Package ‘phytools’

July 10, 2015

Version  0.4-60
Date     2015-7-09
Title    Phylogenetic Tools for Comparative Biology (and Other Things)
Author   Liam J. Revell
Maintainer Liam J. Revell <liam.revell@umb.edu>
Depends  R (>= 2.10), ape (>= 3.0-10), maps
Imports  animation, clusterGeneration, graphics, grDevices, methods,
         mnormt, msm, numDeriv, phangorn (>= 1.6-3), plotrix,
         scatterplot3d, stats, utils
Suggests geiger, rgl
ZipData  no

Description  Package contains various functions for phylogenetic analysis.
              This functionality is concentrated in the phylogenetic analysis of
              comparative data from species. For example, the package includes
              functions for Bayesian and ML ancestral state estimation; visual
              simulation of trait evolution; fitting models of trait evolution
              with multiple Brownian rates and correlations; visualizing
              discrete and continuous character evolution using colors or
              projections into trait space; identifying the location of a change
              in the rate of character evolution on the tree; fast Brownian motion
              simulation and simulation under several other models of
              continuous trait evolution; fitting a model of correlated binary
              trait evolution; locating the position of a fossil or an recently
              extinct lineage on a tree using continuous character data with ML;
              plotting lineage accumulation through time, including across
              multiple trees (such as a Bayesian posterior sample); conducting
              an analysis called stochastic character mapping, in which character
              histories for a discrete trait are sampled from their posterior
              probability distribution under a model; conducting a multiple
              (i.e., partial) Mantel test; fitting a phylogenetic regression model
              with error in predictor and response variables; conducting a
              phylogenetic principal components analysis, a phylogenetic
              regression, a reduced major axis regression, a phylogenetic
canonical correlation analysis, and a phylogenetic ANOVA; projecting
a tree onto a geographic map; simulating discrete character
histories on the tree; and fitting a model in which a discrete
color character evolves under the threshold model. In addition to this
phylogenetic comparative method functionality, the package also
contains functions for a wide range of other purposes in
phylogenetic biology. For instance, functionality in this package
includes (but is not restricted to): adding taxa to a tree
(including randomly, everywhere, or automatically to genera);
generating all bi- and multi-furcating trees for a set of taxa;
reducing a phylogeny to its backbone tree; dropping tips or adding
tips to special types of phylogenetic trees; exporting a tree as an
XML file; converting a tree with a mapped character to a tree with
singleton nodes and one character state per edge; estimating a
phylogeny using the least squares method; simulating birth-death
trees under a range of conditions; rerooting trees; a wide range
of visualizations of trees; and a variety of other manipulations
and analyses that phylogenetic biologists may find useful for
their research.

License  GPL (>= 2)
URL  http://www.phytools.org
Repository  CRAN
Date/Publication  2015-07-10 20:33:04
NeedsCompilation  no

R topics documented:

phytools-package ................................................. 5
add.arrow .................................................. 6
add.color.bar ............................................... 7
add.everywhere ........................................... 8
add.random ................................................... 8
add.simmap.legend ......................................... 9
add.species.to.genus ...................................... 10
allFurcTrees ............................................... 11
anc.Bayes .................................................... 12
anc.ML .......................................................... 13
anc.trend ..................................................... 14
ancThresh .................................................... 16
anoletree ...................................................... 17
applyBranchLengths ....................................... 18
ave.rates .................................................... 18
bind.tip ...................................................... 19
bmPlot ........................................................ 20
branching.diffusion ...................................... 22
brownie.lite ............................................... 23
R topics documented:

brownieREML .......................................................... 24
cladelabels .............................................................. 25
collapse.to.star ...................................................... 26
collapseTree ............................................................ 27
contMap ................................................................. 28
countSimmap ........................................................... 29
densityMap .............................................................. 30
describe.simmap ....................................................... 32
di2multi.simmap ...................................................... 33
drop.clade .............................................................. 34
drop.leaves ............................................................. 34
drop.tip.contMap .................................................... 35
drop.tip.simmap ...................................................... 36
estDiversity ........................................................... 37
evol.rate.mcmc ....................................................... 38
evol.vcv ............................................................... 39
evolvcv_lite .......................................................... 41
exhaustiveMP .......................................................... 42
expm ..................................................................... 43
export.as.xml ........................................................ 43
fancyTree .............................................................. 44
fastAnc ................................................................. 46
fastBM ................................................................. 47
fastMrca ............................................................... 48
findMrca ............................................................... 49
fitBayes ............................................................... 50
fitDiversityModel ................................................... 51
fitPagel ............................................................... 53
gammatest ............................................................ 54
genSeq ................................................................. 55
getCladesofSize ..................................................... 56
getDescendants ...................................................... 56
getExtant ............................................................. 57
getSisters ............................................................ 58
getStates ............................................................. 59
ladderize.simmap ................................................... 59
lambda.transform .................................................... 60
likMlambda ............................................................ 61
locate.fossil ........................................................ 62
locate.yeti ........................................................... 63
ls.tree ................................................................. 64
lt ........................................................................ 64
ltt95 ................................................................. 66
make.era.map ........................................................ 67
make.simmap ........................................................ 68
map.overlap ........................................................ 70
map.to.singleton .................................................... 71
matchNodes .......................................................... 72
mergeMappedStates .......................................................... 73
midpoint.root ............................................................... 74
minRotate ................................................................. 75
minSplit ................................................................. 75
mrp.supertree .............................................................. 76
multi.mantel ............................................................. 77
multiC ................................................................. 78
multiRF ................................................................. 79
nodeHeights ............................................................... 80
optim.phylo.ls ........................................................... 81
orderMappedEdge .......................................................... 82
paintSubTree ............................................................. 83
paste.tree ............................................................... 84
pbtree ................................................................. 85
pgls.Ives ................................................................. 87
phenogram ................................................................. 88
phyl.cca ................................................................. 90
phyl.pairedttest .......................................................... 91
phyl.pca ................................................................. 93
phyl.resid ............................................................... 94
phyl.RMA ............................................................... 95
phyl.vcv ................................................................. 96
phyLANOVA ............................................................... 97
phylo.to.map ............................................................ 98
phylo.toBackbone ........................................................ 99
phyloDesign ............................................................ 100
phylomorphospace ..................................................... 101
phylomorphospace3d .................................................. 102
phylosig ................................................................. 104
plot.backbonePhylo ..................................................... 105
plotBranchbyTrait ........................................................ 106
plotSimmap ............................................................. 108
plotThresh ............................................................... 110
plotTree ................................................................. 111
plotTree.wBars .......................................................... 112
posterior.evolrate ...................................................... 113
print.backbonePhylo ..................................................... 114
ratebystate .............................................................. 114
rateshift ................................................................. 116
read.newick ............................................................. 117
read.simmap ............................................................. 118
reorder.backbonePhylo .................................................. 119
reorderSimmap .......................................................... 120
rep.phylo ............................................................... 120
reroot ................................................................. 121
rerootingMethod ......................................................... 122
rescaleSimmap ........................................................... 124
rotateNodes .............................................................. 125
phytools-package

Index

phytools-package  phytools: Phylogenetic Tools for comparative biology (and other things)

Description

phytools provides functions for phylogenetic comparative biology: as well as several other functions for tree inference, manipulation, and analysis that are not implemented in other R packages.

