TNet-Geo (version 1.0)

https://compbio.engr.uconn.edu/software/tnet-geo

Description

TNet-Geo is a tool for geographical transmission network analysis when multiple strain sequences from different infected hosts are available from the different geographic regions (e.g., countries) under consideration. TNet-Geo is parameter-free and highly scalable and can be easily applied within seconds to datasets with thousands of strain sequences from hundreds of geographical regions. It is built upon the TNet tool, which uses multiple strain sequences per infected host to infer a host-to-host transmission network. We refer the reader to https://compbio.engr.uconn.edu/software/tnet/ for further details on TNet.

TNet-Geo takes as input a phylogeny of the sampled strain sequences and analyzes it to estimate the underlying geographical transmission network. In particular, it labels each internal node of the strain phylogeny with a geographic region label, and this labeled phylogeny can then be used to infer how the infectious disease was transmitted across regional boundaries. The underlying assumption on which TNet-Geo is based is that the transmission of infection is easier within a geographical region than between two different geographical regions.

Note that there are some fundamental differences between traditional (host-to-host) transmission networks and geographical transmission networks. Most importantly, in a traditional transmission network, the question of interest is “Who infected whom?”, while in geographical transmission networks, the relevant question is “What was the extent of infection spread from one region to another in a specified time frame?”. Accordingly, the output from TNet-Geo has to be interpreted differently compared to outputs from traditional transmission network inference tools.

Each run of TNet-Geo on the same input tree can result in a different estimate of the transmission network, and TNet-Geo should therefore be executed multiple times (say 100) on the input phylogeny and an aggregated transmission network should be constructed from the resulting outputs.

TNet-Geo implements algorithms described in the following papers:

> TNet: Transmission Network Inference Using Within-Host Strain Diversity and its Application to Geographical Tracking of COVID-19 Spread
> Saurav Dhar, Chengchen Zhang, Ion Mandoiu, Mukul S. Bansal
> *Under review*.

> TNet: Phylogeny-Based Inference of Disease Transmission Networks Using Within-Host Strain Diversity
> Saurav Dhar, Chengchen Zhang, Ion Mandoiu, Mukul S. Bansal
> *International Symposium on Bioinformatics Research and Applications (ISBRA) 2020*: LNCS 12304: 203-216.

TNet-Geo is implemented in Python (version 3.0 or greater) and is freely available open source under GNU GPL. TNet-Geo also uses the Biopython library that can be freely downloaded as described at https://biopython.org/.

Usage

TNet-Geo takes as input a single rooted binary phylogeny on all strain sequences sampled from all geographical regions considered in the analysis. This phylogeny must be in Newick format. Such a phylogeny can be constructed using standard phylogeny construction tools such as RAxML or PhyML and then rooted using standard rooting methods.

TNet-Geo allows for either a basic transmission network inference analysis or a more advanced time-series transmission network analysis. We discuss these two types of analyses below:

Basic transmission network analysis*:* For this analysis, each leaf label in the input strain phylogeny must be in the format <regionName>_<strainID>. TNet-Geo will output a version of the strain phylogeny in which each internal node is labeled with one of the geographical regions under consideration. This Newick format output tree can then be easily parsed (by the user) to infer a geographical transmission network where the nodes are regions and weighted directed edges represent the magnitude of transmission from one region to another. *For improved accuracy, we strongly recommend aggregating such results over multiple runs of TNet (say 100) and also over multiple candidate input phylogenies (say 10).*

For example, a basic transmission network analysis can be performed using the following command:

```
python3 tnet_geo.py inputFile outputFile
```

The `-t` command line option can be used to automatically run TNet multiple times on the same input file and generate multiple randomly sampled optimal outputs. For example, the following command can be used to generate 100 transmission network samples automatically:

```
python3 tnet_geo.py inputFile outputFile -t 100
```

Time-series transmission network analysis**:** A more useful kind of geographical transmission network analysis is one where one can study how transmission rates between different regions change over time. TNet-Geo supports such time-series transmission network analysis. For this analysis, each leaf label in the input phylogeny must correspond to a unique “strainID” and each internal node must have a unique label. In addition, a separate metadata file must be provided using the `-md` command line option. This metadata file must be in CSV format and must specify an estimated “date” for each leaf and internal node in the input phylogeny along with the “regionName” for each leaf node. The node labels should appear in the “strain” (first) column. The corresponding dates should appear in the “date” (second) column and the corresponding regionNames should appear in the “country” (third) column; see the sample input metadata file. All node labels should exactly match node names given in the input Newick file and all dates should be in “YYYY-MM-DD” format. For internal (non-leaf) nodes the “country” column should be empty. A sample metadata file is provided in the “input” directory (see below).

