PropagAtE

Prophage Activity Estimator

December 2021

Kristopher Kieft
kieft@wisc.edu
Anantharaman Lab
University of Wisconsin-Madison

Current Version

v1.1.0

Citation

If you find PropagAtE useful please consider citing our preprint in *bioRxiv*:
Kieft, K., and Anantharaman, K. (2021). Deciphering active prophages from metagenomes. BioRxiv 2021.01.29.428894.

Table of Contents:

Updates for v1.1.0:

- December 2021
- It is highly suggested to update from v1.0.0 to v1.1.0
- Significant improvements/modifications were made to the code
    - Streamline code to improve runtime
    - Remove Mann-Whitney test
    - Edit and add flag options
        - Add depth of coverage cutoff
        - Add minimum coverage cutoff
        - -o specifies an output folder
    - Python dependencies have been modified
    - Bowtie2 can take paired, interleaved or unpaired reads
    - Change gap/mismatch filtering to percent identity alignment
- The results may differ from v1.0.0
    - The same method of using coverage ratio and Cohen's *d* is retained

Updates for v1.0.0:

- Feb 2 2021: edited default `-c` from 1.75 to 1.65. No new version. No significant effect on results.

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: PySam, Numpy, Numba

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 (version >= 3.5)
2. Bowtie2 (optional)
3. Samtools

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. PySam (version >= 0.15.0)
2. Numpy (version >= 1.17.0)
3. Numba (version >= 0.50.0)

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

1. `git clone https://github.com/AnantharamanLab/PropagAtE`
2. `cd PropagAtE`
3. `pip install -e .` ← *NOTE*: don't forget the dot (`pip install -e [dot]`)

Installing with pip is optional but suggested. Using pip will collect dependencies and add PropagAtE (`Propagate` executable) to your system PATH. Without pip, PropagAtE can still be executed directly from the git clone, just ensure executable permissions (`chmod +x Propagate/*`). Note that a new folder (`PropagAtE.egg-info`) should appear after installing with pip.

Testing PropagAtE

Test out a small dataset of mixed active and dormant prophages. These examples assume the command is being called from the `example_output/active` or `example_output/dormant` folders.

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.

5. *Dormant prophage test:* 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. You may need to specify `python3` at the beginning of the command.
    1. `cd example_output/dormant`
    2. `Propagate -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_dormant --clean -t 2`
6. *Active prophage test:* The inputs are a sorted BAM format alignment file and a manually generated prophage coordinates file.
    1. `cd example_output/active`
    2. `Propagate -f AE017333_partial_genome.fasta -b AE017333_partial_genome.sorted.bam -v manual_prophage_coordinates_AE017333.tsv -o PropagAtE_example_results_active`

Due to large file sizes the full data (i.e., full alignment and read sets) for the active prophage example could not be uploaded to GitHub. Please see the read set SRR1137233 from Hertel et al. 2015 and the genome AE017333.1 for the full data.

Flag Descriptions

Input Data

Quick Guide

1. Specify `-f` and `-v`
2. Pick a coverage input (`-b`, `-s`, `-r`, `-i`, `-u`)
3. (optional) Provide `-o` and `-t`
4. (optional) Modify the methods and outputs with additional flags

Required

Both `-f` and `-v` are required for every run

- `-f`: input genomes/scaffolds (fasta file). This file should contain sequences that include prophage and host regions, not strictly prophages themselves. The definition lines cannot have special characters, namely quotations, pipe symbol or commas. PropagAtE requires at a minimum 1000bp of host and 1000bp of prophage to run analyses. Only the

scaffolds indicated by `-v` will be considered, but this `-f` file can contain extra sequences that will be ignored.

- `-v`: prophage coordinates input in the format of either (1) VIBRANT results coordinate file or (2) manually generated coordinate file. See the next two bullet points.
    - *VIBRANT method*: an automatically generated results file can be used directly from a VIBRANT analysis (>= v1.2.1). The file will be 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*: if prophages were identified by a different method, a manually generate prophage coordinates file can be used 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`. Note that the terms `fragment` and `prophage` are interchangeable in this format.
        1. `scaffold` is the name of the entire host sequencing that contains the prophage(s). Example: `scaffold_999`
        2. `fragment` is the name of the prophage fragment. Example: `scaffold_999_fragment_1` or `prophage_`
        3. `start` is the nucleotide number where the prophage starts. Example: `2500`
        4. `stop` is the nucleotide number where the prophage stops. Example: `58000`

Pick one

For every run, pick one of the following options as input for coverage information. Only one file is given with the exception of `-r` in which forward and reverse read files are given. PropagAtE only functions on a single sample to identify prophage activity rather than multi-sample coverages.

