Introduction

TreeClimber is a computer program that the parsimony-based test statistic described by Maddison and Slatkin (2, 4, 5) and applied to microbial communities by Martin (3) to determine whether the observed difference between the phylogenies of multiple communities are due to an accumulation of evolutionary variation or some perturbation. The goals of the analysis are very similar to those of the LIBSHUFF-type programs and previous implementations with several key exceptions:

1. ***Phylogenetic trees as input:*** The user to select whatever phylogenetic reconstruction method they prefer without being forced to use a distance matrix.
2. ***Multiple libraries compared without increasing Type I error:*** Any number of treatments may be included in a single analysis without having to do multiple pairwise comparisons.
3. ***No pre-formatting required:*** Phylogenetic trees can be constructed in PAUP*, Phylip, Mega, MrBayes, and ARB without modifying the input file.
4. ***Programmed in C++ and freely available as a single program:*** Implementation of the method is streamlined and does not require multiple proprietary software packages.

This manual is designed to achieve four goals:

1. Show how the scoring of a tree works
2. Demonstrate the usage of TreeClimber
3. Explain output
4. Answer frequently asked questions

If you have any questions, complaints, or praise, please do not hesitate to contact Dr. Patrick D. Schloss at pds@plantpath.wisc.edu

How to Compile TreeClimber

To compile TreeClimber in LINUX or MacOSX, type the following in the folder with the source code:

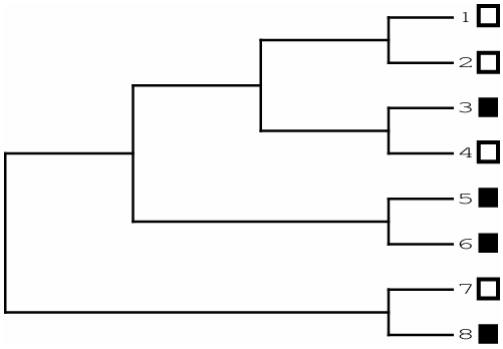
>g++ treeclimber.C –O4 –o treeclimber

If you are using Windows, then you need the precompiled version of TreeClimber available on the website. Be forewarned that TreeClimber does not seem to run as quickly in Windows as it does in Linux and I would encourage everyone to align their sequences in ARB, which uses Linux or OSX, and to run TreeClimber in the same operating system.

Generating Input Files

To test the significance of treatments using TreeClimber you must provide a file containing one or more trees and a file containing the exact name of each sequence and a simple descriptor for what treatment the sequence belongs to. The tree file can be generated manually in newick format or using phylip or nexus formatted trees from any phylogenetic tree reconstruction package such as PHYLIP, PAUP*, MrBayes, or MEGA. The tree file can contain as many trees as desired.

For example, the tree shown here:



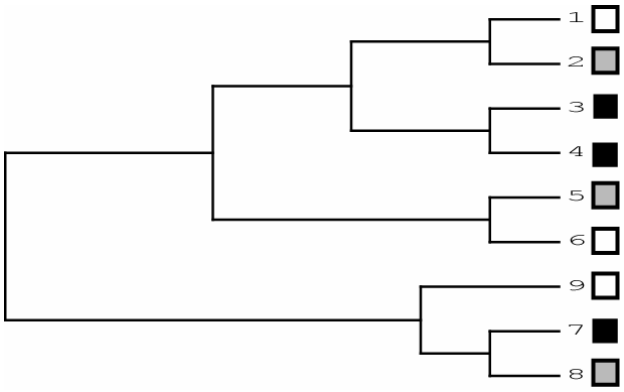
could be represented by the following newick formatted tree file (binary.tree):

((((1,2),(3,4)),(5,6)),(7,8))

The names file (binary.names) would contain the following:

1 white
2 white
3 black
4 white
5 black
6 black
7 white
8 black

A more complicated experimental design with three variables could have a tree that looks like:



With a newick formatted tree in the file triple.tree that looked like:

((((1,2),(3,4)),(5,6)),(9,(7,8)))

The triple.names file could look like:

1 w
2 g
3 b
4 b
5 g
6 w
7 b
8 g
9 w

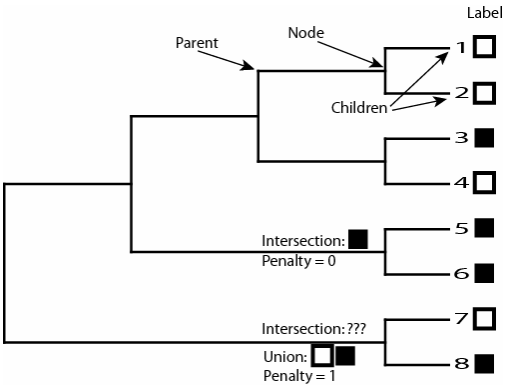
Note that the treatment designations white/black w/g/b could just as easily been w/b, a/b, 1/2, etc. as long as there are no spaces in the treatment names. There is one exception to this. The treatment name xxx is reserved by the program as a way to ignore certain treatments. For example, in the previous case, if you only wanted to compare the "w" and "b" treatments, you would replace the "g" treatment with "xxx" as is shown here:

