numbers, we increase the copy number by doubling 90% of the chromosomes. The following local copy number aberrations that are performed based on the amplified genome will then generate regions that have different copies, including the odd copies. In the absence of odd copy numbers, copy numbers 2, 4 and 6 will be considered as 1, 2, and 3 by any method. Thus, without copy numbers 1, 3 and 5, there is no way for a method to tell the correct absolute copy number.
- **Ploidy = 5**, the case that lacks intermediate copy numbers: **-W 1 -d 0.5 -C 0.5 -E 3 -J 1** Amplification is allowed, and the average number of amplifications for the whole genome is 1 for one allele. The final ploidy is 5. Note that since the parameter p in the geometric distribution (-J) is set to be one, the copy number is amplified by three for the allele that is selected for amplification. This causes a scenario where most of the copy numbers are two, five and eight.
- **Ploidy = 5.26**, the case that has intermediate copy numbers: **-W 1 -d 0.5 -C 0.9 -E 1 -J 0.55** Amplification is allowed, and the average number of amplification for the whole genome is 0.9 for one allele. Setting parameter J to be 0.55, the total amplified copy number for each allele is 1.63 (from $1/p \times 0.9$). The final ploidy is 5.26.

Parameters for varying the read count distribution

Since Lorenz curves have been used to evaluate the variability of read counts [35, 49, 61], we used the Lorenz curves reported in [49] for simulating variabilities at different levels. We sampled the read counts for each bin by the distribution (Beta distribution) corresponding to their Lorenz curves using a Markov Chain Metropolis-Hastings method (Additional file 1: Figure S19 shows the Lorenz curves

(left panel) and their corresponding Beta distributions (right panel) for the four technologies. The key parameters used for the Lorenz curves and Beta distributions corresponding to the four technologies are shown within the panels.).

The Beta-splitting model

For generating the underlying evolutionary trees, we followed a generalization of the Blum-François Beta-splitting model [62] which is inspired by Aldous' Beta-splitting model [63]. The construction of a tree based on this model [64] consists of two major steps: First, we generate two sequences of random values: $B = (b_1, b_2, \cdots)$ and $U = (u_1, u_2, \cdots)$, B is a sequence of independent and identically distributed (i.i.d.) random variables sampled from the $\mathscr{B}(\alpha+1, \beta+1)$ distribution, and, U is a sequence of i.i.d. random variables with the uniform distribution on $[0, 1]$. We call $\{g_i = (u_i, b_i)_{i\in\mathbb{N}}\}$ the *generating sequence* which is the basis of incremental construction of a tree. At the second step, we run the following algorithm on the random values generated at the first step. The process of constructing an evolutionary tree for n cells/leaves based on the Beta-splitting model with the parameters (α, β) combining these two steps is described in the following pseudocode.

Algorithm 1 Algorithm for constructing a tree $\mathscr{T}$ with n leaves with the Beta parameters α and β.

function BUILDBETASPLITTINGTREE(n, α, β)
  Create the root of $\mathscr{T}$
  $\mathscr{T}.root.label \leftarrow (0, 1)$
  for $i = 1 \cdots n$ **do**
    Sample $b_i \sim \mathscr{B}(\alpha+1, \beta+1)$
    Sample $u_i \sim U(0, 1)$
    for each $leaf \in \mathscr{T}.leaves$ **do**
      $(x, y) \leftarrow leaf.label$
      if $u_i \in [x, y]$ **then**
        $l \leftarrow$ Create the left child of $leaf$
        $r \leftarrow$ Create the right child of $leaf$
        $r.label \leftarrow (x + (y-x)b_i, y)$
        $l.label \leftarrow (x, x + (y-x)b_i)$
        $leaf.label \leftarrow i$
      end if
    end for
  end for
  return $\mathscr{T}$
end function

Software availability

The simulator has been implemented in Python and is freely available at https://bitbucket.org/xianfan/cnsc_simulator/src/master/, which also includes the scripts for regenerating the comparison results for both simulated and real datasets.

