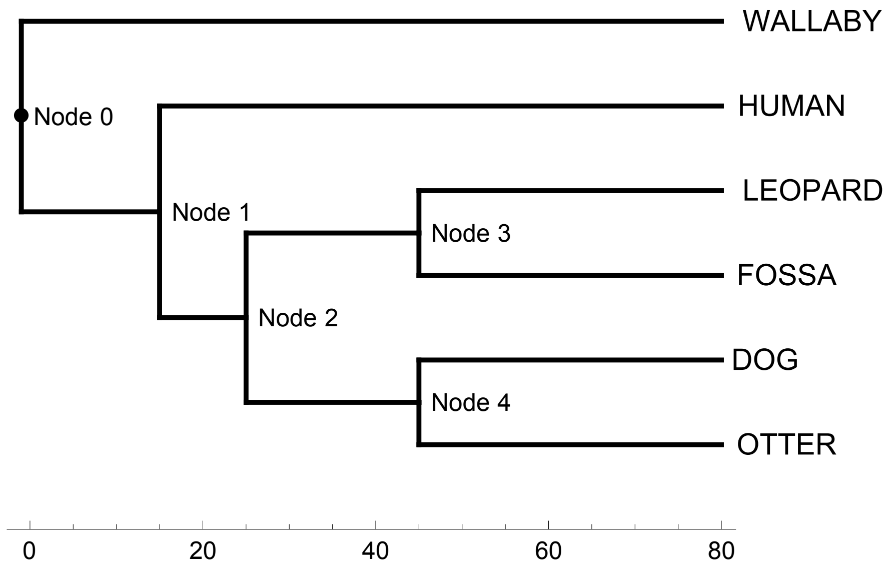
Phylogenetics for Mathematica

User's Guide

Version 2.0, July 2012

This package is used in Indiana University courses
Geol-G 562 Geometric Morphometrics
Geol-G 563 Quantitative Paleontology



P. David Polly
Department of Geological Sciences
Indiana University
Bloomington, Indiana 47405 USA

http://mypage.iu.edu/~pdpolly/
pdpolly@indiana.edu

Table of Contents

Installing the package

1. Download the latest version of the package at http://mypage.iu.edu/~pdpolly/Software.html (right click on link to save as file)
2. Open the file in Mathematica
3. Under the File menu, choose "Install"
4. Under Type of Item choose "Package", under source choose the file you just saved, under Install Name choose a short name for the package (e.g., "PollyPhylogenetics")
5. Once installed, enter the command "<<PollyPhylogenetics` " to use the functions.
6. Use the function *PhylogeneticsVersion[]* to determine which version you have installed.

Using Mathematica

Mathematica has a unique interface that takes a while to get used to. You open to a blank page, like a word processor, where you can type anything youwant. Most of the time you will type commands that do things with your data: connect to databases, plot graphs, carry out calculations. Unlike other statistical or mathematical programs, the commands you type and the output you get remain on the page, which gives you a record of what you've done step-by-step. To help organize your work, you can add headers, format boxes, etc. with the Format Menu.

Cells are an important organizing feature of *Mathematica*. Note the "cell" markers on the right margin. Each cell is bounded by a bracket and commands within a cell are executed together. You can open and close cells by double clicking the bracket. This can be useful if you have lots of stuff in a notebook… you can give a section a heading, which causes cells in that section to be grouped, after which you can close the section by double clicking.

Shift + Enter causes a cell to be executed. Pressing the enter key creates a new line, just like in a word processor, but when you want to execute a command you typed, you type SHIFT+ENTER somewhere in the cell and all commands in the cell are executed.

Mathematica Commands. Mathematic is designed to be as easy to learn as possible so that you can concentrate on working instead of the program. Almost all commands are English words written out in full with capitals at the beginning of words and brackets [] at the end of the command. For example, the command to calculate an average of a set of numbers is *Mean[]*, the command to do a principal components analysis is *PrincipalComponents[]*, and the command to take the logarithm of a number is *Log[]*.

