PropagAtE

Prophage Activity Estimator

01/21/2021
Kristopher Kieft
Anantharaman Lab
University of Wisconsin-Madison
kieft@wisc.edu

Current Version

PropagAtE v1.0.0

Citation

If you find PropagAtE useful please consider citing our preprint in bioRxiv:
Kieft, K., and Anantharaman, K. (2021). Identifying active prophages from metagenomes. bioRxiv.

Table of Contents:

Updates for v1.0.0:

There are currently no updates for PropagAtE. Version 1.0.0 is the initial software release.

Program Description

PropagAtE (Prophage Activity Estimator) uses genomic coordinates of integrated prophage sequences and short sequencing reads to estimate if a given prophage was in the lysogenic (dormant) or lytic (active) stage of infection. Prophages are designated according to a genomic/scaffold coordinate file, either manually generated by the user or taken directly from a VIBRANT (at least v1.2.1) output. The prophage:host read coverage ratio and corresponding effect size are used to estimate if the prophage was actively replicating its genome (significantly more prophage genome copies than host copies). PropagAtE is customizable to take in complete genomes or metagenomic scaffolds along with raw Illumina (short) reads, or instead take pre-aligned data files (sam or bam format). Threshold values are customizable but PropagAtE outputs clear “active” versus “dormant” estimations of given prophages with associated statistics.

Utility

- Allow for exploration of spatial and temporal dynamics of prophage populations
- Benchmarking control tests displayed high recall and accuracy
- Functions regardless of sequencing depth (number of reads)
- Functions whether there is a single prophage or multiple prophages on a host genome/scaffold

Cautions

- Samples that have been size fractionated (e.g., 0.2-micron filtered) may impact results
- Functions explicitly for integrated prophages, not for free or episomal phages
- Not all metagenome-derived integrated prophages may assemble with a host scaffold
- Utilized only for identifying active prophages, not for quantifying the fraction of prophages that are active

Requirements

System Requirements: PropagAtE has been tested and successfully run on Mac, Linux and Ubuntu systems.
Program Dependencies: Python3, Bowtie2, Samtools (see section below)
Python Dependencies: BioPython, SciPy (see section below)

Program Dependencies: Installation

Please ensure the following programs are installed and in your machine's PATH. Note: most downloads will automatically place these programs in your PATH.

Programs:

1. Python3: https://www.python.org (version >= 3.5)
2. Bowtie2: http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml
3. Samtools: http://www.htslib.org/ (version >= 1.11)

Example Installations

1. Python3: see Python webpage.
2. Bowtie2: `conda install -c bioconda bowtie2`, GitHub or follow instructions in the Bowtie2 manual.
3. Samtools: GitHub or follow instructions on the Samtools webpage

Python3 Dependencies: Installation

There are two Python3 dependencies that may not be installed. The remaining dependencies should already be installed.

Packages

1. BioPython: https://biopython.org/wiki/Download
2. SciPy: https://www.scipy.org/install.html

Example Installations

1. BioPython: `pip install biopython` or `apt-get install python-biopython` or `conda install -c conda-forge biopython`
2. SciPy: see the SciPy webpage

Other

VIBRANT is not a dependency but is useful for identifying prophages and can be used to easily input prophage coordinates to PropagAtE. Documentation for VIBRANT can be found on GitHub here. VIBRANT and PropagAtE were developed by the same author.

Running PropagAtE

PropagAtE is built for efficiently running on metagenomes, individual isolates genomes or genome scaffold fragments. Each prophage per genome/scaffold is considered individually, so results will not vary whether the scaffold is run as part of a metagenome or by itself.

Installation/Download

Note: if at any time you are given a "permission denied" error you can run `chmod +x <file_name>` or `chmod -R 777 <folder_name>`. Simply replace `<file_name>` or `<folder_name>` with the file/folder that you would like to add permissions to.

1) Install dependencies. See *Requirements* section above.
2) Download PropagAtE using git clone or download zip file. *Note*: if you download the zip file you will have the parent folder `PropagAtE-master` instead of `PropagAtE`.
`git clone https://github.com/AnantharamanLab/PropagAtE`
3) You may want to add permissions to all files.
`chmod -R 777 PropagAtE`
4) Move parent folder (`PropagAtE`) to desired location. PropagAtE will function no matter where the parent folder is located or moved to. The functioning script is `PropagAtE_run.py` and can be located anywhere.

Testing PropagAtE

Test out a small dataset of mixed active and dormant prophages. See `example_output/PropagAtE_example_dataset` for how the results should look. These examples assume the command is being called from the `example_output` folder.

Note: PropagAtE does not write to standard out (command prompt screen) while running or when it finishes (i.e., not verbose). However, PropagAtE will write to standard out in the event that it encounters an error, such as incorrect use of optional arguments, incorrect input file format, missing dependencies or incorrect dependency versions.

Note: The ways to run PropagAtE (i.e., set up flags) are not limited to these test examples.

Note: The results from these examples should be identical.