Availability of data and materials

The new version of HMMcopy was downloaded from https://github.com/shahcompbio/single_cell_pipeline/tree/master/single_cell/workflows/hmmcopy. The scripts to preprocess files for HMMcopy were downloaded from https://shahlab.ca/projects/hmmcopy_utils/. We use hg19 for all experiments in this manuscript and the mappability file used by HMMcopy was downloaded from http://genome.ucsc.edu/cgi-bin/hgFileUi?db=hg19&g=wgEncodeMapability. CopyNumber was downloaded from

https://bioconductor.org/packages/release/bioc/html/copynumber.html. Ginkgo's command line version which was used in this manuscript was downloaded from https://github.com/robertaboukhalil/ginkgo.

The real biological dataset that we analyzed is available from NCBI Sequence Read Archive under accession SRP114962.

Glossary

- **Segmentation** Computationally segmenting the genome into non-overlapping regions so that each region has a homogeneous copy number.
- **Boundary** and **Breakpoint** Positions on the genome where segmentation occurs.
- **Absolute Copy Number** The integer value representing the number of copies of a region on the genome.
- **Ploidy** The average copy number across the genome.

Acknowledgement

The authors acknowledge the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC resources that have contributed to the research results reported within this paper. We thank Dr. Ruli Gao for her help in explaining the real dataset, and Dr. Alexander Davis for his insights on the ploidy of the single cells.

Competing interests

The authors declare that they have no competing interests.

Author's contributions

XF, ME, NN, and LN designed the study. XF and ME wrote the code and ran the experiments. All authors wrote and approved the manuscript.

Funding

The study was supported by the National Science Foundation grant IIS-1812822 (L.N.). X.F. was supported in part by a Computational Cancer Biology Training Program (CPRIT Grant No. RP170593).

Author details

[1]Department of Computer Science, Rice University, Houston, Texas, USA. [2]Department of Genetics, the University of Texas M.D. Anderson Cancer Center, Houston, Texas, USA. [3]Department of Bioinformatics and Computational Biology, the University of Texas M.D. Anderson Cancer Center, Houston, Texas, USA.

References

1. Feuk, L., Carson, A.R., Scherer, S.W.: Structural variation in the human genome. Nature Reviews Genetics **7**(2), 85 (2006)
2. Sharp, A.J., Cheng, Z., Eichler, E.E.: Structural variation of the human genome. Annu. Rev. Genomics Hum. Genet. **7**, 407–442 (2006)
3. Lupski, J.R., *et al.*: Structural variation in the human genome. New England Journal of Medicine **356**(11), 1169 (2007)
4. Beroukhim, R., Mermel, C.H., Porter, D., Wei, G., Raychaudhuri, S., Donovan, J., Barretina, J., Boehm, J.S., Dobson, J., Urashima, M., *et al.*: The landscape of somatic copy-number alteration across human cancers. Nature **463**(7283), 899 (2010)
5. Li, W., Olivier, M.: Current analysis platforms and methods for detecting copy number variation. Physiological genomics **45**(1), 1–16 (2012)
6. Carter, N.P.: Methods and strategies for analyzing copy number variation using dna microarrays. Nature genetics **39**(7s), 16 (2007)
7. Olshen, A.B., Venkatraman, E., Lucito, R., Wigler, M.: Circular binary segmentation for the analysis of array-based dna copy number data. Biostatistics **5**(4), 557–572 (2004)
8. Shah, S.P., Xuan, X., DeLeeuw, R.J., Khojasteh, M., Lam, W.L., Ng, R., Murphy, K.P.: Integrating copy number polymorphisms into array cgh analysis using a robust hmm. Bioinformatics **22**(14), 431–439 (2006)
9. Ha, G., Roth, A., Lai, D., Bashashati, A., Ding, J., Goya, R., Giuliany, R., Rosner, J., Oloumi, A., Shumansky, K., *et al.*: Integrative analysis of genome-wide loss of heterozygosity and monoallelic expression at nucleotide resolution reveals disrupted pathways in triple-negative breast cancer. Genome research **22**(10), 1995–2007 (2012)
10. Sen, A., Srivastava, M.S.: On tests for detecting change in mean. The Annals of Statistics **3**(1), 98–108 (1975)
11. Czyż, Z.T., Hoffmann, M., Schlimok, G., Polzer, B., Klein, C.A.: Reliable single cell array cgh for clinical samples. PloS one **9**(1), 85907 (2014)
12. Mosen-Ansorena, D., Telleria, N., Veganzones, S., De la Orden, V., Maestro, M.L., Aransay, A.M.: seqcna: an r package for dna copy number analysis in cancer using high-throughput sequencing. BMC genomics **15**(1), 178 (2014)
13. Jang, H., Lee, H.: Multiresolution correction of gc bias and application to identification of copy number alterations. Bioinformatics (2019)
14. Ivakhno, S., Royce, T., Cox, A.J., Evers, D.J., Cheetham, R.K., Tavaré, S.: Cnaseg—a novel framework for identification of copy number changes in cancer from second-generation sequencing data. Bioinformatics **26**(24), 3051–3058 (2010)

