SimulateCNVs: a novel software application for simulating CNVs in WES and WGS data

Yue Xing[1,*], Alan R. Dabney[2], Xiao Li[3] and Claudio Casola[4,*]

[1]Interdisciplinary Program of Genetics, Texas A&M University, College Station TX 77843, [2]Department of Statistics, Texas A&M University, College Station TX 77843, [3]Department of Molecular and Cellular Medicine, Texas A&M University, College Station TX 77843, [4]Department of Ecosystem Science and Management, Texas A&M University, College Station TX 77843

*To whom correspondence should be addressed.

Abstract

Summary: SimulateCNVs is a fast and novel software application for simulating CNVs in WES and WGS data. Current NGS simulators can not simulate rearranged genomes and their alignment files for WES data and are not easy to use. Here, we present SimulateCNVs, the first software application that can simulate CNVs in both WES and WGS data, which is easy to install, has many unique features and can output rearranged genomes, short reads and bam files in a single command.

Availability: SimulateCNVs is freely available from https://yjulyxing.github.io/

Contact: inumia@tamu.edu, ccasola@tamu.edu

Copy Number Variants (CNVs) are insertions and deletions of 1 kb or larger in a genome, a type of structural variants that has been associated with phenotypic changes, including disease in human (Feuk, et al., 2006). Increasingly, next generation sequencing (NGS) data are interrogated to detect CNVs. Many software applications have been developed to detect CNVs using either whole-genome sequencing (WGS) [2] or whole-exome sequencing (WES) [3] data. Most of these methods are based on read depth, although approaches relying on read pairs distance, detection of split reads or de novo genome assemblies have also been developed for WGS data [2]. Combinations of some of these approaches have also been incorporated in CNVs detection programs [2]. Overall, these applications show disagreement in their results due to a variety of issues. Simulations of copy-number variants in WGS and WGE datasets allow to compare the performance of existing and novel CNVs detection algorithm.

There are several existing software applications for simulating CNVs and other structural variants in next generation sequencing (NGS) data, including RSVSim (Bartenhagen and Dugas, 2013), SCNVSim (Qin, et al., 2015), Pysim-sv (Xia, et al., 2017), SVsim (Faust, 2017) and SInC (Pattnaik, et al., 2014). However, these applications are not appropriate for CNVs simulation in the context of WES data and present several limitations when used to generate CNVs in their current versions. In WES data, specific regions of a reference genome, called “target regions”, are captured and sequenced (Goh and Choi, 2012). To reproduce a realistic distribution of structural variants, a CNVs simulator for WES data should generate variants that overlap partly or entirely with one or multiple target regions (Fig. 1). These regions should be included in a separate sequence file, called here “test” genome. Short reads are then produced from both control (same as the reference genome) and test genomes and aligned back to the reference genome. Insertions and deletions in the test genome will appear as an increased read coverage or a lack of coverage, respectively, in the reads alignment file (Fig. 1). Some programs mentioned above can simulate CNVs in selected regions, but they lack the capacity to rearrange the target regions in the rearranged genome according to simulated CNVs as shown in figure 1. These applications cannot be used to simulate WES datasets because they do not provide the locations of the rearranged target regions, which are necessary to synthesize short reads. In short, these applications are not capable for simulating WES datasets. Also, these programs lack the capacity to produce CNVs that overlap partly or entirely with one or multiple target regions.

One application is available to simulate CNVs on WES data. VarSimLab is the only current program specifically designed to this task (https://varsimlab.readthedocs.io/en/latest/index.html, (Zare, et al., 2017); previously called CNV-Sim: https://nabavilab.github.io/CNV-Sim/). However, VarSimLab does not produce “test” genomes and target regions for simulated short reads. Instead, it creates short and temporary “genomes” with target regions only, in which additional copies for the "test" genome are created in the case of insertion, and additional copies for the "control" genome are created in the case of deletion. Short read files are then created from them. Therefore, VarSimLab cannot simulate the realistic scenario of CNVs with different degrees of overlap with target regions. Additionally, pooled samples with a common control cannot be simulated using VarSimLab, whereas many CNV detection applications for WES data require pooled samples as input. Finally, VarSimLab only accepts one chromosome at a time, and the majority of simulated CNVs overlap with each other when target regions are close to each other.

