Simulome

October 21, 2016

Version: 1.2

Title: Simulome: Prokaryote genome and variant simulator.

Author: Adam Price

Maintainer: Adam Price <price0416@gmail.com>

Description: Simulome provides a powerful and easy to use tool for creating pseudo-genomes and mutated variants for prokaryotes. Simulome makes it possible to create genomes based on any bacterial species, while controlling for a variety of factors. Furthermore, it provides a range of options that can be used in combination to create mutated variants of the simulated genome, which allows for controlled testing of specific genomic conditions. Simulome can be used in combination with reads generated from next-generation sequencing platforms or alternatively with NGS read simulation packages.

URL: https://github.com/price0416/Simulome

Dependencies

Simulome was developed in a linux/unix environment and requires the following libraries for proper functionality.

- Python 2.7.2
- Biopython 1.6.1+
- BLAST 2.3.0+

Description

Simulome takes an existing genome and the corresponding annotation information for that genome and samples a subset of the genes to use as a simulated genome. Sampling is performed based gene length and genes are selected to approximate a normal distribution of read lengths. That is, the mean length of all genes in the provided reference genome and the standard deviation are calculated, and genes are then sampled such that the mean and standard deviation of the simulated reference genome approximates that of the originally provided genome. An initial simulation is created by using these sampled genes in conjunction with non-duplicating intergenic regions, or by randomly sampling from the intergenic regions of the provided reference genomes. Once the initial genome is simulated, a variant genome can be simulated to meet desired specifications. Alternatively, users can specify not to simulate a pseudo-genome and can directly apply Simulome's variant tools to create a mutated genome based directly on the provided reference genome. Four run modes are available and can be used in any combination to produce variants containing SNPs, Synonymous/nonsynonymous mutations, indels, and/or duplicate regions. Additional optional arguments are available to allow direct control over selection criteria and genomic structure. The resulting simulations will each be provided as a FASTA nucleotide file, a GTF/GFF3 annotation file, and a variant metadata file.

Usage

python simulome.py --genome <genome.fasta> --anno <genome.gff> --output <destination> <RUN MODE> <OPTIONAL ARGUMENTS>

Required Arguments

--genome	File representing genome. FASTA nucleotide format.
--anno	File containing genome annotation information in GTF/GFF3 format. This file should correspond to the FASTA file representing the selected genome.
--output	Output destination. This option will create a folder named with the supplied argument containing output files. Providing a –o option of ‘ecoli’ will create the directory, ./ecoli/ and populate it with files such as: ./ecoli/ecoli_simulated.fasta

SNP Run Mode Arguments

--snp	Boolean. Set this option to TRUE to enable SNP mutations in the variant genome.
--num_snp	The number of SNPs to simulate per gene. This argument is required for SNP run mode.
--snp_window	Window size in which to simulate SNPs. This option allows control over the density of SNP mutations. If a window size is specified, the number of SNPs specified by the –s option will occur within a randomly determined range of this specified window size. I.E. <-s 5 –w 10> will create 5 SNPs within a 10 base pair window. If this option is not specified, SNPs will be distributed randomly over the length of each gene.
--snp_distrib	Boolean. Create different numbers of SNPs in each gene based on a Gaussian distribution. If this option is true, --num_snp will be used as the mean of the distribution.
--snp_std_dev	This option is required if –-snp_distrib=true. Standard deviation for the distribution of SNP counts for each gene. A larger standard deviation will result in a wider range of SNP counts per gene, and a smaller deviation will result in a more condensed range.

Synonymous/Non-synonymous Run Mode

--syn	Boolean. Set this option to TRUE to enable Synonymous/Non-synonymous run mode. This run mode allows you to specify a percentage of synonymous mutations to occur in each gene. It assumes the start position of the gene to be the open reading frame. Requires “mutation_log.dat” file as provided, in $PATH or local directory.
--syn_percent	The percentage of mutations per gene that will be synonymous.
--syn_mean	The mean number of total mutations desired per gene.
--syn_std_dev	Standard deviation for the distribution of mutations counts per gene. A larger standard deviation will result in a wider range of total number of mutations, and a smaller deviation will result in a more condensed range.