Formatting Output. Mathematica is clever about how it provides output and it tries to keep the results as accurate as possible. For example, if you calculate the average of the following numbers

```
Mean[ {1, 5, 10, 3, 20, 40} ]
```

the answer extends to many decimal places, so Mathematica reports it more precisely as a fraction: 79/6. You may want an ordinary number, however, and you can force Mathematica to format its output the way you want:

```
Mean[ {1, 5, 10, 3, 20, 40} ] // N
```

This command now gives you 13.1667. Another useful formatting function is **//MatrixForm**, which causes a table to be displayed neatly in columns instead of wrapping around the page.

Graphics. Mathematic is good at graphics. You can either use simple functions like *ListPlot[]* to create a generic graph, or you can experiment with *Graphics[]* to create a completely customized graphic.

Basic phylogenetic functions

DrawNewickTree[*tree*]

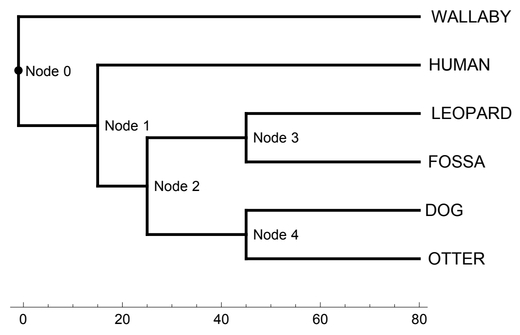
Parses a Newick tree string and renders it graphically as a tree with the tips and nodes labeled and the branches drawn proportional to the branch lengths in the file. This function requires branchlengths.

Arguments:

- *tree* is a string containing a tree with branch lengths in Newick format or a tree table of the sort produced by TreeToTable[].

Example:

DrawNewickTree[tree]



IndependentContrasts[*tree, trait*]

This function calculates standardized independent contrasts (Felsenstein, 1985, 2004) for a single continuous trait on a tree.

Arguments:

- *tree* is a string containing a tree with branch lengths in Newick format or a tree table of the sort produced by TreeToTable[].
- *trait* is a two column matrix with taxon names in the first column and trait values in the second column. Taxon names must match the names in *tree*.

Example:

contrasts = IndependentContrasts[tree, trait];

MakePHYLIPLabels[*labels*]

Reformats a list of labels to be compatible with PHYLIP. Labels that are longer than 10 characters are truncated, and those that are shorter than 10 characters are padded with spaces.

Arguments:

- *labels* is a list of strings to be used as OTU labels for PHYLIP.

Example:

```
phyliplabels = MakePHYLIPLabels[labels]

{"WALLABY   ", "LEOPARD   ", "HUMAN     ", "OTTER     ", "FOSSA     ", "DOG       "}
```

PhylogeneticMatrices[*tree*]

Creates the phylogenetic matrices needed for ancestral node reconstruction and other comparative statistical methods based on Generalized Linear Models (GLM). The function produces three matrices: the first describes the shared ancestry of the tip taxa (VarY), the second describes the shared ancestry between the tips and the nodes (VarAY), and the third describes the shared ancestry of the nodes (VarA). Matrices are returned with row and column labels. These matrices are described by Martins and Hansen (1997).

Arguments:

- *tree* is a string containing a tree with branch lengths in Newick format or a tree table of the sort produced by TreeToTable[].

Example:

```
{varY, varAY, varA} = PhylogeneticMatrices[tree];
```

PhylogeneticsVersion[]

Prints the version number and citation for the current installation.

Example:

```
PhylogeneticsVersion[]

PollyPhylogenetics 2.0
```

RandomWalk[*n, i*]

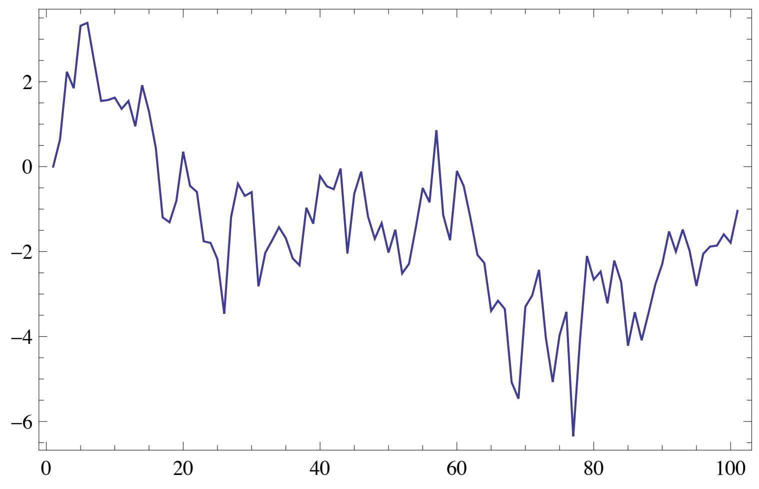
This function performs a Brownian motion random walk for *n* generations with a step rate of *i*. The walk performed by this function has a stochastically variable rate, with the change at each generation is drawn from a normal distribution whose mean is zero and whose standard deviation equals *i.*