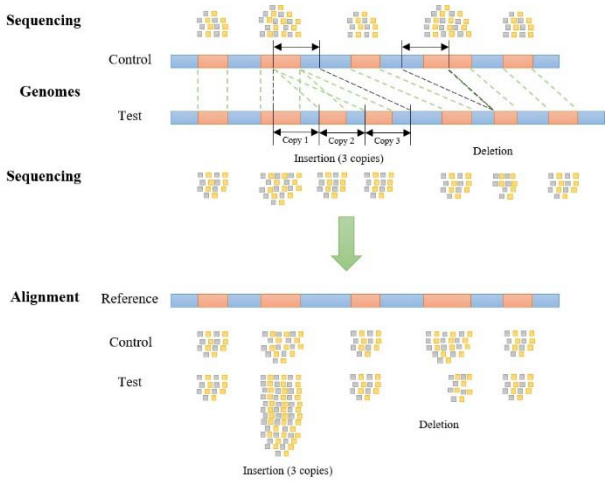


Fig. 1. CNVs detection throughout alignment of WES reads to a reference genome. WES data are obtained by sequencing target regions (red) in genomes of interest (blue). If the "test" genome contains insertions and/or deletions overlapping target regions, these regions will be rearranged (duplicated, deleted or shifted in their genomic coordinates) compared to control and reference genome. Reads (grey and yellow boxes) from the test genomes are aligned to original target regions in the reference genome. CNVs are detected according to the alignment.

Here, we describe SimulateCNVs, a novel tool that overcomes known limitations of existing CNV simulation tools developed to compare the performance of CNVs detection tools. SimulateCNVs is python-based and generates CNVs in both WES and WGS datasets and can be easily installed and used under Unix systems. SimulateCNV is particularly well suited to simulate CNVs in WES data. In a single command/step, this application output consists in rearranged genomes, short reads and bam files for multiple test genomes with the control being the original genome.

Specifically, SimulateCNVs outputs rearranged genomes with non-overlapping CNVs either designed by the user or randomly generated and target regions for sequencing. Additionally, it accepts genomes with multiple chromosomes. Users have the option to then employ SimulateCNVs to generate short read files by simulating Illumina single or paired-end sequencing for test and control samples. This step includes a custom codes written to utilize ART_illumina (Huang, et al., 2011), a WGS short read simulator, to sequence the target regions/genome. Finally, users can choose to simulate bam files from short read files for tests and control. This step is a routine pipeline utilizing BWA (Li, 2013), samtools (Li, 2011; Li, et al., 2009), picard (v2.15.0, https://broadinstitute.github.io/picard/) and GATK (McKenna, et al., 2010).

SimulateCNVs includes the following optional features:

(1) CNVs will not overlap with any gap regions (stretches of "Ns"). A script called "replaceNs.py" is also provided for replacing gap regions with nucleotides randomly.

(2) Set the minimum distance in bp between CNVs.

(3) Simulate CNVs overlapping with custom designed target regions and CNVs not overlapping with such target regions at all.

(4) For CNVs overlapping with target regions, choose the minimum number of base pairs of the target regions to overlap with CNVs. This ensures that CNVs overlap with a satisfactory number of nucleotides from target regions.

(5) Set a distance so that target regions closer than that distance can be connected in a single region for sequencing. Users can also choose to sequence up- and downstream of the target regions.

(6) There are several ways to choose CNV start point and length:

   a. Start point can be chosen randomly, according to either a uniform or normal distribution, or can be read in from a file providing start, end and copy number of each CNV.

   b. Length can be chosen randomly, according to either a uniform or normal distribution, read from a file providing start, end and copy number of each CNV, or read from a file solely containing length and number of CNV of each length.

   c. For both start point and length, if choosing to generate CNVs randomly, users can choose to generate a specified number of CNVs on each chromosome/scaffold, or total CNV number for the whole genome. In the latter case, the number of CNVs will be proportional to the length of chromosomes/scaffolds, with at least 1 CNV generated in the shortest chromosome/scaffold.