The complete list of functions can be displayed with library(help = phytools).

More information on phytools can be found at http://www.phytools.org or http://blog.phytools.org.

Author(s)

Liam J. Revell

Maintainer: Liam J. Revell <liam.revell@umb.edu>

References

add.arrow

Add an arrow pointing to a tip or node on the tree

Description

This function adds an arrow to a plotted tree.

Usage

add.arrow(tree=NULL, tip, ...)

Arguments

tree

an object of class "phylo", "contMap", or "densityMap". If not supplied, the
function will obtain the last plotted phylogeny from the environmental variable
last_plot.phylo.

tip

label of tip or tip or node number. If tree=NULL then the tip or node number
must be supplied.

...

optional arguments to control the shape and size of the arrow including: its
length (arrl) in the units of the plot; the length of the arrowhead (hedl); the
total angle between the wings in the arrowhead (angle); the line width for the
plotted lines (lwd); the offset from the tip or end of tip label, in character widths
(offset); and the color (col).

Details

This function presently works for radial (type="fan") and right facing square phylograms (type="phylogram").
Trees can be plotted using phytools function plotTree, plotSimmap, contMap, densityMap, and
ape method plot.phylo.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

nodelabels
add.color.bar

Add color bar to a plot

Description

This function adds a color bar to a plot created by plotBranchbyTrait. A color bar can be added by clicking on a location within the plot (when prompt=TRUE) or by setting prompt=FALSE and supplying x & y coordinates for the object. This function is also used internally by plot.contMap and plot.densityMap.

Usage

add.color.bar(leg, cols, title=NULL, lims=c(0,1), digits=1, prompt=TRUE, lwd=4, outline=TRUE, ...)

Arguments

- **leg**: numerical value for the length of the legend.
- **cols**: colors for the legend.
- **title**: text to plot above the bar.
- **lims**: range for the bar.
- **digits**: digits for plotted numbers.
- **prompt**: logical value indicating whether the location of the legend should be obtained interactively.
- **lwd**: width of the plotted bar.
- **outline**: logical value indicated whether or not to outline the plotted color bar with a 1 pt line.
- **...**: optional arguments including: x x-coordinate of the legend (if prompt=FALSE); y y-coordinate of the legend; subtitle optional legend subtitle.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

ccontMap, densityMap, plotBranchbyTrait
add.everywhere  Add tip to all edges in a tree

Description
This function adds a tip to all branches to the tree and returns a list of trees as an object of class "multiPhylo".

Usage
add.everywhere(tree, tip.name)

Arguments
  tree    an object of class "phylo".
  tip.name a string containing the name of the tip to add.

Value
A list of trees as an object of class "multiPhylo". Since the tip can be added to any branch, the length of the list is equal to the number of edges in tree.

Author(s)
Liam Revell <liam.revell@umb.edu>

References

See Also
allFurcTrees, exhaustiveMP

add.random  Add tips at random to the tree

Description
This function adds new tips at random to a tree with branch lengths. If no edge lengths are provided, and the tree is ultrametric, then edge lengths are assigned to keep the tree ultrametric. The probability that a new tip is added along any branch is directly proportional to the length of the branch.
Usage

add.simmap.legend(leg=NULL, colors, prompt=TRUE, vertical=TRUE, ...)

Arguments

tree an object of class "phylo".

n a number of tips to add to the tree. If NULL, will use length(tips).

tips a set of tip names for the added tips. If NULL, names will be supplied

eqedge length terminal edge length for the added tips. If NULL, and is.ultrametric(tree)==TRUE, then edge lengths will be assigned to keep the tree ultrametric. Note that if edge lengths are assigned and n>1, then the assigned terminal edge lengths are not guaranteed as subsequent random tip addition could occur along the new terminal edge.

order addition order for the new tips.

Details

Note that sometimes the resultant tree plotted with plot.phylo or plotSimmap may display with branches crossing. If so, the tree can be ‘untangled’ using untangle.

Value

An object of class "phylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

allFurcTrees, add.everywhere

Description

This function adds a legend (by default, interactively) to a plotted stochastic character mapped tree.
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>leg</td>
<td>states for the discrete character in the order of colors.</td>
</tr>
<tr>
<td>colors</td>
<td>colors for the legend in the order of leg, or, if leg=NULL, named vector of colors in which names(colors) are the states of the mapped discrete character.</td>
</tr>
<tr>
<td>prompt</td>
<td>logical value indicating whether the location of the legend should be obtained interactively (i.e., by clicking in the plotting area).</td>
</tr>
<tr>
<td>vertical</td>
<td>logical value indicating whether to plot the legend vertically (if TRUE) or horizontally.</td>
</tr>
<tr>
<td>...</td>
<td>optional arguments including: x x-coordinate of the legend (if prompt=FALSE); y y-coordinate of the legend; and shape which can be shape=&quot;square&quot;, the default, or shape=&quot;circle&quot;.</td>
</tr>
</tbody>
</table>

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

plotSimmap

add.species.to.genus  Add species to genus on a phylogeny

Description

This function adds an additional species to a genus on a phylogeny.

Usage

add.species.to.genus(tree, species, genus==NULL, where=c("root","random"))

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>an object of class &quot;phylo&quot;.</td>
</tr>
<tr>
<td>species</td>
<td>string contain species name in the format &quot;Genus_species&quot; or &quot;Genus species&quot;.</td>
</tr>
<tr>
<td>genus</td>
<td>optional argument containing the genus to which species is to be attached. If NULL then genus will be extracted from species.</td>
</tr>
<tr>
<td>where</td>
<td>location to attach species to the tree. where=&quot;root&quot; will cause the species to be attached to the MRCA of all members of genus. where=&quot;random&quot; will cause species to be attached at random to the subtree descended from the MRCA of all members of genus.</td>
</tr>
</tbody>
</table>
allFurcTrees

Details
If genus contains only one species and `where="root"`, then species will be attached midway along the branch leading to the one species. If `where="random"` then species will be added at a random position along the edge. If genus cannot be found in the tree, then the original tree is returned and a warning printed. If the tree is not ultrametric, then the resultant tree may not contain branch lengths and a warning will be printed. If genus is non-monophyletic then species will be attached to the most inclusive group containing members of genus and a warning will be printed.