Insertion/Deletion Run Mode

--indel	This option specifies insertion/deletion for mutations in the variant genome. Possible values are: 1 = Insertions only. 2 = Deletions only. 3 = Both insertions and deletions.
--ins_len	Length of insertion events. Required for insertion mode.
--num_ins	Number of inserts to simulate in each gene. Default = 1.
--is_copy_event	Boolean. If this option is true, insertions sequences will be randomly copied from existing regions of the genome.
--ins_distrib	Boolean. Create different length insertion sequences in each gene based on a Gaussian distribution. If this option is true, --num_ins will be used as the mean of the distribution.
--ins_std_dev	This option is required if –-ins_distrib=true. Standard deviation for the distribution of insertion lengths. A larger standard deviation will result in a wider range of insertion lengths, and a smaller deviation will result in a more

condensed range.

--del_len	Length of deletion events. Required for deletion mode. Deletions cannot be longer than the target genes, in which event, genes shorter than desired deletion length will be omitted from mutation and warnings will be displayed.
--num_del	Number of deletes to simulate in each gene. Default = 1.
--del_distrib	Boolean. Create different length deletion events in each gene based on a Gaussian distribution. If this option is true, --num_del will be used as the mean of the distribution.
--del_std_dev	This option is required if –-del_distrib=true. Standard deviation for the distribution of deletion event lengths. A larger standard deviation will result in a wider range of deletion lengths, and a smaller deviation will result in a more condensed range.

Duplication Run Mode

--duplicate	Boolean. Set this option to TRUE to create duplications in the variant genome. Allows control for reads that map to multiple locations. Uses the initial genome simulation and appends duplicate regions until the desired level of duplication is reached.
--percent_dup	Percent of duplicate regions to include in the genome. Required for duplication mode.

Optional Arguments

--whole_genome	Boolean. If this is true, the provided genome will be used instead of a simulated pseudo-genome and variants will be performed directly on the provided reference. Cannot be used with –num_genes.
--num_genes	Number of genes to simulate. Default = 100.
--sort_log	How to sort the variant log file. Acceptable options are 'genome' and 'mutation'. 'Genome' will sort the output log by the order mutations occur in the genome, while

	'mutation' will sort the output log in the order mutations were created. (Default=Genome)
--intergenic_len	Length of intergenic regions. For random length intergenic regions, specify 0 for this option. Random intergenic length range is 0-2000. Default = 500.
--random_intergenic	Boolean. If this is true, intergenic regions will be randomly synthesized between genes. If false, intergenic regions from the provided genome will be randomly sampled. (Default=False)
--operon_level	Simulate operons. Input should be approximate percentage of desired operon content. Default = 0.
--seed	Specifies a seed for the random number generator. By default a random seed will be selected for each run. By specifying a seed, the same gene selection and mutations can be repeated identically across multiple runs.
--type	Feature type to simulate from annotation file. I.E: gene, exon, CDS. Case sensitive. Note that this must match the desired feature type in the annotation file provided. Default = gene.
--strict_dup	Boolean. Allow duplicate sequence regions to exist in the initial genome simulation. Selecting FALSE for this option will BLAST each gene and simulated intergenic region against the growing simulation and prevent duplicate regions from being included in the genome. Depending on the level of natural duplication in the genome provided, this may result in fewer genes existing in the genome than specified. Can be memory intensive in some cases. Default = False.
-v, --verbose	Verbose level. Default = 1. [0 = Quiet, 1 = Verbose, 2 = Very Verbose]

Examples

- Simulate a genome based on e.coli containing 100 genes, output files to a folder called ecoli_simulation/.

```
python simulome.py --genome=ecoli_genome.fasta --anno=ecoli_anno.gtf --output=ecoli_simulation --num_genes=100
```

- Simulate a genome based on e.coli containing 500 genes, and a variant of the simulated genome in which each gene contains 10 SNPs, output to a folder called ecoli_simulation/.

```
python simulome.py --genome=ecoli_genome.fasta --anno=ecoli_anno.gtf --output=ecoli_simulation --num_genes=500 --snp=TRUE --num_snp=10
```

- Simulate a genome based on e.coli containing 500 genes, and a variant of the simulated genome in which each gene contains a variable number of SNPs based on a Gaussian distribution with a mean of 10 and a standard deviation of 3, output to a folder called ecoli_simulation/.

```
python simulome.py –genome=ecoli_genome.fasta –anno=ecoli_anno.gtf –output=ecoli_simulation --num_genes=500 --snp=TRUE --num_snp=10 --snp_distrib=true --snp_std_dev=3
```