- `-b`: input BAM sequence alignment file (sorted or unsorted). This will be sorted (if necessary) and indexed (if necessary). This format is used for analysis.
- `-s`: input SAM sequence alignment file. This will be directly converted to BAM format for analysis.
- `-r`: input paired short reads separated by a space. Example: `-r forward_1.fastq reverse_2.fastq`.
- `-i`: input interleaved paired short reads.
- `-u`: input unpaired short reads *NOTE*: For reads input (`-r`,`-i`,`-u`) Bowtie2 (--no-discordant with `-r`,`-i`) will be used to generate a SAM file, which is then converted to BAM format. Reads can be in gzip format.

Common

Here you can specify an output file and number of threads to use. Number of threads will mainly effect the runtime of Bowtie2.

- `-o`: name of an output folder to deposit results. If not specified, the default is 'PropagAtE_results' followed by the basename of `-v`.
- `-t`: number of threads to use for Bowtie2 mapping as well as Samtools converting/sorting/indexing.

Additional flags

These flags are often not used. However, they can be used to modify the method of coverage calculation or how active versus dormant is considered.

- `-p`: minimum percent identity per aligned read for calculating coverage. The default is 0.97 (97%). This option pertains to any coverage input (reads or SAM/BAM). Lowering this value will make the alignment filtering less strict.
- `-e`: minimum effect size for significance by Cohen's *d* test. The default is `0.70` and the minimum is `0.60`. Values greater than `0.70` 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 `2.0` and the minimum is `1.5`. Setting values below `2.0` may introduce false identifications (i.e., dormant prophages identified as active) whereas setting the value too high (e.g., `10`) may reduce identification of active prophages.
- `--mask`: mask coverage values bases on each end of a scaffold. The default is 150 bp. This will attempt to even out the coverage at scaffold ends where short reads are less likely to align properly/completely. This option pertains to any coverage input (reads or SAM/BAM).
- `--min`: minimum average coverage to consider a prophage present and for --depth. The default is 1.0x coverage. Prophage with an average coverage below this threshold will not be active regardless of host coverage. See `--depth` for another function of this flag.
- `--depth`: minimum depth of coverage as fraction of bases >= minimum coverage (`--min`). The default is 0.50 (50%). Example: with `--min 1.0 --depth 0.50`, if > 50% of a prophage genome has a coverage < 1.0, regardless of the average coverage, it will not be considered as active. This helps to ensure uneven and misleading alignments do not result in false positives.
- `--clean`: use this setting to remove any generated SAM, unsorted BAM and/or Bowtie2 index files. All user input data files (regardless of format) and sorted BAM files will always be retained. SAM/BAM files can be very large, and Bowtie2 index files are typically temporary. Off by default.

Output Explanations

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

- Log file: 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. Finally, the number of prophages identified as active will be listed. The log file may contain Error messages when applicable.

- 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:

    1. prophage: the name of the prophage

2. host: the name of the host
3. active: "active" indicates and active prophage in the lytic stage of infection. "dormant" indicates a dormant prophage in the lysogenic stage of infection. "ambiguous" indicates the prophage passed the `-e` and `-c` cutoffs but not the `--min` or `--depth` cutoffs, generally considering it as "not active" but lacking evidence to call it "dormant".
4. CohenD: the Cohen's *d* effect size for the prophage and host coverages
5. prophage-host_ratio: the prophage:host coverage ratio (prophage mean divided by host mean)
6. mean_difference: the difference between the prophage and host coverages (prophage mean minus host mean)
7. prophage_len: the length in nucleotides of the prophage region
8. prophage_mean_cov: the mean (average) coverage of the prophage region
9. prophage_median_cov: the median coverage value of the prophage region
10. prophage_sd_cov: the standard deviation of the prophage region coverage values
11. prophage_cov_depth: the depth of coverage of the prophage region
12. host_len: the length in nucleotides of the host region
13. host_mean_cov: the mean (average) coverage of the host region
14. host_median_cov: the median coverage value of the host region
15. 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

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see https://www.gnu.org/licenses/.