Value
An object of class "phylo".

Author(s)
Liam Revell <liam.revell@umb.edu>

References

See Also
add.random, bind.tip

allFurcTrees Generate all bi- and multifurcating unrooted trees

Description
This function creates all possible unrooted bi- and multifurcating trees and returns a list of trees as an object of class "multiPhylo".

Usage
allFurcTrees(n, tip.label=NULL, to.plot=TRUE)

Arguments
n an integer giving the desired number of species.
tip.label an optional vector of length n containing the tip names.
to.plot an optional logical value indicating whether or not to plot the trees.

Details
This function should be used with caution for n greater than about 8, as in this case the number of possible trees is extremely large.
anc.Bayes

Description

This function uses Bayesian MCMC to sample from the posterior distribution for the states at internal nodes in the tree.

Usage

anc.Bayes(tree, x, ngen=10000, control=list())

Arguments

tree an object of class "phylo".
x a vector of tip values for species; names(x) should be the species names.
nngen a integer indicating the number of generations for the MCMC.
control a list of control parameters containing the following elements: sig2: starting value for $\sigma^2$ (BM rate); a: starting for the state at the root node; y: starting values for the states at all internal nodes excluding the root (should be labeled with node numbers); pr.mean: means for the prior distributions in the following order - sig2, a, y, note that the prior probability distribution is exponential for sig2 and normal for a and y; pr.var: variances on the prior distributions, same order as pr.mean (but the variance is not used for sig2); prop: variances on the normal proposal distributions in the same order as pr.mean; sample: sample frequency from the MCMC.

References


See Also

add.everywhere, exhaustiveMP
### Value

A matrix with number of rows ngen/sample+1 containing the posterior sample and likelihoods. Matrix columns are labeled either sig2 or by the node number of the internal node.

### Author(s)

Liam Revell <liam.revell@umb.edu>

### References


### See Also

ace, anc.ML, anc.trend, evol.rate.mcmc, fastAnc

### Examples

```r
# simulate using fastBM
X<--anc.Bayes(tree,x,ngen=10000) # sample ancestral states
estimates<-colMeans(X[21:nrow(X),]) # get estimates, excluding burnin
```

---

### Description

This function estimates the evolutionary parameters and ancestral states for Brownian evolution using likelihood. It is also possible (for model="BM") to allow for missing data for some tip taxa.

### Usage

```r
anc.ML(tree, x, maxit=2000, model=c("BM","OU"), ...)
```

### Arguments

- **tree**: an object of class "phylo".
- **x**: a vector of tip values for species; names(x) should be the species names.
- **maxit**: an optional integer value indicating the maximum number of iterations for optimization.
- **model**: model of continuous character evolution on the tree. It’s possible that only model="BM" works in the present version as model="OU" has not be thoroughly tested & some bugs were reported for an earlier version.
- **...**: other arguments.
Details

Because this function relies on a high dimensional numerical optimization of the likelihood function, `fastAnc` should probably be preferred for most purposes. If using `anc.ML`, users should be cautious to ensure convergence. This has been ameliorated in phytools>=0.2-48 by seeding the ML optimization with the result from `fastAnc`.

Value

A list with the following components:

- `sig2` the variance of the BM process.
- `ace` a vector with the ancestral states.
- `logLik` the log-likelihood.
- `convergence` the value of $\text{convergence}$ returned by `optim()` (0 is good).

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

`ace, anc.Bayes, fastAnc, optim`

Examples

```r
ri<-pbtree(n=50)
x<-fastBM(ri) # simulate using fastBM
anc.ML(ri,x) # fit model & estimate ancestral states
```

Description

This function estimates the evolutionary parameters and ancestral states for Brownian evolution with directional trend.

Usage

```r
anc.trend(tree, x, maxit=2000)
```
**anc.trend**

**Arguments**

- `tree` an object of class "phylo".
- `x` a vector of tip values for species; `names(x)` should be the species names.
- `maxit` an optional integer value indicating the maximum number of iterations for optimization.

**Details**

Note that this will generally only work and produce sensible results for a phylogeny with some non-contemporary tips (i.e., a tree with some fossil species). The function uses `optim` with `method="L-BFGS-B"`; however optimization is only constrained for the `sig2` which must be >0.

**Value**

A list with the following components:

- `ace` a vector with the ancestral states.
- `mu` a trend parameter per unit time.
- `sig2` the variance of the BM process.
- `logL` the log-likelihood.
- `convergence` the value of $convergence$ returned by `optim()` (0 is good).

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

`ace, anc.Bayes, anc.ML, optim`

**Examples**

```r
  tree<-rtree(20)
  x<-fastBM(tree, mu=2) # simulate using fastBM with a trend (m!=0)
  anc.trend(tree,x) # fit model & estimate ancestral states
```
ancThresh  

*Ancestral character estimation under the threshold model using Bayesian MCMC*

**Description**

This function uses Bayesian MCMC to estimate ancestral states and thresholds for a discrete character under the threshold model from quantitative genetics (Felsenstein 2012).

**Usage**

```r
ancThresh(tree, x, ngen=1000, sequence=NULL, method="mcmc", model=c("BM","OU","lambda"), control=list(), ...)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>phylogenetic tree.</td>
</tr>
<tr>
<td>x</td>
<td>a named vector containing discrete character states; or a matrix containing the tip species, in rows, and probabilities of being in each state, in columns.</td>
</tr>
<tr>
<td>ngen</td>
<td>number of generations to run the MCMC.</td>
</tr>
<tr>
<td>sequence</td>
<td>assumed ordering of the discrete character state. If not supplied and x is a vector then numerical/alphabetical order is assumed; if not supplied and x is a matrix, then the column order of x is used.</td>
</tr>
<tr>
<td>method</td>
<td>only method currently available is &quot;mcmc&quot;.</td>
</tr>
<tr>
<td>model</td>
<td>model for the evolution of the liability. Options are &quot;BM&quot; (Brownian motion, the default), &quot;OU&quot; (Ornstein-Uhlenbeck), or &quot;lambda&quot; (the lambda model).</td>
</tr>
<tr>
<td>control</td>
<td>list containing the following elements: sample, the sampling interval; propliab variance of the proposal distribution for liabilities; propthresh variance on the proposal distribution for the thresholds; propalpha variance on the proposal distribution for alpha (for model=&quot;OU&quot;); pr.anc prior probability distribution on the ancestral states for each node, in a matrix - not all nodes need to be supplied; pr.th prior density on the thresholds; burnin number of generations to exclude for burn-in when plotting posterior probabilities on the tree; plot logical value indicating whether or not to plot the posterior probabilities; print logical value indicating whether or not to print the state of the MCMC; piecol colors for the posterior probabilities plotted as pie charts at internal nodes; and tipcol which indicates whether the tip colors should be based on the input data (&quot;input&quot;) or sampled tip liabilities (&quot;estimated&quot;). These will only differ if there is uncertainty in the tip states.</td>
</tr>
<tr>
<td></td>
<td>additional arguments to be passed to plotThresh (called internally).</td>
</tr>
</tbody>
</table>

...
**anoletree**

### Value

This function returns a list with four elements:

- `ace` posterior probabilities for each character state at each internal node, with the burn-in excluded.
- `mcmc` full posterior sample for the states.
- `par` full posterior sample for the thresholds, the alpha parameter of the "OU" model (if applicable), and the likelihood.
- `liab` full posterior sample of the liabilities at internal and tip nodes.

### Author(s)

Liam Revell <liam.revell@umb.edu>

### References


### See Also

- `ancBayes`, `threshBayes`

---

**anoletree**

*Phylogeny of Greater Antillean anole ecomorph species with mapped discrete character*

### Description

A phylogeny of Greater Antillean anole species with a mapped discrete character - 'ecomorph class.' Data and tree are from Mahler et al. (2010).

### Usage

