TreeFix-VP Manual

Mukul Bansal
Department of Computer Science
University of Connecticut

Samuel Sledzieski
Department of Computer Science
University of Connecticut

July 30, 2018

Contents

1 Introduction

TreeFix-VP is a program for reconstructing highly accurate viral phylogenetic trees. TreeFix-VP incorporates host information with sequence data to reconstruct the tree using a mixed maximum likelihood / maximum parsimony framework. Given a maximum likelihood phylogeny and a multiple sequence alignment, TreeFix-VP searches among multiple topologies which are supported equally by sequence data (using the Shimodaiara-Hasegawa test statistic). The selected topology is the one that minimizes the number of necessary transmissions.

Requirements

- Python (2.5.4 or greater)
- C compiler (gcc)
- SWIG (1.3.29 or greater)
- Numpy (1.5.1 or greater)
- Scipy (0.7.1 or greater)
- Additionally, Python modules are required for computing the p-value for likelihood equivalence.

Likelihood TreeFix-VP uses the Shimodaira-Hasegawa (SH) test statistic with RAxML site-wise likelihoods to compute p-values for each candidate tree.

Parsimony TreeFix-VP scores each statistically equivalent candidate tree using Fitch's algorithm. Given host labels at the leaf nodes, the Fitch module computes a score equivalent to the minimum necessary number of transmissions needed to label the internal nodes.

2 Tutorial

Usage `treefixVP -[options] <treefile>`

Input Treefix-VP requires a seed tree, generally a maximum likelihood tree, and a multiple sequence alignment.

Options TreeFix-VP assumes that the multiple sequence alignment and seed tree have the same root name, with different extensions. The default extensions are ".align" and ".tree", but other extensions can be specified. The output tree file will have the same root name, and will have either the default extension ".treefix.tree", or a user specificed extension.

- `-A <alignment file extension>, --alignext=<alignment file extension>`
- `-o <old tree file extension>, --oldext=<old tree file extension>`
- `-n <new tree file extension>, --newext=<new tree file extension>`

The default likelihood model uses RAxML site-wise likelihoods, but a different user-defined module can be substituted. The default likelihood test is the SH test, but again a user-defined test can be used.

- `-m <module for likelihood calculations>, --module=<module for likelihood calculations>`
- `-e <extra arguments to module>, --extra=<extra arguments to module>`
- `-t <test statistic>, --test=<test statistic>`
- `--alpha=<alpha>`: alpha threshold (default: 0.05)
- `-p <alpha>, --pval=<alpha>`: same as alpha

The default module for transmission cost uses Fitch's algorithm. A user defined cost model can be substituted.

- `-M <module for transmission cost calculation>, --smodule=<module for transmission cost calculation>`

A number of options can be specified to guide the search conducted by TreeFix-VP.

- `-x <seed>, --seed=<seed>`: seed value for random generator
- `--niter=<number of iterations>`: number of iterations (default: 1000)
- `--nquickiter=<number of quick iterations>`: number of subproposals (default: 100)

Finally, there are a handful of informational options.

- `--version`: show program's version number and exit
- `-h, --help`: show the help message and exit
- `-V <verbosity level>, --verbose=<verbosity level>`
- `-l <log file>, --log=<log file>`: log filename. Use '-' to display on stdout.

3 Preliminary Test Results

TreeFix-VP was tested on 8 sample trees and multiple sequence alignments. Below is a list of the 8 datasets. The column labeled "Old Score" is the Fitch score of the original tree, and the column labeled "New Score" is the Fitch score of the tree reconstructed by TreeFix-VP. Also included are the number of leaves for the given tree, and the time of the run.

Name	**Leaves**	**Old Score**	**New Score**	**Runtime (minutes)**
AA	118	7	5	48
AB	48	2	1	28
AC	86	3	3	32
AD	76	2	1	67
AE	29	3	2	15
AH	53	2	1	31
AI	163	71	51	71
AJ	49	3	3	21

3.1 Example

Find a sample dataset that we can release and walk through an example here

4 Testing Pipeline

This section describes the pipeline which we have developed for testing of TreeFix-VP. In this pipeline, ground-truth phylogenies, contact networks, and gene sequences are generated using FAVITES [1]. The sequences are then used to generate a maximum likelihood phylogeny using RAxML [2].

The RAxML phylogeny and sequences are provided as input to TreeFix-VP, which will generate the error-collected phylogeny. The two phylogenies are then scored using the transmission cost calculated by Fitch's algorithm, and are compared to the ground-truth phylogeny using Robinson-Foulds distance. This process can be done using three separate scripts, `data_gen`, `tree_gen`, and `tree_score`, or by running `full_test`. All steps require the inclusion of a `run name` argument, which will be used to connect the different scripts. A directory with the run name will be created in the current working directory containing all files generated by the test.

Additional Requirements

- RAxML
- FastTree
- ete3
- dendropy

4.1 data_gen

This script will generate test data using a FAVITES configuration file. Some preprocessing is done to ensure that the sequences and phylogeny can be processed by TreeFix-VP, including renaming of tree nodes. A directory will be created using the provided run name, and two sub-directories will also be created. `FAVITES_output/` is set as the output directory for FAVITES, while `clean_data/` contains all data necessary for subsequent steps of the pipeline.

