SNP Mutator Documentation

Release 1.2.0

Hugh Rand

May 15, 2019

Contents

Contents:

CHAPTER 1

SNP Mutator

Generate mutated sequence files from a reference genome.

SNP Mutator was developed by the United States Food and Drug Administration, Center for Food Safety and Applied Nutrition.

- Free software
- Documentation: https://snp-mutator.readthedocs.io/en/latest/readme.html
- Source Code: https://github.com/CFSAN-Biostatistics/snp-mutator
- PyPI Distribution: https://pypi.python.org/pypi/snp-mutator

1.1 Features

- Reads a fasta file and generates any number of mutated fasta replicate files.
- Mutations can be any number of single-base substitutions, insertions, and deletions at randomly chosen positions, uniformly distributed across the genome.
- Mutations can be chosen from a subset (pool) of all possible positions.
- Replicates can be partitioned into multiple groups with each group sharing a pool of eligible positions.
- Generates a summary file listing the original base and the mutation for all mutated positions.
- Mutations can be either monomorphic or polymorphic.
- VCF file generation.
- Various metrics can be saved to an output file.
- The fasta defline sequence id can be customized.

1.2 Citing SNP Mutator

To cite SNP Mutator, please cite the publication below:

> Davis S, Pettengill JB, Luo Y, Payne J, Shpuntoff A, Rand H, Strain E. (2015) CFSAN SNP Pipeline: an automated method for constructing SNP matrices from next-generation sequence data. PeerJ Computer Science 1:e20 https://doi.org/10.7717/peerj-cs.20

1.3 License

See the LICENSE file included in the SNP Mutator distribution.

CHAPTER 2

Installation

At the command line:

```
$ pip install --user snp-mutator
```

Update your .bashrc file with the path to user-installed python packages:

```
export PATH=~/.local/bin:$PATH
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv snp-mutator
$ pip install snp-mutator
```

2.1 Upgrading SNP Mutator

If you previously installed with pip, you can upgrade to the newest version from the command line:

```
$ pip install --user --upgrade snp-mutator
```

2.2 Uninstalling SNP Mutator

If you installed with pip, you can uninstall from the command line:

```
$ pip uninstall snp-mutator
```

2.3 Requirements

SNP Mutator requires requires python version 2.7, 3.4, 3.5, 3.6 or 3.7. It has not been tested on other python versions.

CHAPTER 3

Usage

SNP Mutator is a single script called `snpmutator`. After installation, it will be on your path, ready to use.

3.1 Quick Start

Step 1 - Create a work directory:

```
$ mkdir workdir
$ cd workdir
```

Step 2 - Gather a reference fasta file:

```
# For this example we will use a fasta file from our sister project, snp-pipeline
$ wget https://raw.githubusercontent.com/CFSAN-Biostatistics/snp-pipeline/master/
↪snppipeline/data/agonaInputs/reference/NC_011149.fasta
```

Step 3 - Generate the mutated sequences:

```
# The mutated sequence files are generated in the current working directory
# -r 1, set random seed
# -n 1000, generate 1000 mutated sequences
# -s 900, nine hundred substitutions in each mutated sequence
# -i 50, fifty insertion in each mutated sequence
# -d 50, fifty deletions in each mutated sequence
# -o summary.tsv, generate a mutation summary file called summary.tsv
# -v variants.vcf, generate a VCF file of mutations
# -p 100000, choose mutations from a pool of 100000 positions
# -g 100, partition the 1000 replicates into 10 groups of 100 replicates, with each␣
↪group having a separate pool of positions
# -m, create monomorphic alleles within each pool
# -M metrics, generate a metrics output file
# -R seq.fasta, generate the contatenated reference fasta file
```

(continues on next page)

(continued from previous page)

```
# -F fasta, directory where the fasta replicates will be generated
$ snpmutator -r 1 -n 1000 -s 900 -i 50 -d 50 -o summary.tsv -v variants.vcf -p 100000␣
↪-g 100 -m -M metrics -R seq.fasta -F fasta NC_011149.fasta
```

Step 4 - Examine the results:

```
$ ls NC_011149_mutated_*.fasta
$ less summary.tsv
$ cat metrics
```

3.2 Input Files

The only input file is the reference fasta file.

Note: Multi-fasta files may produce unexpected results. All the sequences are concatenated into a single sequence. All description lines after the first are discarded.

3.3 Output Files

3.3.1 Summary

An optional summary file in tab-delimited format lists the positions of the mutations for each of the replicates with the original base and the resulting mutation at each position.

3.3.2 VCF

An optional VCF file lists the mutations in Variant Call Format.

3.3.3 Replicate Files

Multiple mutated replicate files are generated in a directory of your choice. Files are named with the basename of the original reference file, suffixed with `_mutated_#.fasta`.

For example, if the reference file name is `NC_011149.fasta`, the first two replicate files are named `NC_011149_mutated_1.fasta` and `NC_011149_mutated_2.fasta`.

The defline (description) of the generated fasta files is copied from the original reference fasta file, but with a suffix describing the mutations. For example, the defline suffix `(mutated s=2 i=1 d=0)` indicates there are two substitutions, one insertion, and zero deletions.

3.3.4 Reference File

An optional concatenated reference file can be generated. This is the original fasta file with all the sequences concatenated into a single sequence. All the replicates will be mutations of this file.

3.3.5 Metrics

With the `--metrics` option, a file of metrics is created describing the mutated positions.