```r
data(anoletree)
```

### Format

The data are stored as a modified object of class "phylo" with a mapped discrete character. (E.g., see `read.simmap`.)

### Source

**applyBranchLengths**  
*Applies the branch lengths of a reference tree to a target*

**Description**

This function applies the set of branch lengths from a reference tree to a target tree while reconciling any mappings (as in `read.simmap`) with the new branch lengths.

**Usage**

```
applyBranchLengths(tree, edge.length)
```

**Arguments**

- `tree`: target tree.
- `edge.length`: number of digits for rounding. Passed to `round`.

**Value**

A tree with branch lengths, or modified "phylo" object with a mapped discrete character.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


---

**ave.rates**  
*Average the posterior rates*

**Description**

Primarily used internally by `posterior.evolrate`.

**Usage**

```
ave.rates(tree, shift, tips, sig1, sig2, ave.shift, showTree=TRUE)
```
bind.tip

Arguments

- **tree**: a tree.
- **shift**: the shift point for this sample.
- **tips**: tip names tipward of shift.
- **sig1**: rate 1.
- **sig2**: rate 2.
- **ave.shift**: average shift from all samples.
- **showTree**: logical value indicating whether to plot the rate-stretched tree.

Value

A list of the rates.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

- `evol.rate.mcmc`, `minSplit`, `posterior.evolrate`

Description

Functions adds a new tip to the tree. If the tree is ultrametric and no branch length is specified, then `edge.length` is scaled so that the tree remains ultrametric after the new tip is added.

Usage

`bind.tip(tree, tip.label, edge.length=NULL, where=NULL, position=0)`
**Arguments**

- **tree**: receptor tree.
- **tip.label**: a string containing the species name for the new tip.
- **edge.length**: edge length for the new tip (a scalar).
- **where**: node number to attach new tip. If position>0 then the tip will be attached below the specified node. Node numbers can also be tips, in which case the new tip will be added along the terminal edge. To find out the tip number for given species with name "species" type: which(tree$tip.label=="species").
- **position**: distance below node to add tip.

**Details**

Wrapper function for 'ape' bind.tree.

**Value**

An object of class "phylo".

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


---

**bmPlot**

_Simulates and visualizes discrete-time Brownian evolution on a phylogeny_

**Description**

This function conducts discrete-time Brownian motion simulation on an input tree, plots the outcome, and returns the tip and internal node states to the user as a named vector. The function will first rescale and round the branch lengths to integer length, if they are not already in integer values. If integer branch lengths are provided, the user should also set ngen=max(nodeHeights(tree)). For type="threshold" the visualization is of the threshold model (Felsenstein 2012), in which the evolving character is liability and the segments of evolution are colored by their value for the threshold trait. If type="threshold" is used, the function requires at least one additional input: thresholds, a vector containing the ordered thresholds between states. The user can also provide the colors for plotting in colors. Note that one more color than threshold should be provided as one threshold implies two states; two thresholds, three states; etc. If no value for colors is provided, the function will recycle a set of four colors up to the number of times required by thresholds. Finally, the optional argument return.tree=TRUE will tell the function to return a list with the tip and note states and an object of class "phylo" with (for type="threshold"), the state for the threshold model through time mapped on the branches of the tree in discrete time.
Usage

```
bmPlot(tree, type="BM", anc=0, sig2=1/1000, ngen=1000, ...)  
```

Arguments

- `tree`: a phylogenetic tree in "phylo" format.
- `type`: the type of plot to create. See Description.
- `anc`: the ancestral value for the root node.
- `sig2`: the BM rate (variance of the Brownian evolution process).
- `ngen`: number of generations for the simulation: will rescale the tree to this total length.
- `...`: arguments to be passed to different methods.

Value

This function conducts and plots discrete time Brownian simulation and returns a vector containing the simulated states at internal nodes and tips of the tree. It also returns, by default (although this can be turned off) a tree with the branch lengths in discrete time and with a mapped discrete character (for type="threshold").

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

`fastBM`, `pbtree`, `phenogram`, `threshBayes`

Examples

```
# plot BM simulation on 20 taxon tree
tree<-pbtree(n=20)
x<-bmPlot(tree)

# plot simulation of a threshold character
tree<-pbtree(n=20)
x<-bmPlot(tree,type="threshold",thresholds=c(0,1,2))
```
branching.diffusion  

*Animation of branching random diffusion*

**Description**

This function creates an animation of branching random diffusion (i.e., BM with speciation).

**Usage**

```r
branching.diffusion(sig2=1, b=0.0023, time.stop=1000, ylim=NULL,
smooth=TRUE, pause=0.02, record=NULL, path=NULL)
```

**Arguments**

- `sig2` variance of BM process.
- `b` birthrate for branching process.
- `time.stop` number of generations to run.
- `ylim` y limits (for plotting).
- `smooth` no longer used.
- `pause` pause (in s) between generations.
- `record` filename for video file output (no video if NULL).
- `path` full path to file for video rendering (by default is C:/Program Files/ffmpeg/bin/ffmpeg.exe).

**Value**

An animated plot and (optionally) a recorded video file. For animation to be recorded to file, the function requires the package "animation" as well as a video renderer.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

`bmPlot`, `fastBM`
**Description**

This function takes a modified "phylo" object with a mapped binary or multistate trait (see `read.simmap`) and data for a single continuously valued character. It then fits the Brownian rate variation ("non-censored") model of O'Meara et al. (2006; *Evolution*). This is also the basic model implemented in Brian O'Meara's "Brownie" program.

**Usage**