(7) Different numbers of CNVs can be specified per chromosome, for the whole genome or provided in files described in item 6 above. If CNV number is specified by chromosome, each chromosome will have the same number of CNVs. Otherwise, the number of CNVs on each chromosome will be proportional to the length of that chromosome.

(8) If a chromosome is too short or there are too many gap regions to accommodate the number of CNVs requested, the program will report in a log file how many CNVs are generated instead of the planned amount and continue to the next chromosome. This feature prevents program-killing errors like those thrown in such cases by other applications.

(9) The program is able to set the proportion of CNVs to be insertions/deletion.

(10) The program is able to simulate CNVs in either WES or WGS data.

SimulateCNVs can be used after downloading and unpacking it. The download link, requirements and manual are available at https://yjulyxing.github.io/. To run the program, the user is required to provide a genome file in fasta format. For WES simulation, the user also needs to provide a file of target regions. Users need to specify if gap regions should be avoided for CNV simulation, and the minimum length between any of the two CNVs. Users should then specify the method for CNV simulation, for example, number, distribution, range of length and copy numbers of CNVs, and proportion of insertions. For WES simulation, users also need to choose the minimum base pairs of target regions to be overlapping with CNVs and specify how many CNVs overlapping with exons and how many CNVs outside of exons should be simulated. Output directory, output name, how many test samples are simulated and if control should be given will also need to be specified. Rearranged genomes and target regions are output generated during this step. Then,

if short reads are required to be simulated, users need to choose the maximum distance for target regions to be connected for sequencing, in addition to distance up- and down-stream of target regions to be sequenced. Users should also decide on other parameters relevant to sequencing, for example, fold coverage, single or paired-end sequencing, etc. Finally, if users choose to simulate bam files with indexes, the local absolute path to picard and GATK must be provided; samtools and BWA are also required to be installed.

The computation time for various genomes is shown in Supplementary Table 1.

Acknowledgements

The high performance research computing system ada at Texas A&M University was used for testing the software application.

Funding

This work has been supported by the National Institute of Food and Agriculture, U.S. Department of Agriculture, under award number TEX0-1-9599, the Texas A&M AgriLife Research, and the Texas A&M Forest Service.

Conflict of Interest: none declared.

References

Bartenhagen, C. and Dugas, M. RSVSim: an R/Bioconductor package for the simulation of structural variations. Bioinformatics 2013;29(13):1679-1681.

Faust, G. SVsim: a tool that generates synthetic Structural Variant calls as benchmarks to test/evaluate SV calling pipelines. In.; 2017.

Feuk, L., Carson, A.R. and Scherer, S.W. Structural variation in the human genome. Nature reviews. Genetics 2006;7(2):85-97.

Goh, G. and Choi, M. Application of whole exome sequencing to identify disease-causing variants in inherited human diseases. Genomics & informatics 2012;10(4):214-219.

Huang, W., et al. ART: a next-generation sequencing read simulator. Bioinformatics 2011;28(4):593-594.

Li, H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. Bioinformatics 2011;27(21):2987-2993.

Li, H. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. arXiv preprint arXiv:1303.3997 2013.

Li, H., et al. The sequence alignment/map format and SAMtools. Bioinformatics 2009;25(16):2078-2079.