- Simulate a genome based on e.coli containing 500 genes, and a variant of the simulated genome in which each gene contains a number of Synonymous/nonsynonymous mutations based on Gaussian distribution with a mean of 10 and a standard deviation of 3. In each case, approximate 70% of mutations to be synonymous. Output to a folder called ecoli_simulation/.

```
python simulome.py –genome=ecoli_genome.fasta –anno=ecoli_anno.gtf –output=ecoli_simulation --num_genes=500 --syn=TRUE --syn_percent=70 --syn_mean=10 --syn_std_dev=3
```

- Simulate a genome based on e.coli containing 500 genes, and a variant of the simulated genome in which each gene contains 10 SNPs that are concentrated in 50 base pair windows, output to a folder called ecoli_simulation/.

```
python simulome.py –genome=ecoli_genome.fasta –anno=ecoli_anno.gtf –output=ecoli_simulation –num_genes=500 --snp=TRUE –-num_snp=10  --snp_window=50
```

- Simulate a genome based on e.coli containing 100 genes, and a variant of the simulated genome in which each gene contains an insertion event of length 100, output files to a folder called ecoli_simulation/.

```
python simulome.py --genome=ecoli_genome.fasta --anno=ecoli_anno.gtf –output=ecoli_simulation --num_genes=100 --indel=1 --ins_len=100
```

- Simulate a genome based on e.coli containing 100 genes, and a variant of the simulated genome in which each gene contains an insertion event of length 100, and two deletion events of length 25, output files to a folder called ecoli_simulation/.

```
python simulome.py --genome=ecoli_genome.fasta --anno=ecoli_anno.gtf --output=ecoli_simulation --num_genes=100 --indel 3 --ins_len=100 --del_len 25 --num_del=2
```

- Simulate a genome based on e.coli containing 100 genes, and a variant in which 10% of the genome is duplicated, output files to a folder called ecoli_simulation/.

```
python simulome.py --genome=ecoli_genome.fasta --anno=ecoli_anno.gtf --output=ecoli_simulation --num_genes=100 --duplicate=TRUE  --percent_dup=10
```

- Simulate a genome based on e.coli containing 100 genes, with a variant genome in which each gene contains 5 SNPs, an insertion of length 500, a deletion of length 100, 10% genome duplication, and random intergenic region lengths. Output files to a folder called ecoli_simulation/.

```
python simulome.py --genome=ecoli_genome.fasta --anno=ecoli_anno.gtf --output=ecoli_simulation --num_genes=100 --snp=TRUE --num_snp=5 --indel 3 --ins_len=500 --del_len=100 --duplicate=TRUE --percent_dup=10
```

- Using the whole reference genome, simulate a variant genome in which each gene contains insertions with lengths based on a distribution with a mean of 100 and a standard deviation of 20, and a number of codon mutations with a total mean number of mutations of 15 and a standard deviation of 7, of which approximately 60 percent will be synonymous, output files to a folder called ecoli_simulation/.

```
python simulome.py --genome=ecoli_genome.fasta --anno=ecoli_anno.gtf --output=ecoli_simulation --indel=1 --ins_len=100 --ins_distrib=TRUE --ins_std_dev=20 --syn=TRUE --syn_percent=60 --syn_mean=15 --syn_std_dev=7
```

- Using the whole reference genome, simulate a variant genome in which each gene contains deletions with lengths based on a distribution with a mean of 50 and a standard deviation of 25, and a number of codon mutations with a total mean number of mutations of 10 and a standard deviation of 3, of which approximately 30 percent will be synonymous, additionally creating 10% genome duplication. Use full verbose mode. Output files to a folder called ecoli_simulation/.

```
python simulome.py --genome=ecoli_genome.fasta --anno=ecoli_anno.gtf --output=ecoli_simulation --indel=2 --del_len=100 --del_distrib=TRUE --del_std_dev=50 --syn=TRUE --syn_percent=30 --syn_mean=10 --syn_std_dev=3 --duplicate=TRUE --percent_dup=10 --verbose=2
```