```r
brownie.lite(tree, x, maxit=2000, test="chisq", nsim=100, se=NULL, ...)
```

**Arguments**

- `tree` a phylogenetic tree in modified "phylo" format (see `read.simmap`, `make.simmap`, or `paintSubTree`).
- `x` a vector of tip values for species; `names(x)` should be the species names.
- `maxit` an optional integer value indicating the maximum number of iterations for optimization - may need to be increased for large trees.
- `test` an optional string indicating the method for hypothesis testing - options are "chisq" or "simulation".
- `nsim` number of simulations (only used if `test="simulation"`).
- `se` a vector containing the standard errors for each estimated mean in `x`.
- `...` optional arguments.

**Details**

Sampling error in the estimation of species means can also be accounted for by assigning the vector `se` with the species specific sampling errors for `x`.

**Value**

A list with the following components:

- `sig2.single` is the rate for a single rate model - this is usually the "null" model.
- `a.single` is the estimated state at the root node for the single rate model.
- `var.single` variance on the single rate estimator - obtained from the Hessian.
- `logL1` log-likelihood of the single-rate model.
- `k1` number of parameters in the single rate model (always 2).
- `sig2.multiple` is a length p (for p rates) vector of BM rates from the multi-rate model.
- `a.multiple` is the estimated state at the root node for the multi-rate model.
brownieREML

var.multiple $p \times p$ variance-covariance matrix for the $p$ rates - the square-roots of the diagonals should give the standard error for each rate.

logL.multiple log-likelihood of the multi-rate model.

k2 number of parameters in the multi-rate model ($p+1$).

P.chisq P-value for a likelihood ratio test against the $\chi^2$ distribution; or

P.sim P-value for a likelihood ratio test against a simulated null distribution.

convergence logical value indicating if the likelihood optimization converged.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

brownieREML, evol.vcv

---

**brownieREML**  
*REML version of brownie.lite*

**Description**

This function takes a modified "phylo" object with a mapped binary or multistate trait (see `read.simmap`) and data for a single continuously valued character. It then uses restricted maximum likelihood (REML) to fit the Brownian rate variation ("noncensored") model of O’Meara et al. (2006; *Evolution*). This function is similar to `brownie.lite` but uses REML (which is faster and unbiased) instead of ML. REML optimization takes advantage of Felsenstein’s (1985) contrasts algorithm.

**Usage**

brownieREML(tree, x, maxit=2000)

**Arguments**

- **tree** a phylogenetic tree in modified "phylo" format (see `read.simmap` and `make.simmap`).
- **x** a vector of tip values for species; names(x) should be the species names.
- **maxit** an optional integer value indicating the maximum number of iterations for optimization - may need to be increased for large trees.
cladelabels

Value

A list with the following components:

- `sig2.single` is the rate for a single rate model - this is usually the “null” model.
- `logL1` log-likelihood of the single-rate model.
- `sig2.multiple` is a length $p$ (for $p$ rates) vector of BM rates from the multi-rate model.
- `logL2` log-likelihood of the multi-rate model.
- `convergence` numerical value from `optim`.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

- `brownie.lite`
- `evol.vcv`
- `evol.rate.mcmc`

---

cladelabels

Add labels to subtrees of a plotted phylogeny

Description

This function adds clade labels to a plotted tree.

Usage

`cladelabels(tree=NULL, text, node, offset=NULL, wing.length=NULL, cex=1)`

Arguments

- `tree` an object of class “phylo”. If not supplied, the function will obtain the last plotted phylogeny from the environmental variable `last_plot.phylo`.
- `text` desired clade label text.
- `node` node number for the most recent common ancestor of members of the clade.
- `offset` offset (as a multiplier of character width) for the label. Defaults to offset=1 if tree is supplied or offset=8 otherwise.
- `wing.length` length of the wings to add to the top & bottom of the label bar (in character widths).
- `cex` character expansion factor.
Details

This function presently works only for rightward facing plotted phylogenies - but no warning will be returned if your tree does not conform to this requirement!

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

nodelabels

collapse.to.star Collapse a subtree to a star phylogeny

Description

This function collapses a subtree to a star, keeping the tips at the same height above the root as in the original tree.

Usage

collapse.to.star(tree, node)

Arguments

tree an object of class "phylo".
node node for the clade to be collapsed.

Value

An object of class "phylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References

collapseTree

See Also

splitTree, starTree

collapseTree  Interactive tree visualizer

Description

Function creates an interactive visualization of collapsing & expanding clades on the tree.

Usage

collapseTree(tree, ...)

Arguments

tree  an object of class "phylo".

...  optional arguments. These mostly match the arguments of plotSimmap.

Details

Function first plots a fan style tree, and then the user collapses node on the tree by clicking on them. Collapsed nodes are collapsed to the common ancestor of the clade. Nodes that have been collapsed can also be expanded by clicking.

Value

Returns the final plotted tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

plotTree, plotSimmap
contMap

Map continuous trait evolution on the tree

Description

Function plots a tree with a mapped continuous character. The mapping is accomplished by estimating states at internal nodes using ML with fastAnc, and the interpolating the states along each edge using equation [2] of Felsenstein (1985).

Usage

contMap(tree, x, res=100, fsize=NULL, ftype=NULL, lwd=4, legend=NULL, lims=NULL, outline=TRUE, sig=3, type="phylogram", direction="rightwards", plot=TRUE, ...)

# S3 method for class 'contMap'
plot(x, ...)

Arguments

tree

object of class "phylo".

x

vector of phenotypic trait values for species. names(x) should contain the species names and match tree$tip.label. Or, for plot.contMap, an object of class "contMap".

res

resolution for gradient plotting. Larger numbers indicate a finer (smoother) gradient.

fsize

relative font size - can be a vector with the second element giving the font size for the legend.

ftype

font type - see options in plotSimmap. As with fsize, this can be a vector with the second element giving font type for the legend.

lwd

line width for branches.

legend

if FALSE no legend is plotted; if a numeric value, it gives the length of the legend in units of branch length. Default is 0.5 times the total tree length.

lims

range for the color map. By default, this will be c(min(x),max(x)), and should always include this range.

outline

logical value indicating whether or not to outline the branches of the tree in black.

sig

the number of decimal places to show on the legend limits.

type

type of plot desired. Options are "phylogram" for a rightward square phylogram; and "fan" for a circular phylogram.

direction

plotting direction for type="phylogram".

plot

logical value indicating whether or not to plot the tree. If plot=FALSE then an object of class "contMap" will be returned without plotting.
countSimmap

... optional arguments for plot.contMap which include all the arguments of contMap except for tree, x, res, and lims. Also method, "fastAnc" or "anc.ML", specifying which function to use for ancestral state estimation; and hold specifies whether or not to hold output to graphical device before plotting (defaults to hold=TRUE).

Value

Plots a tree. An object of class "contMap" is returned invisibly.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

anc.ML, densityMap, fastAnc, plotSimmap

countSimmap  Counts the number of character changes on a SIMMAP style tree or set of trees

Description

This function takes a tree or a set of trees with a mapped discrete character (SIMMAP style, e.g., see make.simmap or read.simmap), and computes the total number of character changes as well as the number of character changes between all states.

Usage

countSimmap(tree, states=NULL, message=TRUE)

Arguments

tree  a single tree or a set of trees with a mapped discrete character (e.g. see make.simmap or read.simmap.

states optional argument with the states for the mapped character. If not provided, these will be computed from the tree. This is useful if averaging across many trees, some of which may lack certain states.

message optional logical argument indicating whether or not to return an informative message about the function output.
Value

A list with up to three elements: N is an integer value giving the total number of character changes on the tree; Tr gives the number of transitions between row and column states (or a matrix containing both N and the transitions between states, in rows, for an object of class "multiPhylo"); and (optionally) message contains an explanatory message about the function output.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


Examples

tree<-pbtree(n=100, scale=1)
Q<-matrix(c(-2,1,1,-2,1,1,1,-2),3,3)
colnames(Q)<-rownames(Q)<-c("A","B","C")
mtree<-sim.history(tree,Q)
countSimmap(mtree, states=rownames(Q))

densityMap  Plot posterior density of stochastic mapping on a tree

Description

Function plots a tree with the posterior density for a mapped character from stochastic character mapping on the tree. Since the mapped value is the probability of being in state "1", only binary [0,1] characters are allowed.

Usage

densityMap(trees, res=100, fsize=NULL, ftype=NULL, lwd=3, check=FALSE, legend=NULL, outline=FALSE, type="phylogram", direction="rightwards", plot=TRUE, ...)
## S3 method for class 'densityMap'
plot(x, ...)

Arguments

trees  set of phylogenetic trees in a modified "multiPhylo" object. Values for a two-state discrete character are mapped on the tree. See make.simmap and read.simmap for details.
res  resolution for gradient plotting. Larger numbers indicate a finer (smoother) gradient.
**densityMap**

- **fsize**: relative font size - can be a vector with the second element giving the font size for the legend.
- **ftype**: font type - see options in `plotSimmap`. As with `fsize`, can be a vector with the second element giving font type for the legend.
- **lwd**: line width for branches.
- **check**: check to make sure that the topology and branch lengths of all phylogenies in trees are equal.
- **legend**: if FALSE no legend is plotted; if a numeric value, it gives the length of the legend in units of branch length. Default is 0.5 times the total tree length.
- **outline**: logical value indicating whether or not to outline the branches of the tree in black.
- **type**: type of plot desired. Options are "phylogram" for a rightward square phylogram; and "fan" for a circular phylogram.
- **plot**: logical value indicating whether or not to plot the tree. If plot=FALSE then an object of class "densityMap" will be returned without plotting.
- **direction**: plotting direction for type="phylogram".
- **x**: for `plot.densityMap`, an object of class "densityMap".
- **...**: optional arguments for `plot.densityMap`. These include all the arguments of `densityMap` except `trees` and `res`. Additional optional arguments include `mar` (margins), `offset` (tip label offset), and `hold` (whether or not to use `dev.hold` to hold output to graphical device before plotting; defaults to `hold=TRUE`).

**Value**

Plots a tree and returns an object of class "densityMap" invisibly.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

`make.simmap`, `plotSimmap`, `read.simmap`
describe.simmap | Summarizes a stochastic mapped tree or set of trees

Description

This function summarizes the result of one or more stochastic maps.

Usage

describe.simmap(tree, ...)

Arguments

tree | a single tree or a set of trees with a mapped discrete character (e.g. see make.simmap or read.simmap).

... | optional arguments which include: plot, a logical value indicating whether or not to plot the posterior probabilities at nodes (default is plot=FALSE); check.equal, a logical value indicating whether or not to check if all trees are equal using all.equal.phylo (default is check.equal=FALSE); and message, a logical indicating whether or not to print an informative message to the screen (default is message=TRUE).

Value

A list with the following elements is returned (invisibly if message=TRUE):

count | a matrix containing the number and types of transitions for each tree, if class(tree)="multiPhylo".

times | a matrix containing the times spent in each state on each tree.

ace | the posterior probabilities of each node being in each state, if class(tree)="multiPhylo".

legend | a vector containing the plot legend, if plot=TRUE.

if class(tree)="phylo" then the function simply returns the results of countSimmap combined with the states at each node of the tree and a matrix containing the total and relative times spent in each state on the tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References

**Description**

This function collapses branches of zero length (or, more specifically, branches with length shorter than `tol`) to create a polytomy in a stochastic-map style tree.

**Usage**