15. Gusnanto, A., Wood, H.M., Pawitan, Y., Rabbitts, P., Berri, S.: Correcting for cancer genome size and tumour cell content enables better estimation of copy number alterations from next-generation sequence data. Bioinformatics **28**(1), 40–47 (2011)
16. Boeva, V., Popova, T., Bleakley, K., Chiche, P., Cappo, J., Schleiermacher, G., Janoueix-Lerosey, I., Delattre, O., Barillot, E.: Control-freec: a tool for assessing copy number and allelic content using next-generation sequencing data. Bioinformatics **28**(3), 423–425 (2011)
17. Medvedev, P., Fiume, M., Dzamba, M., Smith, T., Brudno, M.: Detecting copy number variation with mated short reads. Genome research **20**(11), 1613–1622 (2010)
18. Abyzov, A., Urban, A.E., Snyder, M., Gerstein, M.: Cnvnator: an approach to discover, genotype, and characterize typical and atypical cnvs from family and population genome sequencing. Genome research **21**(6), 974–984 (2011)
19. Miller, C.A., Hampton, O., Coarfa, C., Milosavljevic, A.: Readdepth: a parallel r package for detecting copy number alterations from short sequencing reads. PloS one **6**(1), 16327 (2011)
20. Malekpour, S.A., Pezeshk, H., Sadeghi, M.: Mseq-cnv: accurate detection of copy number variation from sequencing of multiple samples. Scientific reports **8**(1), 4009 (2018)
21. Nowell, P.C.: The clonal evolution of tumor cell populations. Science **194**(4260), 23–28 (1976)
22. Navin, N.E.: Cancer genomics: one cell at a time. Genome biology **15**(8), 452 (2014)
23. Navin, N., Kendall, J., Troge, J., Andrews, P., Rodgers, L., McIndoo, J., Cook, K., Stepansky, A., Levy, D., Esposito, D., *et al.*: Tumour evolution inferred by single-cell sequencing. Nature **472**(7341), 90 (2011)
24. Carter, N.P., Bebb, C.E., Nordenskjo, M., Ponder, B.A., Tunnacliffe, A., *et al.*: Degenerate oligonucleotide-primed pcr: general amplification of target dna by a single degenerate primer. Genomics **13**(3), 718–725 (1992)
25. Arneson, N., Hughes, S., Houlston, R., Done, S.: Whole-genome amplification by degenerate oligonucleotide primed pcr (dop-pcr). Cold Spring Harbor Protocols **2008**(1), 4919 (2008)
26. Baslan, T., Kendall, J., Rodgers, L., Cox, H., Riggs, M., Stepansky, A., Troge, J., Ravi, K., Esposito, D., Lakshmi, B., *et al.*: Genome-wide copy number analysis of single cells. Nature protocols **7**(6), 1024 (2012)
27. Bakker, B., Taudt, A., Belderbos, M.E., Porubsky, D., Spierings, D.C., de Jong, T.V., Halsema, N., Kazemier, H.G., Hoekstra-Wakker, K., Bradley, A., *et al.*: Single-cell sequencing reveals karyotype heterogeneity in murine and human malignancies. Genome biology **17**(1), 115 (2016)
28. van den Bos, H., Spierings, D.C., Taudt, A., Bakker, B., Porubskỳ, D., Falconer, E., Novoa, C., Halsema, N., Kazemier, H.G., Hoekstra-Wakker, K., *et al.*: Single-cell whole genome sequencing reveals no evidence for common aneuploidy in normal and alzheimer's disease neurons. Genome biology **17**(1), 116 (2016)
29. Nilsen, G., Liestøl, K., Van Loo, P., Vollan, H.K.M., Eide, M.B., Rueda, O.M., Chin, S.-F., Russell, R., Baumbusch, L.O., Caldas, C., *et al.*: Copynumber: efficient algorithms for single-and multi-track copy number segmentation. BMC genomics **13**(1), 591 (2012)
30. Garvin, T., Aboukhalil, R., Kendall, J., Baslan, T., Atwal, G.S., Hicks, J., Wigler, M., Schatz, M.C.: Interactive analysis and assessment of single-cell copy-number variations. Nature methods **12**(11), 1058 (2015)