5) The inputs are scaffold sequences, short reads, and a VIBRANT prophage coordinates file. The reads may be unzipped or in gzip format depending on preference. Here they are gzipped for easier upload/download on GitHub.
`python3 ../PropagAtE_run.py -f example_sequence.fasta -r sample_forward_reads.fastq.gz sample_reverse_reads.fastq.gz -v VIBRANT_integrated_prophage_coordinates_example.tsv -o PropagAtE_example_results_from_reads.tsv`

6) The inputs are a SAM format alignment file and a VIBRANT prophage coordinates file.
`python3 ../PropagAtE_run.py -sb example_sequence.sorted.bam -v VIBRANT_integrated_prophage_coordinates_example.tsv -o PropagAtE_example_results_from_sb.tsv`

Input Data

There are two main formats for sequence data input: (1) sequences (`-f`) with reads (`-r`) or (2) a SAM/BAM (`-s` / `-b` / `-sb`) alignment file (just one type). The flag used for data input depends on the format of the input. There are two main formats for prophage input: (1) VIBRANT results coordinate file or (2) manually generated coordinate file. The same flag (`-v`) is used for either prophage input. *Note:* Either format for sequence data input is compatible with either format for prophage input. Here are the possible combinations of input data flags: (`-f -r -v`), (`-s -v`), (`-b -v`) or (`-sb -v`).

- For the `-f` input the phrase "_fragment_" should not appear in the definition line (the line starting with ">" before each sequence) because this term is used during analysis and prophage parsing. For details why, see VIBRANT documentation.

- Sequences input by the `-f` flag should be host genomes/scaffolds containing prophage sequences, not strictly prophages themselves. PropagAtE requires at a minimum 1000bp of host and 1000bp of prophage to run analyses.

- For the `-f` input the definition lines cannot have spaces (due to Bowtie2 processing). In most cases this is not an issue because PropagAtE will automatically detect spaces and replace them with "~~" characters. However, in an instance for which spaces were replaced in a SAM/BAM input file and the resulting scaffold names do not match the input coordinates file (`-v`) because of spaces, then the `-z` flag can be used. Use the `-z` flag to indicate what symbol(s) were used to replace spaces.

Prophage Coordinate Files

There are two main formats for prophage coordinates input (`-v`): (1) VIBRANT results coordinate file or (2) manually generated coordinate file.

- *VIBRANT method*: an automatically generated results file can be used directly from a VIBRANT analysis. VIBRANT v1.2.1 or greater must be used to have this file available. The file will named `VIBRANT_integrated_prophage_coordinates` and can be found in the `VIBRANT_results` output folder. No modification needs to be done for this file to be used an input for PropagAtE. The columns used are `scaffold` , `fragment` , `nucleotide start` and `nucleotide stop` .

- *manual method*: the other option, if prophages were identified by a different method, is to manually generate a prophage coordinates file compatible with PropagAtE. This method is also simple and requires only four columns of data. The columns must be in tab-separated format and have the following headers: `scaffold` , `fragment` , `start` and `stop` .

    i. `scaffold` is the name of the entire host sequencing that contains the prophage(s). Example: `scaffold_999`
    ii. `fragment` is the name of the prophage fragment. *IMPORTANT*: to be consistent with VIBRANT terminology the prophage fragment must contain the term `_fragment_#` at the end of the name, where `#` is an ID number (any number) in case of multiple prophage fragments per scaffold. Example: `scaffold_999_fragment_1`
    iii. `start` is the nucleotide number where the prophage starts
    iv. `stop` is the nucleotide number where the prophage stops

Input File Extensions

- `-f` : *.fasta*, *.fna*, *.fa* or *.fsa*
- `-s` : *.sam*
- `-b` or `-sb` : *.bam*
- `-r` : *.fastq* or *.fastq.gz*

Arguments and Flags

PropagAtE comes with a couple of very simple optional arguments. At any point you can see the help menu with `PropagAtE_run.py -h` .

Common optional arguments

- `-o` : name of output tab-separated file. The suggested extension is `.tsv` or `.txt`. The default output will be named according to a randomly generated unique run ID.

- `-t` : number of threads used for Bowtie2 read mapping. Read mapping is the only multi-thread process. Increasing threads used will increase speed. This flag is only compatible with `-r/-f` data input. Default = `1`.

Uncommon optional arguments

- `-g` : number of gap extensions allowed in read alignment (per read). The default is `1` and is optimized for more accurate read alignment. Increasing this value will decrease sensitivity but likely increase the number of reads aligned. This setting is compatible with any data input format. The allowed values are between 0 and 3. Use the `-all` flag to skip both `-g` and `-m`.

- `-m` : number of mismatches allowed in read alignment (per read). The default is `3` and is optimized for more accurate read alignment while allowing for prophage genome replication error. Increasing this value will decrease sensitivity but will increase the number of reads aligned. This setting is compatible with any data input format. The allowed values are between 0 and 10. Use the `-all` flag to skip both `-g` and `-m`.