```r
di2multi.simmap(tree, tol=1e-08)
```

**Arguments**

- `tree` modified object of class "phylo" containing a stochastically mapped discrete character.
- `tol` length below which edges should be treated as having zero length.

**Details**

This function should theoretically perform similarly to `di2multi` in ape.

**Value**

A tree with a stochastically mapped discrete character

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

`di2multi`, `make.simmap`, `read.simmap`
drop.clade    

*Drop a clade from a tree*

**Description**

Mostly internal function for `posterior.evolrate`; function drops the clade containing the species in `tip`.

**Usage**

```r
drop.clade(tree, tip)
```

**Arguments**

- `tree`: object of class "phylo".
- `tip`: set of tips in a clade.

**Details**

Probably should not use unless you know what you're doing.

**Value**

An object of class "phylo".

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


---

drop.leaves    

*Drop all the leaves (tips) from a tree*

**Description**

Drops all the leaves from a tree, leaving behind only the structure leading to internal nodes.

**Usage**

```r
drop.leaves(tree, ...)
```
drop.tip.contMap

Arguments

- `tree` object of class "phylo".
- `...` optional arguments. Presently includes only the logical value `keep.tip.labels` which tells the function how to label the tips on the reduced tree.

Value

An object of class "phylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


drop.tip.contMap

Description

This function drops one or multiple tips from an object of class "contMap" or "densityMap". This function is equivalent to `drop.tip` but for an object of this class.

Usage

drop.tip.contMap(x, tip)
drop.tip.densityMap(x, tip)

Arguments

- `x` an object of class "contMap" or "densityMap".
- `tip` name or names of species to be dropped.

Details

For more information about objects of class "contMap" or "densityMap", please refer to the documentation pages for `contMap` or `densityMap`, respectively.

Value

An object of class "contMap" or "densityMap".

Author(s)

Liam Revell <liam.revell@umb.edu>
References


See Also

cntMap, densityMap, drop.tip, drop.tip.simmap

---

drop.tip.simmap

Drop tips or extract clade from tree with mapped discrete character

Description

This function drops one or multiple tips from the modified "phylo" object with a mapped binary or multistate trait (see read.simmap) while maintaining the matrix $mapped.edge and list of mappings by branch maps. This function is equivalent to drop.tip but for a tree with a mapped discrete character.

extract.clade.simmap is functionally equivalent to extract.clade but preserves discrete character mappings on the tree.

Usage

drop.tip.simmap(tree, tip)
drop.tip.simmap(tree, node)

Arguments

tree a modified object of class "phylo" (see read.simmap).
tip name or names of species to be dropped.
node node number for the root node of the clade to be extracted.

Value

A modified object of class "phylo" containing the elements maps and $mapped.edge with the time spent in each state along each edge of the tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

brownie.lite, drop.tip, extract.clade, make.simmap, read.simmap, sim.history
estDiversity

Estimate diversity at each node of the tree

Description

This function estimates the lineage density at each node in the tree based on a biogeographic model (based on Mahler et al. 2010).

Usage