Arguments:

- *n* is the number of steps (generations) in the random walk.
- *i* is a positive real number for the per-step rate of change

Example:

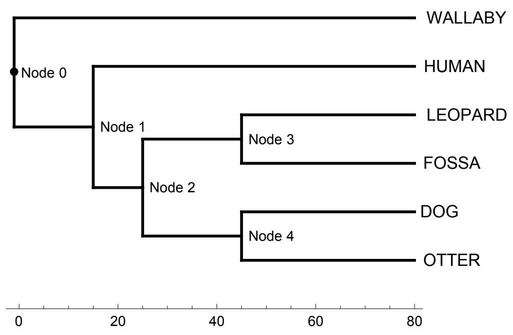
```
walk = RandomWalk[100, 1];

ListPlot[walk, Axes->False, Frame->True, Joined->True, PlotRange->All]
```



ReadNewick[*filename*]

Reads one or more Newick format trees from a text file and prepares them for use by removing carriage returns, line feeds, and other extraneous characters. A Newick tree is represented as a string consisting of tip names, branch lengths (separated from tip names or nodes by colons), and parentheses. Newick tree formats are described in detail in Felsenstein (2004). For example, this Newick string describes the following tree:

```
(WALLABY:80,(HUMAN:65,((LEOPARD:35,FOSSA:35):20,(DOG:35,OTTER:35):20):10):15);
```



Arguments:

- *filename* is the name, with path if needed, of a text file containing only Newick format trees.

Example:

tree=ReadNewick["/Users/Data/mytree.tre"]

(WALLABY:80,(HUMAN:65,((LEOPARD:35,FOSSA:35):20,(DOG:35,OTTER:35):20):10):15);

ReconstructNodes[*Tree, Trait*]

This function returns the estimated rate of evolution of the trait on the tree, a list of reconstructed values of trait Y for each node on the tree, and the standard deviation of the node estimate due to variation in the evolutionary process.

The per-step rate of evolution is estimated as the square-root of the mean of the squared standardized independent contrasts (Felsenstein, 1985), which is equivalent to the method described by Martins (1994) when a Brownian motion model of evolution is assumed. Note that the rate parameter returned here is the per-step rate per unit of branch length in the tree, not the squared rate (the latter also referred to as the variance rate or σ^2). The rate is returned in the first matrix of the results.

The ancestral node values are reconstructed using the generalized linear model (also known as phylogenetic generalized least squares, or PGLS) described by Martins and Hansen (1997; see also Rohlf, 2001). This method assumes a Brownian motion model of evolution and provides the same estimates as squared-change parsimony (Maddison, 1991) and maximum-likelihood (Schluter *et al.*, 1997). The node values are returned in the second matrix of the results.

The standard deviation of each node describes the uncertainty due to the evolutionary process, which in this case is assumed to be Brownian motion. The node values are distributed normally (Felsenstein 1973, 2004). The standard deviations reported by this function are the square roots of the node variances based on the corrected equations given by Rohlf (2001) and calculated by successive re-rooting of the tree (Garland and Ives, 2000). The standard deviations of the nodes are returned in the third matrix of the results.

Note that the rate, the node reconstructions, and the standard deviations of the nodes are all parameters which have uncertainties due to estimation. The uncertainty is a complicated function of the rate, the number of branches on the tree, the topology and lengths of the branches, and the mode of evolution. One can explore the magnitude of the estimation errors by Monte Carlo simulation of data on the tree using the *SimulateContinuousTraitOnTree[]* function that is included in this package.

Arguments:

- *Tree* is a string containing a tree with branch lengths in Newick format or a tree table of the sort produced by TreeToTable[].
- *Trait* is a 2 x *n* matrix with tip labels in the first column and values of a continuous trait in the second column. Tip labels must match the labels used in the tree.