31. Zahn, H., Steif, A., Laks, E., Eirew, P., VanInsberghe, M., Shah, S.P., Aparicio, S., Hansen, C.L.: Scalable whole-genome single-cell library preparation without preamplification. Nature methods **14**(2), 167 (2017)
32. Knouse, K.A., Wu, J., Amon, A.: Assessment of megabase-scale somatic copy number variation using single-cell sequencing. Genome research **26**(3), 376–384 (2016)
33. Harmanci, A.S., Harmanci, A.O., Zhou, X.: Casper: Identification, visualization and integrative analysis of cnv events in multiscale resolution using single-cell or bulk rna sequencing data. bioRxiv, 426122 (2018)
34. Wang, Y., Guo, L., Feng, L., Zhang, W., Xiao, T., Di, X., Chen, G., Zhang, K.: Single nucleotide variant profiles of viable single circulating tumour cells reveal ctc behaviours in breast cancer. Oncology reports **39**(5), 2147–2159 (2018)
35. Laks, E., Zahn, H., Lai, D., McPherson, A., Steif, A., Brimhall, J., Biele, J., Wang, B., Masud, T., Grewal, D., et al.: Resource: Scalable whole genome sequencing of 40,000 single cells identifies stochastic aneuploidies, genome replication states and clonal repertoires. bioRxiv, 411058 (2018)
36. Wang, X., Chen, H., Zhang, N.R.: Dna copy number profiling using single-cell sequencing. Briefings in bioinformatics **19**(5), 731–736 (2017)
37. Kim, C., Gao, R., Sei, E., Brandt, R., Hartman, J., Hatschek, T., Crosetto, N., Foukakis, T., Navin, N.E.: Chemoresistance evolution in triple-negative breast cancer delineated by single-cell sequencing. Cell **173**(4), 879–893 (2018)
38. Gao, R., Davis, A., McDonald, T.O., Sei, E., Shi, X., Wang, Y., Tsai, P.-C., Casasent, A., Waters, J., Zhang, H., *et al.*: Punctuated copy number evolution and clonal stasis in triple-negative breast cancer. Nature genetics **48**(10), 1119 (2016)
39. Vitak, S.A., Torkenczy, K.A., Rosenkrantz, J.L., Fields, A.J., Christiansen, L., Wong, M.H., Carbone, L., Steemers, F.J., Adey, A.: Sequencing thousands of single-cell genomes with combinatorial indexing. Nature methods **14**(3), 302 (2017)
40. Xue, Y., Martelotto, L., Baslan, T., Vides, A., Solomon, M., Mai, T.T., Chaudhary, N., Riely, G.J., Li, B.T., Scott, K., *et al.*: An approach to suppress the evolution of resistance in braf v600e-mutant cancer. Nature medicine **23**(8), 929 (2017)
41. Martelotto, L.G., Baslan, T., Kendall, J., Geyer, F.C., Burke, K.A., Spraggon, L., Piscuoglio, S., Chadalavada, K., Nanjangud, G., Ng, C.K., *et al.*: Whole-genome single-cell copy number profiling from formalin-fixed paraffin-embedded samples. Nature medicine **23**(3), 376 (2017)
42. Macaulay, I.C., Teng, M.J., Haerty, W., Kumar, P., Ponting, C.P., Voet, T.: Separation and parallel sequencing of the genomes and transcriptomes of single cells using g&t-seq. Nature protocols **11**(11), 2081 (2016)
43. Ortega, M.A., Poirion, O., Zhu, X., Huang, S., Wolfgruber, T.K., Sebra, R., Garmire, L.X.: Using single-cell multiple omics approaches to resolve tumor heterogeneity. Clinical and translational medicine **6**(1), 46 (2017)
44. Sho, S., Winograd, P., Lee, S., Hou, S., Graeber, T.G., Tseng, H.-R., Tomlinson, J.S., *et al.*: Precision oncology using a limited number of cells: optimization of whole genome amplification products for sequencing