1 w
2 xxx
3 b
4 b
5 xxx
6 w
7 b
8 xxx
9 w

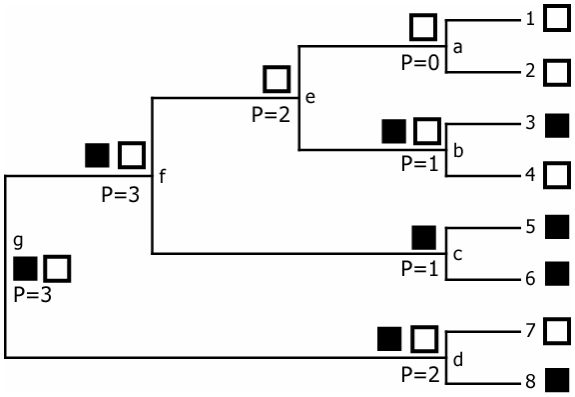
The nice thing about this feature is that it does not require changing the tree file, only the name file.

How analysis works

Given a phylogenetic tree where each taxa is labeled according to its treatment, the most parsimonious number of transitions between labels can be calculated to explain the covariation of the phylogeny with treatment using Fitch's algorithm (1). This approach is advantageous because a score can be calculated rapidly and it is possible to analyze trees using any number of treatments.



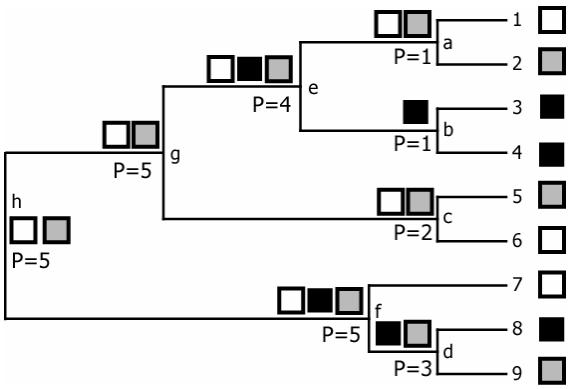
A phylogenetic tree is composed of linked nodes describing the relationship between the various entities being compared. In a bifurcating tree, there are two children for each node and each node has one parent. For example, in the tree to the right, the children of the indicated node are branches 1 and 2 and its parent is indicated. The parsimony test algorithm uses the labeled phylogenetic tree to determine the appropriate label or labels for each of the 7 nodes (a-g) in the tree based on the labels of each node's children. The test uses the Fitch parsimony algorithm, which first compares the labels of the two children of each node. If there is an intersection between the two sets of labels, then the node is labeled with the result of the intersection and a penalty is not assessed to the parsimony score. If there is not an intersection between the two children of the node, then the union of the two labels is applied and a penalty of 1 is assessed to the to the parsimony score. This scoring algorithm can be applied to any bifurcating tree with as many different labels as are desired.



The figure to the left shows the two label case for an 8 branched (branches 1 through 8) phylogenetic tree with where the sequence identifier has been replaced with a colored (white or black) block representing one of two different treatments (white or black). This example was described above for the binary.tree/binary.names example. The algorithm is then applied to the 7 nodes (nodes a through g) of the phylogenetic tree. To determine the label for node a, intersection of the labels from branches 1 and 2 are compared and their intersection is retained (i.e. white) without increasing the parsimony score. Because there is no intersection for the labels of branches 3 and 4 (i.e. white and black), the union of the two labels is applied to node b and the parsimony score is increased from 0 to 1. Labels for nodes c and d are determined similar to that of nodes a and b, respectively so that the parsimony score is now 2. Node e's label is determined by the intersection of the labels of nodes a and b. Since node a's label was white and b's was white and black, their intersection is white and this label is applied to node e without increasing the parsimony score. The label of node f is white and black since the labels for nodes b and c are white and black, respectively, and the parsimony score is increased to 3. The final node, g, is

evaluated by the intersection of the labels from nodes d and f. Since the label of both of the nodes is black and white, the label of node g is also black and white. The final parsimony score for this tree is 3.