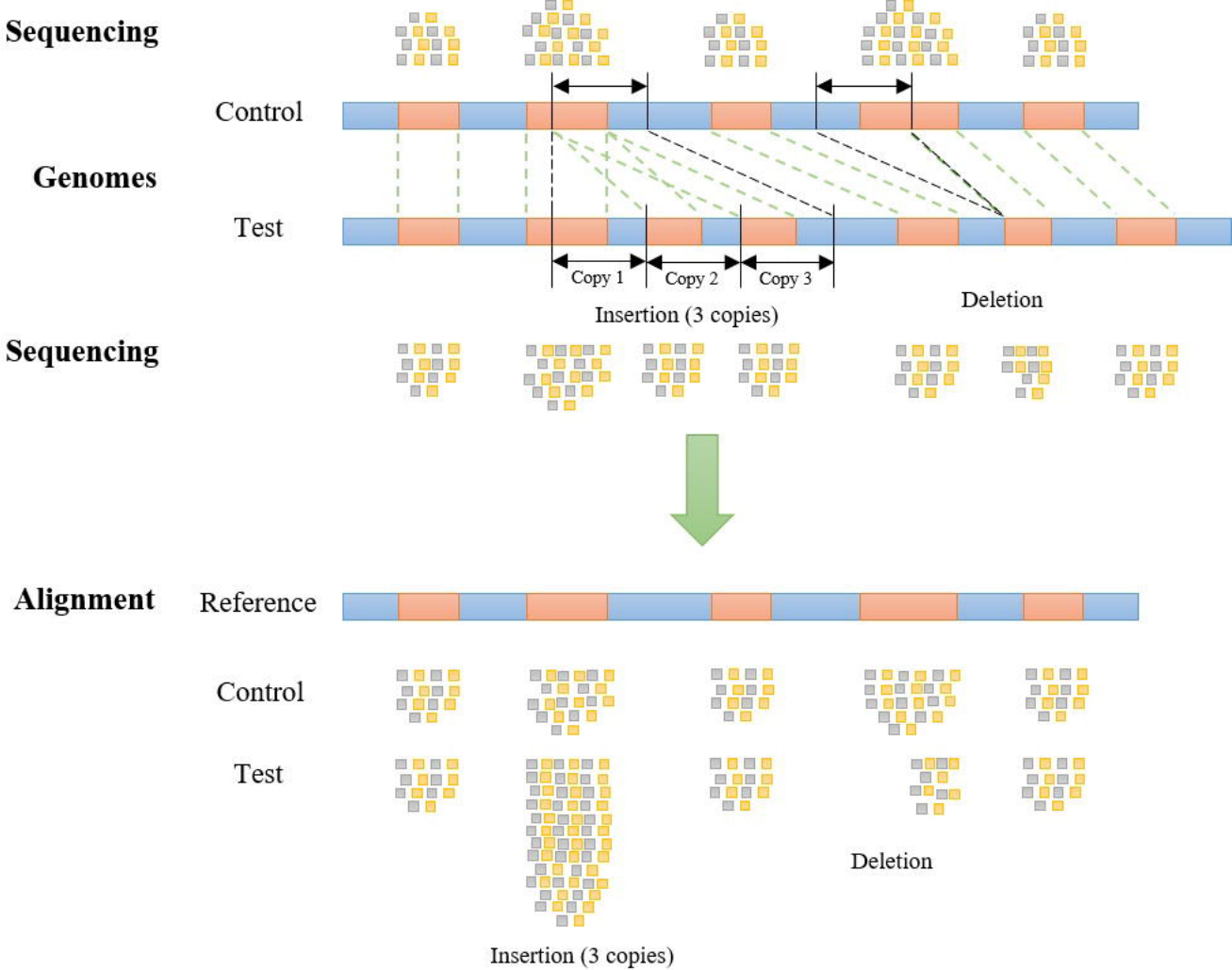
McKenna, A., et al. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. Genome research 2010.

Pattnaik, S., et al. SInC: an accurate and fast error-model based simulator for SNPs, Indels and CNVs coupled with a read generator for short-read sequence data. BMC bioinformatics 2014;15(1):40.

Qin, M., et al. SCNVSim: somatic copy number variation and structure variation simulator. BMC bioinformatics 2015;16(1):66.

Xia, Y., et al. Pysim-sv: a package for simulating structural variation data with GC-biases. BMC bioinformatics 2017;18(3):53.

Zare, F., et al. An evaluation of copy number variation detection tools for cancer using whole exome sequencing data. BMC bioinformatics 2017;18(1):286.


Sequencing
Control
Genomes
Test
Copy 1
Copy 2
Copy 3
Insertion (3 copies)
Deletion
Sequencing
Alignment
Reference
Control
Test
Deletion
Insertion (3 copies)