applications. BMC cancer **17**(1), 457 (2017)
45. Pal, D., Pertot, A., Shirole, N.H., Yao, Z., Anaparthy, N., Garvin, T., Cox, H., Chang, K., Rollins, F., Kendall, J., *et al.*: Tgf-β reduces dna ds-break repair mechanisms to heighten genetic diversity and adaptability of cd44+/cd24- cancer cells. Elife **6**, 21615 (2017)
46. Wang, L., Livak, K.J., Wu, C.J.: High-dimension single-cell analysis applied to cancer. Molecular aspects of medicine **59**, 70–84 (2018)
47. Liu, J., Adhav, R., Xu, X.: Current progresses of single cell dna sequencing in breast cancer research. International journal of biological sciences **13**(8), 949 (2017)
48. Alves, J.M., Posada, D.: Sensitivity to sequencing depth in single-cell cancer genomics. Genome medicine **10**(1), 29 (2018)
49. Xi, L., Belyaev, A., Spurgeon, S., Wang, X., Gong, H., Aboukhalil, R., Fekete, R.: New library construction method for single-cell genomes. PloS one **12**(7), 0181163 (2017)
50. Li, H., Durbin, R.: Fast and accurate short read alignment with burrows–wheeler transform. bioinformatics **25**(14), 1754–1760 (2009)
51. Li, H.: Aligning sequence reads, clone sequences and assembly contigs with bwa-mem. arXiv preprint arXiv:1303.3997 (2013)
52. Swofford, D.L.: Phylogenetic analysis using parsimony (1998)
53. Martincorena, I., Raine, K.M., Gerstung, M., Dawson, K.J., Haase, K., Van Loo, P., Davies, H., Stratton, M.R., Campbell, P.J.: Universal patterns of selection in cancer and somatic tissues. Cell **173**(7), 1823 (2018)
54. Tomlinson, I.P., Novelli, M., Bodmer, W.: The mutation rate and cancer. Proceedings of the National Academy of Sciences **93**(25), 14800–14803 (1996)
55. Zack, T.I., Schumacher, S.E., Carter, S.L., Cherniack, A.D., Saksena, G., Tabak, B., Lawrence, M.S., Zhang, C.-Z., Wala, J., Mermel, C.H., *et al.*: Pan-cancer patterns of somatic copy number alteration. Nature genetics **45**(10), 1134 (2013)
56. Girirajan, S., Campbell, C.D., Eichler, E.E.: Human copy number variation and complex genetic disease. Annual review of genetics **45**, 203–226 (2011)
57. Krijgsman, O., Carvalho, B., Meijer, G.A., Steenbergen, R.D., Ylstra, B.: Focal chromosomal copy number aberrations in cancer—needles in a genome haystack. Biochimica et Biophysica Acta (BBA)-Molecular Cell Research **1843**(11), 2698–2704 (2014)
58. Cowell, J.K.: Double minutes and homogeneously staining regions: gene amplification in mammalian cells. Annual review of genetics **16**(1), 21–59 (1982)
59. Davis, A., Gao, R., Navin, N.: Tumor evolution: Linear, branching, neutral or punctuated? Biochimica et Biophysica Acta (BBA)-Reviews on Cancer **1867**(2), 151–161 (2017)
60. Pham-Gia, T., Turkkan, N.: Determination of the beta distribution form its lorenz curve. Mathematical and computer modelling **16**(2), 73–84 (1992)
61. Wang, Y., Waters, J., Leung, M.L., Unruh, A., Roh, W., Shi, X., Chen, K., Scheet, P., Vattathil, S., Liang, H., *et al.*: Clonal evolution in breast cancer revealed by single nucleus genome sequencing. Nature **512**(7513), 155 (2014)
62. Blum, M., François, O.: Which random processes describe the tree of life? a large-scale study of phylogenetic tree imbalance. Systems biology **55**(8), 685–691 (2006)
63. Aldous, D.: Probability distributions on cladograms. in *Random discrete structure*, 1–18 (1996)
64. Sainudiin, R., Véber, A.: A beta splitting model for evolutionary trees. Royal Society open science **3**(160016) (2016)