A slightly more complicated example was the three-treatment case described above. The figure to the right demonstrates the algorithm's ability to calculate a parsimony score for a tree with three labels (black, white, and gray) for a 9 branched tree (branches 1 through 9) with 8 nodes (nodes a through h). The labels for nodes a, b, c, and d are straightforward to determine and the resulting parsimony score after evaluating these nodes is 3. Node e is the union of the labels white and gray from node a and black from node b, which results in labeling node e with all three labels and increasing the parsimony score to 4. A similar union is performed at node f and the score increases to 5. The label for node g is the intersection of the labels black, white and gray with gray and white yielding the label gray and white; there is no change in the parsimony score. Similarly, the label for node h is gray and white. The final parsimony score for the tree in Figure 1B is 5.



How to Run TreeClimber to Analyze Input Tree(s)

In its typical usage, TreeClimber is run from the command line prompt with a file containing a tree(s) and a file containing the names of each sequence and its treatment designation.

```
>./treeclimber -n binary.names -t binary.tree
```

The precision of the Monte Carlo procedure can be altered by increasing or decreasing the number of iterations that the sampling without replacement procedure is run. The default value is 1,000 but can be changed to 10,000 or any other number as follows:

```
>./treeclimber -i 10000 mccaig.dist
```

Execution in Windows and Linux (and Mac OSX) is essentially the same. In Windows, you cannot merely double click on the icon to get the program to execute. You must use the "Command Prompt" program found by going Start -> Program Files -> Accessories -> Command Prompt. Then you must type in the path of TreeClimber and your tree and names files to execute the program:

```
C:\> "Documents and Settings\pds\Desktop\treeclimber.exe" -t "Documents and
Settings\pds\Desktop\binary.tree" -n "Documents and
Settings\pds\Desktop\binary.names"
```

Alternatively, you can change the root path to move to the desired directory and execute TreeClimber from there:

```
C:\PATH\> treeclimber.exe -t binary.tree -n binary.names
```

Once executed, the program will begin to churn and you will see the progress of the calculations and you will be given a table on the screen and in an output file.

If you would like to analyze replicate trees in the same analysis, then the trees need to be in the same file. This can be done easily by using PHYLIP's programs to generate bootstrap replicate trees, PAUP*, or MrBayes. To see the structure of these datafiles and guidance on how to generate them, please consult the example datasets on the "Downloads" page. We have successfully used trees generated in PHYLIP, PAUP*, MrBayes, and ARB as input files for TreeClimber. Presently, if you generate a tree in ARB, then when you export the tree you must tell ARB to not include branch lengths in the tree; this will be fixed in future versions of TreeClimber.

How to Run TreeClimber to Analyze Random Trees

If you do not have any input trees to analyze but are interested in the random distribution of parsimony scores for various experimental designs you can do this analysis with the "-r" flag.

>./treeclimber –r

You will then be prompted for the number of groups you would like to run the analysis for and the number of sequences in each group.

Output Files

TreeClimber generates a file whose root is the same as the tree file name and has the extension "*.p". The first column in this file is the score of trees. The second column is the proportion of trees in your input file that had that score. The third column is the cumulative proportion of trees that have had that score. The fourth and fifth columns represent the same data as columns two and three, except these values were taken from the scores of the randomly generated trees. The five columns in the "*.p" file are also outputted to the screen.

Frequently Asked Questions

How do I cite TreeClimber?

Schloss, P.D. & Handelsman, J. 2005. Introducing TreeClimber, a test to compare microbial community structure. *Applied and Environmental Microbiology*. **In press.**

In windows, why does the command window open and close quickly when I double click on the TreeClimber icon in windows?

This is because you haven't given TreeClimber an input file and you will get an error message quickly followed by the screen closing. Please see the above section on how to run TreeClimber and remember that it must be run from a command prompt in windows.

I use XYZ program to construct distance trees. Can I use TreeClimber?

We are anxious to help people use TreeClimber in a way that is easiest for them. Please contact me (pds@plantpath.wisc.edu) with an example tree file, and I will incorporate the format into TreeClimber.

Why doesn't TreeClimber…?

If you would like to see something added to TreeClimber, please let me know (pds@plantpath.wisc.edu). It may take me a while to get around to implementing the feature, but I am generally reasonable. For example, providing the coverage data in an output file was implemented because people asked about it.

References

1. **Fitch, W. M.** 1971. Toward defining the course of evolution: Minimum change for a specific tree topology. Syst Zool **20:**406-416.
2. **Maddison, W. P., and M. Slatkin.** 1991. Null models for the number of evolutionary steps in a character on a phylogenetic tree. Evolution **45:**1184-1197.
3. **Martin, A. P.** 2002. Phylogenetic approaches for describing and comparing the diversity of microbial communities. Appl. Environ. Microbiol. **68:**3673-3682.
4. **Slatkin, M., and W. P. Maddison.** 1989. A cladistic measure of gene flow inferred from the phylogenies of alleles. Genetics **123:**603-613.
5. **Slatkin, M., and W. P. Maddison.** 1990. Detecting isolation by distance using phylogenies of genes. Genetics **126:**249-260.