```
estDiversity(tree, x, method=c("asr","simulation"), model="ER", ...)
```

Arguments

tree is a phylogenetic tree in "phylo" format.
x a vector containing the biogeographic area for each of the tip taxa.
method method for reconstructing ancestral biogeography.
model model for ancestral character estimation. In theory, any model from ace; however only symmetric models permitted for method="asr".
... optional arguments. So far, this includes only nsim, the number of stochastic mappings to conduct using make.simmap for method="simulation".

Details

Two different methods are implemented in the current version. For method="asr" the state at the current node, and at each position along each co-extant internal edge, is computed as the marginal (empirical Bayesian) ancestral state reconstruction using the re-rooting method of Yang (2006). The lineage density is then computed as the sum of the marginal reconstructions (posterior probabilities) times the summed marginal ancestral reconstructions across co-extant edges. In method="simulation", stochastic character mapping is used to generate optional argument nsim stochastic maps of ancestral biogeography. Then the lineage density at each node is computed as the number of co-existing lineages with the same biogeography as the focal node, averaged acrossed stochastic maps. The importance of this distinction may depend on the degree to which reconstructions at internal nodes are independent, which relates to the distinction between marginal and joint reconstruction (e.g., see Yang 2006).

Value

A vector containing the estimated lineage density at each node

Author(s)

Liam Revell <liam.revell@umb.edu>
References


See Also

`fitDiversityModel`

---

evol.rate.mcmc  
*Bayesian MCMC method for identifying exceptional phenotypic diversification in a phylogeny*

Description

This function takes a phylogenetic tree and data for a single continuously valued character and uses a Bayesian MCMC approach to identify the phylogenetic location of a shift in the evolutionary rate through time.

Usage

```r
evol.rate.mcmc(tree, x, ngen=10000, control=list())
```

Arguments

tree  
a phylogenetic tree in "phylo" format.

x  
a vector of tip values for species; names(x) should be the species names.

ngen  
an optional integer value indicating the number of generations for the MCMC.

control  
a list of control parameters containing the following elements: sig1: starting value for \( \sigma(1)^2 \); sig2: starting value for \( \sigma(2)^2 \); a: starting value for \( a \); sd1: standard deviation for the normal proposal distribution for \( \sigma(1)^2 \); sd2: standard deviation for the normal proposal distribution for \( \sigma(2)^2 \); kloc: scaling parameter for tree move proposals - 1/\( \lambda \) for the reflected exponential distribution; sdlnr: standard deviation on the log-normal prior on \( \sigma(1)^2/\sigma(2)^2 \); rand.shift: probability of proposing a random shift in the tree (improves mixing); print: print frequency for the MCMC; sample: sample frequency.

Details

Default values of control are given in Revell et al. (2012).
Value
A list with the following components:

- **mcmc**: results from the MCMC run.
- **tips**: list of stips in rate \( \sigma(1)^2 \) for each sampled generation of MCMC (to polarize the rate shift).

Author(s)
Liam Revell <liam.revell@umb.edu>

References


See Also
- `anc.Bayes`, `brownie.lite`, `evol.vcv`, `minSplit`, `posterior.evolrate`

---

**evol.vcv**

*Likelihood test for variation in the evolutionary VCV matrix*

Description
This function takes a modified "phylo" object with a mapped binary or multistate trait and data for an arbitrary number of continuously valued character. It then fits the multiple evolutionary variance-covariance matrix (rate matrix) model of Revell & Collar (2009; *Evolution*).

Usage
`evol.vcv(tree, X, maxit=2000, vars=FALSE, ...)`

Arguments
- **tree**: a phylogenetic tree in modified "phylo" format (see `read.simmap`).
- **X**: an \( n \times m \) matrix of tip values for \( m \) continuously valued traits in \( n \) species - row names should be species names.
- **maxit**: an optional integer value indicating the maximum number of iterations for optimization - may need to be increased for large trees.
- **vars**: an optional logical value indicating whether or not to estimate the variances of the parameter estimates from the Hessian matrix.
- **...**: optional arguments.
Details

This function performs optimization by first optimizing the likelihood with respect to the Cholesky matrices using `optim`. Optimization is by method="Nelder-Mead". Using box constraints does not make sense here as they would be applied to the Cholesky matrix rather than the target parameters. May have to increase `maxit` for large trees and more than 2 traits.

Value

A list with the following components:

- `R.single` vcv matrix for the single rate matrix model.
- `vars.single` optionally, a matrix containing the variances of the elements of `R.single`.
- `logL1` log-likelihood for single matrix model.
- `k1` number of parameters in the single matrix model.
- `R.multiple` \( m \times m \times p \) array containing the \( p \) estimated vcv matrices for the \( p \) regimes painted on the tree.
- `vars.multiple` optionally, an array containing the variances of the parameter estimates in `R.multiple`.
- `logL.multiple` log-likelihood of the multi-matrix model.
- `k2` number of parameters estimated in this model.
- `p.chisq` \( P \)-value of the \( \chi^2 \) test on the likelihood ratio.
- `convergence` logical value indicating whether or not the optimization has converged.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

evol.rate.mcmc, brownie.lite
evolvcev.lite

Likelihood test for a shift in the evolutionary correlation between traits

Description

This function takes a modified "phylo" object with a mapped binary or multistate trait and data for two and only two continuously valued character. It then fits four different evolutionary models: common rates and correlation; different rates, common correlation; different correlations, common rates; no common structure.

Usage