Figures

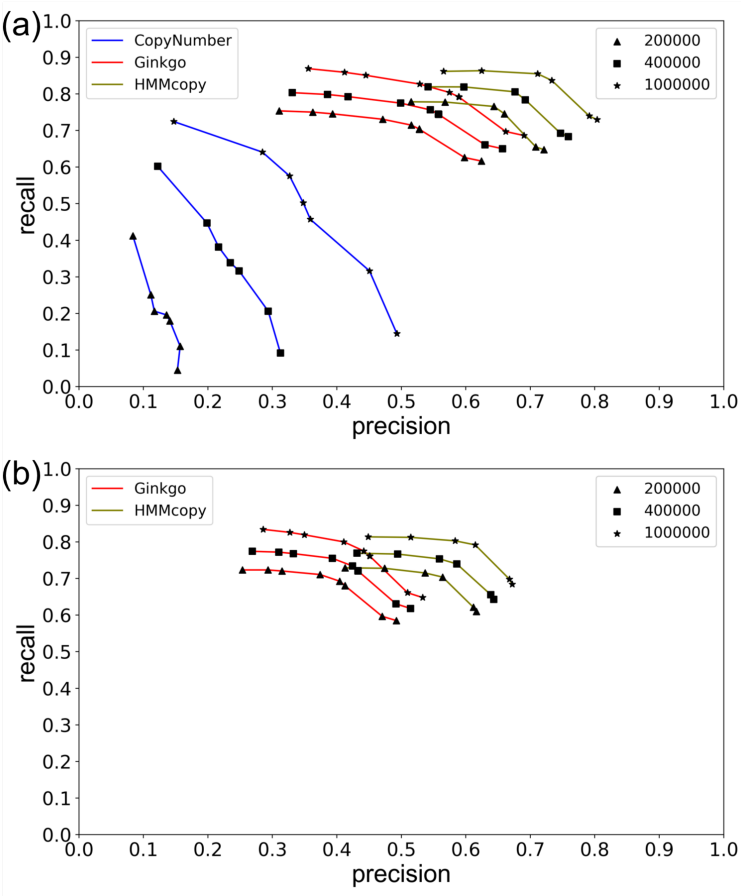

Figure 1 ROC curves of the three methods HMMcopy, Ginkgo, and CopyNumber. (a) Coarse-grained analysis results, and (b) fine-grained results. For each method, the results based on three thresholds of correctness are plotted. For HMMcopy, nu, which controls the suggested degree of freedom between states, was tuned to take on the values 0.01 (rightmost), 0.1, 2.1 (the tool's default), 4, 10, and 20 (leftmost). For Ginkgo, $alpha$, which controls the significance level to accept a change point, was tuned to take on the values 1e-1000 (rightmost), 1e-100, 1e-10, 1e-5, 1e-4, 1e-3, 1e-2 (the tool's default), 0.02, and 0.05 (leftmost). The dots corresponding to values 1e-5 and 1e-10 in coarse-grained analysis overlap. For CopyNumber, $gamma$, which is the weight of the penalty on changing a state, was tuned to take on the values 40 (rightmost, and the tool's default), 10, 5, 4, 3, 2, and 1 (leftmost).