- `-clean` : SAM/BAM files can be very large, and Bowtie2 index files are typically temporary. Use this setting to remove any generated SAM, unsorted BAM and/or Bowtie2 index files. All input data files (regardless of format) and sorted BAM files will always be retained. Default = `off`.

- `-z` : character(s) used to replace spaces in genome/scaffold names. Use this setting if spaces were replaced in the SAM/BAM alignment files but not in the prophage coordinates file. Use "~~" for PropagAtE-generated alignments.)

- `-all` : use this setting to skip gap/mismatch processing and keep all aligned reads. Default = `off`.

- `-e` : minimum effect size for significance by Cohen's *d* test. The default is `0.75` and the minimum is `0.6`. Values greater than `0.75` will represent a more significant difference in a prophage:host coverage ratio. Setting values below `0.75` may introduce false identifications (i.e., dormant prophages identified as active) whereas setting the value too high (e.g., `1.5`) may reduce identification of active prophages.

- `-c` : minimum prophage:host coverage ratio for significance. The default is `1.75` and the minimum is `1.5`. Setting values below `1.75` may introduce false identifications (i.e., dormant prophages identified as active) whereas setting the value too high (e.g., `3`) may reduce identification of active prophages.

- `-i` : a unique ID to append to any SAM files generated by Bowtie2. This is useful for running multiple input read datasets on a single input sequence file. It is suggested to use the read dataset ID.

Likely unused optional arguments

- `-p` : p-value cutoff for Mann-Whitney statistical test. There is no minimum or maximum value. This setting will likely not significantly effect results, but stetting values too low (e.g., `0.000001`) may impact active prophage identification. Default = `0.05`.

- `-n` : number of coverage sub-samples, each of size 100 nucleotides, to take for Mann-Whitney statistical analysis. The default and minimum values are sufficient and set to `5`. Increasing this value will not significantly impact results besides increasing the robustness of the Mann-Whitney statistical test.

- `-a` : remove outlier coverage values to "a" standard deviations from the average coverage value. This setting takes into account alignment errors, such as those resulting from sequence repeats. Coverage values that appear to be artificially high are removed. Default = `4`.

- `-x` : path to the Samtools executable if it is not in the system's PATH. Depending on the installation method the Samtools executable may or may not be in the system's PATH. The executable can be added to the PATH or this flag can be used.

The executable itself (e.g., `/home/User/anaconda3/bin/samtools` or the folder location can be given in this flag (e.g., `/home/User/anaconda3/bin/`).

Output Explanations

PropagAtE will always generate two files: the results tab-separated spreadsheet (`-o`) and a log file. The presence or absence of generated SAM, BAM and Bowtie2 index files will depend on the data inputs and user set flags.

- Log file: The log file (ending in `.log`) is named according to the results file (ending in `.tsv` , unless otherwise specified). At the top it will contain information about the input command and well as run info (date, time, version). The next section includes an overview of processes that were run, the time post-start and general info for some of the processes. For example, this will include the number of hosts and prophages detected, the number of reads removed post-processing, the number of valid reads used, and the estimated read length. This info will vary depending on the data inputs and user set flags. Next, there will be a list of all of the generated files (log, results, SAM, BAM, and sorted BAM). Finally, the number of prophages identified as active will be listed.

- Results file: The results spreadsheet contains the finalize active versus dormant results as well as all relevant metrics and statistics. The following are column names and explanations of the results file:

  i. prophage: the name of the prophage
  ii. host: the name of the host
  iii. active: "yes" indicates and active prophage in the lytic stage of infection. "no" indicates a dormant prophage in the lysogenic stage of infection.
  iv. MW_pvalue: the Mann-Whitney statistical test p-value
  v. CohenD: the Cohen's *d* effect size for the prophage:host coverage ratio
  vi. prophage-host_ratio: the prophage:host coverage ratio (prophage mean divided by host mean)
  vii. mean_difference: the difference between the prophage and host coverages (prophage mean minus host mean)
  viii. prophage_len: the length in nucleotides of the prophage region
  ix. prophage_mean_cov: the mean (average) coverage of the prophage region
  x. prophage_median_cov: the median coverage value of the prophage region
  xi. prophage_sd_cov: the standard deviation of the prophage region coverage values
  xii. host_len: the length in nucleotides of the host region
  xiii. host_mean_cov: the mean (average) coverage of the host region
  xiv. host_median_cov: the median coverage value of the host region
  xv. host_sd_cov: the standard deviation of the host region coverage values

Contact

Please contact Kristopher Kieft (kieft@wisc.edu or GitHub Issues) with any questions, concerns or comments.

Thank you for using PropagAtE!

```
                                                               ##
                                                             ##  ##
                                                           ##      ##
######   ##  ##     ##     #######   ######    #####       ##      ##
##  ##   ##  ##   ##  ##   ##        ##       ##             ##  ##
######   ######   ######   ##  ###   ######    ###             ##
##       ##  ##   ##  ##   ##   ##   ##           ##           ##
##       ##  ##   ##  ##   #######   ######   #####            ##
                                                            #  ##  #
                                                           # # ## # #
                                                         #            #
```

Copyright