For example, a metadata file can be specified using the `-md` command line option as follows:

```
python3 tnet_geo.py inputFile outputFile -t 100 -md metadata.csv
```

The output of the above command will, again, be a version of the strain phylogeny in which each internal node is labeled with one of the geographical regions under consideration. To facilitate the kind of time-series analysis described above, TNet-Geo provides the option of outputting a more detailed file containing additional information. This file, in JSON format, can be generated by using the `-ex` command line option as follows.

```
python3 tnet_geo.py inputFile outputFile -t 100 -md metadata.csv -ex
```

The resulting JSON file consists of three blocks of information that can be analyzed further. The first block is the "Country of exposure" which lists, the region(s) of exposure, along with their counts in multiple runs, for each strain (leaf node) included in the analysis. The second block is "Transmission edges" which lists all transmission edges inferred in at least one of the multiple samples along with the number of samples in which that transmission was observed. The third block is "Dated edges" which is a collection of all the transmission edges inferred in the specified number of multiple samples along with their corresponding dates. Note that each transmission edge may appear multiple times in this third block even if only one sample is computed (since multiple transmissions may have occurred between the same two geographical regions), possibly with different dates.

For Country of exposure the output format is:

```
      "EPI_ISL_457979": {"count": {"Oman": 39, "UnitedKingdom": 35,
      "Belgium": 15, "Morocco": 11}, "country": "Oman"}
```

For "Transmission edges" the output format is:

```
      "Germany->Italy": 88
```

For "Dated edges" the output format is:

```
      ["Italy->Hungary", "2020-02-10"]
```

Note 1: *For any analysis using TNet-Geo, we strongly suggest including roughly comparable numbers of strain sequences from each region under consideration so as not to bias geographical transmission network inference results.*

Note 2: *By default, TNet-Geo executes the uniform random sampling algorithm described in the manuscripts cited above. Even though options for minimum back-transmission and highest-probability solution sampling are also available, we strongly recommend against changing this default setting. In particular, the minimum back-transmission sampling is not well suited for geographical transmission network inference.*

List of command-line options

Usage:

```
tnet_geo.py INPUT_TREE_FILE OUTPUT_FILE [-h] [-sd SEED] [-bs] [-mx] [-
md METADATA] [-t TIMES] [-ex] [-v]
```

```
positional arguments:
  INPUT_TREE_FILE       input file name
  OUTPUT_FILE           output file name

optional arguments:
  -h, --help                    show this help message and exit
  -md, --metadata               csv file with meta data for the tree nodes
  -sd SEED, --seed SEED         random number generator seed
  -bs, --biasedsampling         sample optimal solutions with minimum back-transmission
  -mx, --maxprob                compute highest-probability solution
  -t, --times                   sample TNet multiple times
  -ex, --extradata              output a JSON file with extra output data for further analysis
  -v, --version                 show program's version number and exit
```

Example datasets

The "input" directory contains two sample input phylogenies and a metadata file that can be used as input for TNet.

For example,

```
python3 tnet_geo.py input/0.tree output/0.out -md input/0.metadata.csv
-t 100 -ex
```

and

```
python3 tnet_geo.py input/1.without_metadata.tree output/0.out -t 100
```

Further analysis of the JSON file

To easily replicate the kind of analysis we performed in the first paper cited above, we provide another python script "analyze_json.py" that takes as input the JSON file described earlier and groups all transmission edges in the "dated edges" block by month. It generates a CSV file listing all inferred transmission edges and their respective counts during each month. The "output" folder includes a sample output CSV file: "0.edgegroups.csv".

This script can be used as follows:

```
python3 analyze_json.py inputFile.json -eg OutputFile.csv
```

(Note that the `–eg` option is required.)