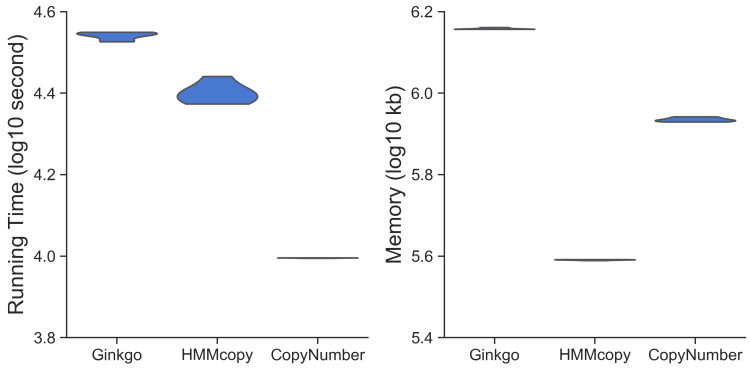

Figure 2 Computational requirements of Ginkgo, HMMcopy, and CopyNumber for analyzing a 1000-cell dataset on Intel(R) Xeon(R) CPU E5-2650 v2 whose clock speed is 2.60GHz. Left and right panels correspond to running time (in log10 of seconds) and memory consumption (in log10 of kb). The running time and memory were recorded for using different parameters as described in Fig. 1. As Ginkgo's running time increases more than twofold for $\alpha = 0.05$, we treated it as an outlier and did not include this running time point in this plot.

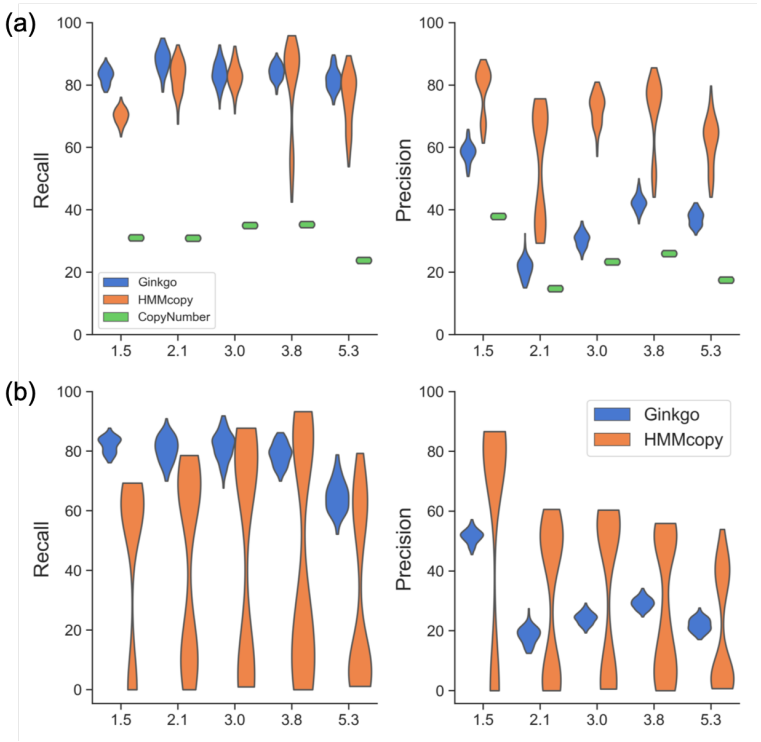

Figure 3 Recall and Precision of Ginkgo, HMMcopy, and CopyNumber on varying ploidy level based on the (a) coarse-grained (b) fine-grained analysis. The ploidies of the simulated data were 1.5, 2.1, 3.0, 3.8, and 5.3.