Usage: `./data_gen [options] <run name>`

Options:

- `-h`: Show help message
- `-s <extension>`: Sequence file extension. Default is '.fasta'
- `-t <extension>`: Tree file extension. Default is '.tree'
- `-n <extension>`: Network file extension. Default is '.network'
- `-c <file>`: Required argument. FAVITES configuration file
- `--favites=<path>`: Path to run_favites.py if not on path
- `-l <file>`: Log file name. If not set, default is `<run name>.log`

4.2 tree_gen

This script will calculate a maximum likelihood phylogeny using RAxML. The default settings for RAxML are outlined below, but non-default parameters can be passed using `--raxopts`. The RAxML tree and the sequence alignment from `data_gen` are used as inputs to TreeFix-VP, which will generate an error corrected phylogeny in `<run name>/clean_data/`. By default 1000 iterations of TreeFix-VP are run, however the number of iterations as well as other non-default parameters can be passed to TreeFix-VP using `--treefixopts`. All RAxML output files are written to `RAxML_output/`, and the maximum likelihood tree is copied into `clean_data/`

Usage: `./tree_gen [options] <run name>`
Note that this run name must match the run name provided to `data_gen`.

Options:

- `-h`: Show help message
- `-s <extension>`: Sequence file extension. Should match the extension used for `data_gen`
- `-r <extension>`: ML tree file extension. Default is '.raxml'
- `-f <extension>`: TreeFix tree file extension. Default is '.treefix'
- `--fast`: This flag can be set to use FastTree instead of RAxML for quick ML tree generation
- `--ml=<path>`: Path to RAxML if not on path
- `-l <file>`: If a non-default log file was used for `data_gen`, it should be specified here to maintain a singular logging location

4.3 tree_score

This script will score both the maximum likelihood and TreeFix-VP trees using a variety of metrics. Fitch Score represents the minimum necessary number of transmissions, calculated using Fitch's algorithm for the small parsimony problem. The Fitch score of the simulated tree should be equal to the number of transmissions performed by FAVITES. Each tree will also be compared to the true tree generated by FAVITES using the Robinson-Foulds distance. For more information about these metrics see Sections 1.6 and 4.3 of *Computational Phylogenetics* [3].

Usage: `./tree_score [options] <run name>`
Note that this run name must match the run name provided to `data_gen`.

Options:

- `-h`: Show help message
- `-r <extension>`: ML tree file extension. Should match the extension used for `tree_gen`
- `-f <extension>`: TreeFix-VP tree file extension. Should match the extension used for `tree_gen`
- `-t <extension>`: Simulated tree file extension. Should match the extension used for `data_gen`
- `--fitch=<path>`: Path to fitch.linux if not on path
- `-o <file>`: Desired outfile for scores. By default, scores will be written to `<run name>.out`
- `-l <file>`: If a non-default log file was used for `data_gen`, it should be specified here to maintain a singular logging location.

4.4 run_treefix_test

This script will call `data_gen`, `tree_gen`, and `tree_score` consecutively to perform a full simulation testing run. All options from the previous scripts can be used here. If none are specificed, the default options are used.

Usage: `./run_treefix_test [options] -c <config file> <run name>`

5 Test Parameters

6 Changes from TreeFix-DTL

TreeFix-VP was built by modifying TreeFix-DTL [4], with the most significant change being the replaced cost module (fitch.linux rather than ranger-dtl). All changes are documented below.

6.1 treefixVP

- Based on treefixDTL script
- Removed most options in parser, only need alignment extension, old extension, new extension
- Changed default smodule to FitchModel
- Call treefix_for_VP rather than treefix

6.2 fitchmodel.py

- Copy of rangerdtlmodel.py
- Changed compute_cost() to call fitch.linux executable
- Removed stree, smap from FitchModel.optimize_model
- Removed call to CostModel.optimize_model()

6.3 treefix_for_VP

- Renamed treefix to treefix_for_VP - this avoids path collision with treefix and treefixDTL
- Removed smap and stree as required arguments
- Changed to common.check_req_options(species=False)
- Removed stree and gene2species from check_input_tree()
- Removed stree and gene2species from search_landscape()
- Removed check_congruent_tree() function, no longer applies
- Removed 'if flag: return mintree'
- Removed reading species tree and species map from main()
- Removed stree and gene2species from calls to optimize_model(), check_input_tree(), and search_landscape() in main

6.4 General

- Removed all ranger executables
- Removed treefix_compute
- Removed rangerdtl model from models module
- Modified setup files to install the proper executables
- Updated treefixVP.py with version and software info
- Updated INSTALL.txt, CHANGES.txt, and README.txt
- Added README.pdf

References

[1] Niema Moshiri. FAVITES: simultaneous simulation of transmission networks, phylogenetic trees, and sequences. https://www.biorxiv.org/content/early/2018/04/18/297267, 2018.

[2] A. Stamatakis. RAxML Version 8: A tool for Phylogenetic Analysis and Post-Analysis of Large Phylogenies. *Bioinformatics*, 2014.

[3] Tandy Warnow. *Computational Phylogenetics*. Cambridge University Press, 2017.

[4] Yi-Chieh Wu, Matthew D. Rasmussen, Mukul S. Bansal, and Manolis Kellis. Treefix: Statistically informed gene tree error correction using species trees. *Systematic Biology*, 2013.