3.4 Pooling

The `--pool` option increases the likelihood of multiple replicates having mutations at the same positions by limiting the number of positions along the genome where mutations will be introduced. The positions in the pool are choosen randomly and uniformly from the positions in the genome.

3.5 Grouping

The `--group` option partitions the replicates into groups with each group having a different pool of eligible positions. This has the effect of creating more closely related replicates within groups and more distant replicates between groups.

3.6 Monomorphic Alleles

The `--mono` option ensures that when multiple replicates have a mutation at the same position, the mutation will be identical in each replicate. However, when used with the `--group` option, the monomorphic mutations are only within the group. Different groups of replicates may have polymorphic alleles with respect to other groups of replicates.

3.7 Command Reference

```
usage: snpmutator [-h] [-o FILE] [-n INT] [-s INT] [-i INT] [-d INT] [-r INT]
                  [-p INT] [-g INT] [-m] [-I SEQID] [-v FILE] [-M FILE]
                  [--version]
                  input_fasta_file

Generate mutated sequence files from a reference genome. Takes a fasta file
and creates a specified number of randomly generated base substitutions,
insertions, and deletions. Outputs the mutated genomes, and optionally, a
summary file listing the mutations by position.

positional arguments:
  input_fasta_file      Input fasta file.

optional arguments:
  -h, --help            show this help message and exit
  -n INT, --num-simulations INT
                        Number of mutated sequences to generate. (default:
                        100)
  -s INT, --num-substitutions INT
                        Number of substitutions. (default: 500)
  -i INT, --num-insertions INT
                        Number of insertions. (default: 20)
  -d INT, --num-deletions INT
                        Number of deletions. (default: 20)
  -r INT, --random-seed INT
                        Random number seed; if not set, the results are not
                        reproducible. (default: None)
  -p INT, --pool INT    Choose variants from a pool of eligible positions of
                        the specified size (default: 0)
  -g INT, --group INT   Group size. When greater than zero, this parameter
```

(continues on next page)

(continued from previous page)

```
                      chooses a new pool of positions for each group of
                      replicates. (default: None)
  -m, --mono          Create monomorphic alleles (default: False)
  -I SEQID, --seqid SEQID
                      Output fasta description line sequence ID. Each
                      mutated output file has only one sequence. If not
                      specified, the defline id will be the id of the first
                      sequence in the input fasta file. The defline is
                      always suffixed with an annotation in this format:
                      (mutated s=900 i=50 d=50). The seq id is also written
                      to the CHROM column of the output VCF file. (default:
                      None)
  -R FILE, --ref FILE Output concatenanted reference file with no mutations,
                      but all sequences concatenanted together. All the
                      replicates will be mutations of this file. (default:
                      None)
  -o FILE, --summary FILE
                      Output positional summary file. (default: None)
  -v FILE, --vcf FILE Output VCF file. (default: None)
  -M FILE, --metrics FILE
                      Output metrics file. (default: None)
  --version           show program's version number and exit
```

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/CFSAN-Biostatistics/snp-mutator/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

4.1.4 Write Documentation

SNP Mutator could always use more documentation, whether as part of the official SNP Mutator docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/CFSAN-Biostatistics/snp-mutator/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *snp-mutator* for local development.

1. Fork the *snp-mutator* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/snp-mutator.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv snp-mutator
$ cd snp-mutator/
$ pip install Numpy
$ python setup.py develop
$ pip install sphinx_rtd_theme    # the documentation uses the ReadTheDocs theme
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 snp-mutator tests
$ python setup.py test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Update the documentation and review the changes locally with sphinx:

```
$ cd docs
$ make html
$ xdg-open _build/html/index.html
```

7. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

8. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, 3.4, 3.5, 3.6 and 3.7. Make sure that the tests pass for all supported Python versions.

CHAPTER 5

Credits

5.1 CFSAN Bioinformatics Team

- James Pettengill
- Hugh Rand
- Steve Davis
- Nathan Xue

5.2 External Contributors

None yet. Why not be the first?

CHAPTER 6

History

CHAPTER 7

1.2.0 (2019-05-15)

- Added support for Python 3.6 and 3.7.
- Added a command line option to specify the output directory for the generated fasta files.

CHAPTER 8

1.1.0 (2018-10-23)

- Added the capability to specify the fasta defline sequence id in the output files.
- The unmutated, but concatenated sequence reference can be written to a separate output file. All the replicates will be mutations of this file.

CHAPTER 9

1.0.0 (2018-10-12)

- Mutations can be chosen from a subset (pool) of all possible positions.
- Replicates can be partitioned into multiple groups with each group sharing a pool of eligible positions.
- Mutations can be either monomorphic or polymorphic.
- Add the capability to generate VCF output files.
- Add the capability to generate a metrics output file.
- The installer creates an executable script called `snpmutator`.
- Insertions are now placed after the original reference position, not before, as in prior versions of snpmutator. This means it will not be possible to exactly reproduce the results of prior versions of this software.

CHAPTER 10

0.2.0 (2016-01-15)

- Allow lowercase bases in the input fasta file.
- Do not mutate gaps or ambiguous positions.
- Add a command line switch to show the program version.

CHAPTER 11

0.1.1 (2015-06-17)

- Remove spaces from the summary file column headings. This will simplify downstream analysis in some scripting languages.

CHAPTER 12

0.1.0 (2015-05-21)

- First release on GitHub, Read the Docs, and PyPI.

CHAPTER 13

Indices and tables

- genindex
- search