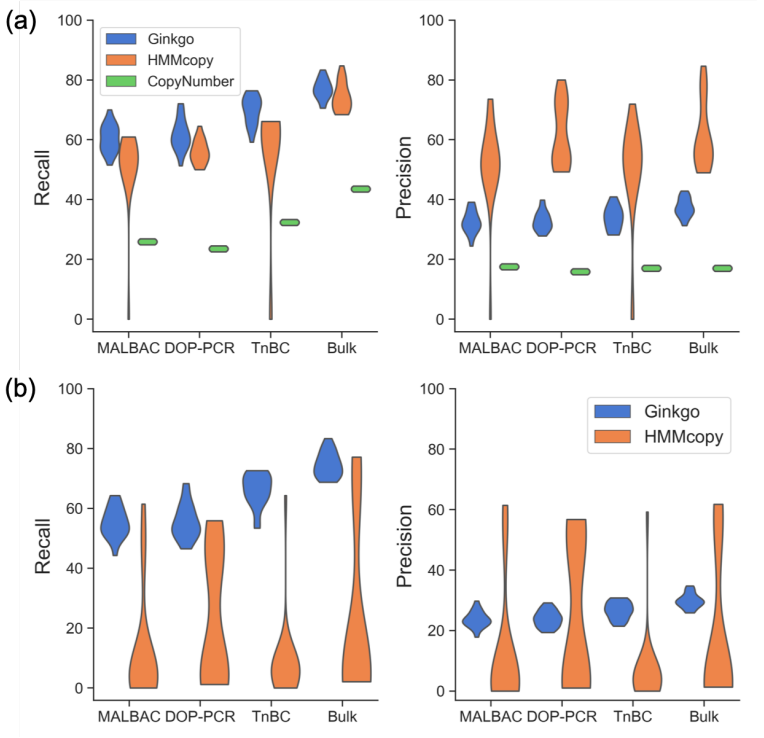

Figure 4 Recall and Precision of Ginkgo, HMMcopy, and CopyNumber on varying coverages based on the (a) coarse-grained (b) fine-grained analysis. The coverages are varied to mimic those produced by MALBAC, DOP-PCR, TnBC and Bulk sequencing.

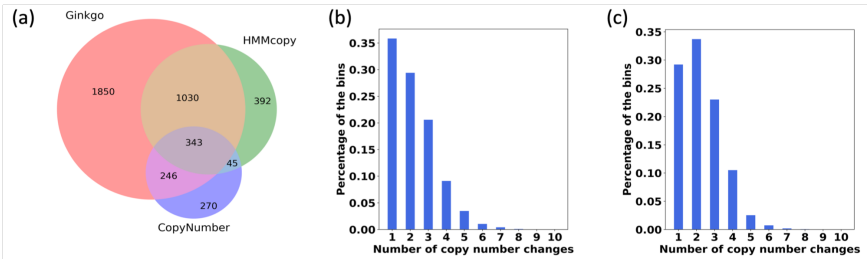

Figure 5 Comparison of HMMcopy, Ginkgo and CopyNumber on Sample 102 in [37]. (a) Venn diagram of the breakpoints from Ginkgo, HMMcopy and CopyNumber. Breakpoints from two methods are counted as overlapping if they are within 400,000bp of each other. (b) Distribution of the copy number changes (under a parsimony analysis) per bin based on the copy number profiles obtained by HMMcopy for the seven samples. (c) Distribution of the copy number changes (under a parsimony analysis) per bin based on the copy number profiles obtained by Ginkgo for the seven samples. For (b) and (c), a maximum parsimony tree was inferred from the copy number profiles of the cells, and the minimum number of copy number changes per bin along all branches of the tree was computed by parsimony analysis. The percentages of bins with each number of copy number changes are plotted.