SCSsim User Guide

Zhenhua Yu

zhyu@nxu.edu.cn

1. Introduction

SCSsim is a tool designed for simulating single-cell genome sequencing (SCS) data. It consists of three modules: 1) “simuVars” module aims to generate single cell genome from a given reference sequence by inserting user-defined genomic variations into genomic loci; 2) “learnProfile” component is developed to infer sequencing platform dependent profiles from real data; 3) “genReads” utility is provided to mimic single-cell genome amplification and read generation procedures based on the results of “simuVars” and “learnProfile”.

2. Requirements

✓ Linux systems.

✓ CMake2.8+.

✓ g++.

3. Installation

To build binary, do as follows:

```
tar -zxvf SCSsim.tar.gz
cd SCSsim
cmake .
make
```

After the installation, the main programs of SCSsim are generated in “bin” directory. Type following command if you want to add SCSsim to system PATH:

```
make install
```

4. Usage

4.1. Use of “scssim simuvars” subcommand

Users can use “scssim simuvars” subcommand to simulate the genome sequence of single cell by defining various genomic variations. The types of variations include single nucleotide polymorphism (SNP), single nucleotide variation (SNV), short insert and deletion (indel), and copy number variation (CNV). The inputs of “simuvars” are listed as follows:

Parameter	Description	Possible values
-r, --ref	[required] FASTA reference file from which reads will be sampled	Ex: /path/to/hg19.fa
-v, --var	[optional] a file defining the variations (indel, SNV and CNV) of single cell to be simulated	Ex: /path/to/variation.txt Default: null
-s, --snp	[optional] a file defining the SNPs of single cell to be simulated	Ex: /path/to/hg19_snp138_chr1.txt Default: null
-o, --output	[required] output file (.fasta) to save generated sequences	Ex: /path/to/simu.fa

Example:

```
scssim simuvars -r <reference>.fasta –s /path/to/snp.txt -v /path/to/variation.txt –o /path/to/output.fasta
```

The SNP file contains all the SNPs to be simulated, and the data is formatted as follows:

```
rs58108140 chr1     10583   A/G +    G
rs189107123     chr1     10611   C/G +    C
rs71252251 chr1     14976   C/T -    G
```

Each row gives the detailed information of one SNP, and the columns represent the name of the SNP, chromosome name, 1-based position of the SNP, observed alleles, the strand of the SNP and reference allele, respectively. The SNP data can be manually defined by users or downloaded from UCSC Genome Browser https://genome.ucsc.edu/cgi-bin/hgTables (an example of the downloaded SNP data can be found in the subdirectory "testData/snps" of the software package). The downloaded data should be further processed to only include the required columns. We provide a shell script named "snpFilter.sh" included in subdirectory "util" of the SCSsim software package, and users can use this script to preprocess the downloaded SNP data to generate SNP file that is required by the "scssim simuvars" subcommand. For example, if the downloaded SNP file is "downloaded_snp.txt", then users can save filtered SNPs to file "output_snp.txt" by typing following command:

```
snpFilter.sh downloaded_snp.txt output_snp.txt
```

The variation file defines genomic variations to be simulated, and needs to be manually created by the users. All variations should be given in a single file. Here is an example of the variation file defining 2 CNVs, 2 SNVs, 2 inserts and 2 deletions (no header is required):

```
c   chr20   1    500000 1    1
c   chr20   10000000    15000000    4    3
s   chr20   2000000     c    T    homo
s   chr20   4000000     a    g    het
i   chr20   1300000     cgtccgtc    homo
i   chr20   2500000     tcgag   het
d   chr20   5000000     10  het
d   chr20   13500000    4   homo
```

Each row defines a variation, and different types of variations are distinguished by the first column: *c* denotes CNV, *s* represents SNV, *i* refers insert, and *d* denotes deletion. A CNV contains these items: chromosome name, 1-based start position, 1-based end position, total copy number, and major allele copy number. A SNV is depicted by chromosome name, 1-based position, reference allele, mutated allele, and type of the SNV ("homo" refers homozygous mutation and "het" denotes heterozygous mutation). A short insert is described by chromosome name, 1-based position, nucleotide sequence to insert, and type of the insert ("homo" or "het"). Finally, a short deletion is specified using items including chromosome name, 1-based start position, length and type of the deletion ("homo" or "het").

