Simulome

October 21, 2016

Version: 1.0.0

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: <PUT URL HERE>

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 read length and genes are selected to approximate a normal distribution of read lengths. An initial simulation is created by using these sampled genes in conjunction with non-duplicating intergenic regions, whose properties can be specified by the user. Once the initial genome is simulated, a variant genome can be simulated to meet desired specifications. Three run modes are available and can be used in any combination to produce variants of the simulated genome containing SNPs, 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 and a GTF/GFF3 annotation file.

Usage

```
python simulome.py -f <genome.fasta> -a <genome.gff> -o <destination> <RUN MODE>
<OPTIONAL ARGUMENTS>
```

Required Arguments

-f, --genome File representing genome. FASTA nucleotide format.

-a, --annotation File containing genome annotation information in GTF/GFF3 format. This file should correspond to the FASTA file representing the selected genome.

-o, --output_prefix 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

-1, --snp
Boolean. Set this option to TRUE to enable SNP mutations in the variant genome.

-s, --num_snp
The number of SNPs to simulate per gene. This argument is required for SNP run mode.

-w, --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.

Insertion/Deletion Run Mode

-2, --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.

-n, --insert_length
Length of insertion events. Required for insertion mode.

-l, --num_insert
Number of inserts to simulate in each gene. Default = 1.

-m, --delete_length
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.

-d, --num_delete
Number of deletes to simulate in each gene. Default = 1.

Duplication Run Mode

-3, --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.

-c, --percent
Percent of duplicate regions to include in the genome. Required for duplication mode.

Optional Arguments

-g, --num_genes
Number of genes to simulate. Default = 100.

-l, --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.

-p, --operon_level
Simulate operons. Input should be approximate percentage of desired operon content. Default = 0.

-x, --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.

-t, --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.

-y, --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. Default = TRUE.

-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 -f ecoli_genome.fasta -a ecoli_anno.gff -o ecoli_simulation -g 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 -f ecoli_genome.fasta -a ecoli_anno.gff -o ecoli_simulation -g 500 --snp TRUE –s 10
```

- 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 -f ecoli_genome.fasta -a ecoli_anno.gff -o ecoli_simulation -g 500 --snp TRUE –s 10
-w 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 -f ecoli_genome.fasta -a ecoli_anno.gff -o ecoli_simulation -g 100 --indel 1 –n 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 -f ecoli_genome.fasta -a ecoli_anno.gff -o ecoli_simulation -g 100 --indel 3 –n 100
–m 25 –d 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 -f ecoli_genome.fasta -a ecoli_anno.gff -o ecoli_simulation -g 100 --duplicate TRUE
–c 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 -f ecoli_genome.fasta -a ecoli_anno.gff -o ecoli_simulation -g 100 --snp TRUE –s 5
--indel 3 –n 500 –m 100 --duplicate TRUE –c 10
```