Example:

{rate, nodes, sd} = ReconstructNodes[tree, Y];

RerootTree[*tree, node*]

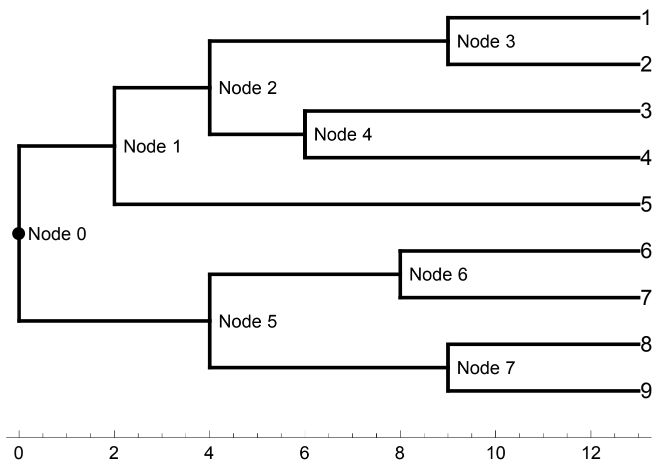
This function reroots a Newick tree at a specified node.

Arguments:

- *tree* is a string containing a tree with branch lengths in Newick format or a tree table of the sort produced by TreeToTable[].
- *node* is the name of the node where the new tree will be rooted. The available names can be displayed by plotting the tree with DrawNewickTree[tree].

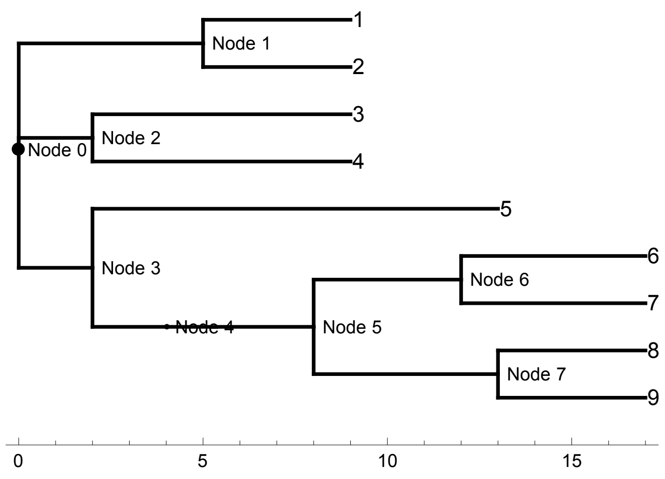
Example:

DrawNewickTree[tree]



newtree = RerootTree[tree, “Node 2”];

DrawNewickTree[newtree]



SimulateContinuousTraitOnTree[*Tree, Rate (,StartValue, IncludeNodes)*]

This function simulates the evolution of a continuous trait on a tree using a Brownian motion model of evolution. By default the trait starts with a value of 0.0 at the root of the tree (this can be changed by specifying a different value as the third input parameter of the function). Evolution is simulated along each branch by Brownian motion using the rate specified by the second input parameter. Note that *Rate* is the per-step rate of change, not the squared per-step rate of change (σ^2 of authors). The function returns a list of node and tip labels followed by the simulated trait value.

Arguments:

- *Tree* is a string containing a tree with branch lengths in Newick format or a tree table of the sort produced by TreeToTable[].
- *Rate* is the amount of trait change per unit of branch length (as specified in the tree)
- *StartValue* is the optional starting value of the trait at the root of the tree (by default the value is 0)
- *IncludeNodes* is the optional parameter that indicates whether simulated node values should be returned (1) or not returned (0, default)

Example:

SimulateContinuousTraitOnTree[*myTable*, 1]

SimulateCorrelatedTraitsOnTree[*Tree, CovarianceMatrix (,StartVector, IncludeNodes)*]

This function simulates the evolution of *k* continuous correlated traits on a tree using a Brownian motion model of evolution. By default each trait starts with a value of 0.0 at the root of the tree (this can be changed by specifying a different value as part of a vector in the third input parameter of the function). The number of traits is determined by the covariance matrix, as is the rate. The step variance rate for each trait (σ^2 of authors) is the variance recorded for that trait in the diagonal of the covariance matrix. The function returns a list of node and tip labels followed by vectors containing the simulated trait values.