4.2. Use of "scssim learn" subcommand

The "scssim learn" subcommand is designed to infer sequencing profiles from real sequencing data. In the current version, four profiles including indel error distributions, base substitution probabilities, Phred quality distributions and GC-content bias are measured. Users can build their own profiles from a given real dataset using this program. The inputs of the program are described as follows:

Parameter	Description	Possible values
-b, --bam	[required] a non-tumor BAM file	Ex: /path/to/sample.bam
-t, --target	[optional] a BED file defining the target regions if whole-exome sequencing was used	Ex: /path/to/targets.bed Default: null
-v, --vcf	[required] a VCF file generated from the non-tumor BAM	Ex: /path/to/sample.vcf
-r, --ref	[required] FASTA reference file to which the reads were aligned	Ex: /path/to/hg19.fa

-w, --wsize	[optional] the length of windows used to infer GC-content bias	Ex: wsize=10000 Default: 1000
-k, --kmer	[required] the length of kmer sequence to infer base substitution probabilities	Ex: kmer=4 Default: 3
-s, --samtools	[optional] the path of samtools	Ex: /path/to/samtools Default: samtools
-o, --output	[optional] output file to save results	Ex: /path/to/profile.txt Default: null

The minimum requirements to build a sequencing profile include a non-tumor BAM file, a VCF file derived from the BAM file and a FASTA file representing the reference. Users should first generate the VCF file using GATK software by following command like:

```
java -jar GATK HaplotypeCaller -I <sample>.bam -O <sample>.vcf -R <reference>.fasta
```

Then infer profiles using SCSsim subcommand “scssim learn”:

```
scssim learn -b <sample>.bam -t <targets>.bed -v <sample>.vcf -r <reference>.fasta > -o <sample>.profile
```

for whole-exome sequencing data or

```
scssim learn -b <sample>.bam -v <sample>.vcf -r <reference>.fasta > -o <sample>.profile
```

for whole-genome sequencing data.

4.3. Use of “scssim genreads” subcommand

The “scssim genreads” subcommand is developed to simulate single-end or paired-end reads based on the results of “scssim simuvars” and “scssim learn”. Single-cell genome amplification and read generation procedures are implemented. The inputs of the program are listed as follows:

Parameter		Description	Possible values
	-i, --input	[required] FASTA sequence file generated by “scssim simuvars” subcommand	Ex: /path/to/simu.fa
MALBAC options	-p, --primers	[optional] the number of each type of primer	Ex: primers=10000 Default: 100000
	-r, --gamma	[optional] a parameter controlling the number of primers used in each cycle	Ex: gamma=5e-10 Default: 1e-9
Read simulation options	-m, --model	[required] a file defining sequencing profiles inferred by “scssim learn” subcommand	Ex: /path/to/profile.txt
	-l, --layout	[optional] read layout (SE for single end, PE for paired-end)	Ex: layout=SE Default: PE
	-c, --coverage	[optional] sequencing coverage	Ex: coverage=30 Default: 5
	-s, --isize	[optional] mean insert size for paired-end sequencing	Ex: isize=300 Default: 260
	-t, --threads	[optional] the number of threads to use	Ex: threads=8 Default: 1
	-o, --output	[required] the prefix of output file	Ex: /path/to/reads

Example:

```
scssim genreads -i /path/to/simu.fa -m /path/to/hiseq2500.profile -t 5 -o /path/to/reads
```

5. Contact

If you have any questions, please contact zhyu@nxu.edu.cn.