Arguments:

- *Tree* is a string containing a tree with branch lengths in Newick format or a tree table of the sort produced by TreeToTable[].
- *CovarianceMatrix* is a square symmetric matrix describing the variance and covariances of the trait. Note that the variances are used as the rate parameters in the simulation.
- *StartVector* is an optional vector containing the starting value of each trait at the root of the tree (by default the value is 0)
- *IncludeNodes* is the optional parameter that indicates whether simulated node values should be returned (1) or not returned (0, default)

Example:

SimulateCorrelatedTraitsOnTree[*myTree*, {{.9, .3},{.3, .5}}]

TableToTree[*table*]

Parses a tree table and returns a Newick format tree in a single string with no spaces or line feeds. The table should be in the same format as the ones returned by the *TreeToTable[]* function.

Arguments:

- *table* has one row for each branch in the tree and four columns: the first column contains the name of the descendant tip or node, the second column contains the name of the ancestor tip or node, the third column contains the length of the branch, and the fourth column indicates whether the branch ends in a tip (1) or a node (2).

Example:

```
myTable = {{"Descendant", "Ancestor", "Branch Length", "Tip?"}, {"1", "Node 0", 1,  1}, {"2", "Node 1", 1, 1}, {"3", "Node 1", 1, 1}, {"Node 1", "Node 0", 1,  0}}

TableToTree[myTable]

(1:1,(2:1,3:1):1);
```

TreeToTable[*tree*]

Parses a Newick format tree, supplied as a single string with no spaces or line feeds (such as the string given by the function ReadNewick[]) and returns a table with one row for each branch in the tree and four columns: the first column contains the name of the descendant tip or node, the second column contains the name of the ancestor tip or node, the third column contains the length of the branch, and the fourth column indicates whether the branch ends in a tip (1) or a node (2). This table (minus its header row) is used by other functions in this package, notably PhylogeneticMatrices[].

Arguments:

- *tree* is a Newick format tree entered as a single string, similar to the format produced by ReadNewick[].

Example:

myTree = "(1:1,(2:1,3:1):1);"

TreeToTable[myTree] //MatrixForm

```
Descendant Ancestor Branch Length Tip?
    1       Node 0        1         1
    2       Node 1        1         1
    3       Node 1        1         1
  Node 1    Node 0        1         0
```

Acknowledgements

Jim Rohlf and Emilia Martins helped interpret some of their equations. Joe Felsenstein gave advice on converting branch lengths to variance units for some of the earliest versions of functions contained in this package. Michelle Lawing, Aida Gómez Robles, Jesualdo Fuentes Gonzales, and members of my G562 course on geometric morphometrics have made suggestions that have improved the accuracy and operation of these functions.

Bibliography

Felsenstein, J. 1973. Maximum-likelihood estimation of evolutionary trees from continuous characters. *American Journal of Human Genetics*, 25: 471-492.

Felsenstein, J. 1985. Phylogenies and the comparative method. *American Naturalist*, 125: 1-15.

Felsenstein, J. 2004. *Inferring Phylogenies*. Sinauer Associates, Sunderland, Mass.

Garland, T. Jr. and A. R. Ives. 2000. Using the past to predict the present: confidence intervals for regression equations in phylogenetic comparative methods. *American Naturalist*, 155: 346-364.

Maddison, W. P. 1991. Squared-change parsimony reconstructions of ancestral states for continuous-valued characters on a phylogenetic tree. *Systematic Zoology*, 40: 304—314.

Martins, E. P. 1994. Estimating the rate of phenotypic evolution from comparative data. *American Naturalist*, 144: 193-209.

Martins, E. P. and T. F. Hansen. 1997. Phylogenies and the comparative method: a general approach to incorporating phylogenetic information into the analysis of interspecific data. *American Naturalist*, 149: 646-667.

Rohlf, F. J. 2001. Comparative methods for the analysis of continuous variables: geometric interpretations. *Evolution*, 55: 2143-2160.

Schluter, D., T. Price. A. O. Mooers, and D. Ludwig. Likelihood of ancestor states in adaptive radiation. *Evolution*, 51: 1699-1711.