```r
evolvcev.lite(tree, X, maxit=2000, tol=1e-10)
```

Arguments

- **tree**: a phylogenetic tree in modified "phylo" format (see `read.simmap`).
- **X**: an n x m matrix of tip values for m continuously valued traits in n species - row names should be species names.
- **maxit**: an optional integer value indicating the maximum number of iterations for optimization - may need to be increased for large trees.
- **tol**: tolerance value for "L-BFGS-B" optimization.

Value

A list with the results summarized for each model.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

`brownie.lite`, `evol.vcv`
**exhaustiveMP**

*Exhaustive and branch & bound MP optimization*

**Description**

This function does exhaustive and branch & bound MP searches.

**Usage**

`exhaustiveMP(data, tree=NULL, method="branch.and.bound")`

**Arguments**

- `data` is a `phyDat` (Schliep 2011) object containing DNA or other data.
- `tree` an optional input tree (used only with method="branch. and. bound").
- `method` an optional string indicating method to use: "branch. and. bound" or "exhaustive".

**Details**

Should probably not be used for more than about 8 species (and definitely not more than 10 species). Performs parsimony calculations using `parsimony` in the "phangorn" package (Schliep 2011).

**Value**

A "phylo" or "multiPhylo" object that is the MP tree or set of MP trees. It also returns the parsimony scores in `attr(trees,"pscore")` or `attr(trees[[i]],"pscore")` for the ith tree.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

`mrp.supertree`, `optim.parsimony`, `pratchet`
**expm**

*Matrix exponential*

---

**Description**

Wrapper for `MatrixExp` that retains row/column names.

**Usage**

`expm(Y)`

**Arguments**

- `Y`: a matrix.

**Value**

A matrix.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


---

**export.as.xml**

*Export trees & data in XML format*

---

**Description**

This function exports trees & character data in XML format.

**Usage**

`export.as.xml(file, trees, X)`

**Arguments**

- `file`: filename for export.
- `trees`: a phylogenetic tree or trees in "phylo" or "multiPhylo" format.
- `X`: a matrix of class "DNAbin" or a matrix with discretely valued non-DNA character data.
Details

Can be used to create input file for the program SIMMAP v1.5 (Bollback 2006), also see: http://www.simmap.com.

Value

A file.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

make.simmap, read.nexus, read.simmap, write.simmap

---

**fancyTree**  
Plots special types of phylogenetic trees

Description

This function plots different types of phylogenetic trees. If `type="extinction"` (or any unambiguous abbreviation) the function will plot a tree in which branches preceding the MRCA of all extant taxa and branches leading only to extinct lineages are plotted with dashed red lines. If `type="traitgram3d"` the function will plot a three dimensional traitgram (that is, a projection of the tree into three dimensional morphospace where two dimensions are the phenotypic trait and the third axis is time since the root). In this case, the additional argument `X`, a matrix containing the tip values of all species (with species IDs as row names) should be supplied. Optionally, the user can also supply the matrix `A`, which contains the ancestral states in the tree with rows labeled by node number. If `type="dropTip"` the function will create a two panel figure in which the first panel is the tree with lineages to be pruned highlighted; and the second panel is the pruned tree. In this case, the additional argument `tip`, the tip name or vector of tip names to be dropped, must be supplied. If `type="densityMap"`, a posterior probability density "heat-map" is created based on a set of trees in a "multiPhylo" object containing a binary [0,1] mapped character. (See `densityMap` for additional optional arguments if `type="densityMap"`.) If `type="contMap"`, reconstructed continuous trait evolution is mapped on the tree. Again, see `contMap` for additional arguments if `type="contmap"`. If `type="phenogram95"` a 95-percent phenogram is plotted using transparency to visualize uncertainty at ancestral nodes and along branches. Most of the options of `phenogram` are available. Finally, if `type="scattergram"` a phylogenetic scatter plot matrix
containing `contMap` style trees on the diagonal and `phylomorphospace` plots in non-diagonal panels is produced. For `type`="scattergram" a trait matrix `X` must be supplied. The only additional arguments available for this type are `fsize`, `colors`, and `label`. (See `phylomorphospace` for details.) Presently only `type`="traitgram3d" uses the list control which can be supplied the same set of control parameters as `phylomorphospace3d`, as well as the control parameter `maxit` which will be passed to `anc.ML`. Finally, the optional argument `hold` will be passed to multiple methods if supplied. It is a logical value that indicates whether or not the output to the graphical device should be held using `dev.hold` before plotting (defaults to `hold=TRUE`).

Usage

```r
fancyTree(tree, type=c("extinction","traitgram3d","droptip","densitymap",
"contmap","phenogram95","scattergram"), ..., control=list())
```

Arguments

- `tree` a phylogenetic tree in “phylo” format.
- `type` the type of special plot to create. See Description.
- `...` arguments to be passed to different methods.
- `control` a list of control parameters, depending on `type`.

Value

This function plots different types of phylogenetic trees. For `type="droptip"` the function also returns the pruned tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

`contMap`, `densityMap`, `drop.tip`, `phenogram`, `phylomorphospace3d`, `plot.phylo`, `plotSimmap`

Examples

```r
# plot tree with extinction
set.seed(10)
tree<-pbtree(b=1,d=0.4,t=4)
fancyTree(tree,type="extinction")

## Not run:
# plot 3D traitgram
tree<-pbtree(n=50,scale=10)
Y<-sim.corrs(tree,vcv=matrix(c(1,0.75,0.75,1),2,2))
```
fancyTree(tree,type="traitgram3d",X=Y,control=list(spin=FALSE))

# plot with internal nodes from simulation
Y<-sim.corrs(tree,vcv=matrix(c(1,0.75,0.75,1),2,2),internal=TRUE)
B<-Y[1:length(tree$tip)+1:tree$Nnode,]; Y<-Y[1:length(tree$tip),]
fancyTree(tree,type="traitgram3d",X=Y,A=B,control=list(simple.axes=TRUE,spin=FALSE))

## End(Not run)

# plot with dropped tips
tree<-pbtree(n=30)
tips<-sample(tree$tip.label)[1:10]
pruned<-fancyTree(tree,type="droptip",tip=tips)

## Not run:
# plot 95-percent CI phenogram
tree<-pbtree(n=30)
x<-fastBM(tree)
fancyTree(tree,type="phenogram95",x=x)

## End(Not run)

---

**fastAnc**

*Fast estimation of ML ancestral states*

**Description**

This function performs fast estimation of the ML ancestral states for a continuous trait by taking advantage of the fact that the state computed for the root node of the tree during Felsenstein’s (1985) contrasts algorithm is also the MLE of the root node. Thus, the function reroots the tree at all internal nodes and computes the contrasts state at the root each time. The function can also (optionally) compute variances or 95-percent confidence intervals on the estimates.

**Usage**

```
fastAnc(tree, x, vars=FALSE, CI=FALSE)
```

**Arguments**

- `tree`  
an object of class "phylo".
- `x`  
a vector of tip values for species; names(x) should be the species names.
- `vars`  
a logical value indicating whether or not to compute variances on the ancestral state estimates. Variances are based on Equation (6) of Rohlf (2001).
- `CI`  
a logical value indicating whether or not to compute 95-percent confidence intervals on state estimates.
**Value**

A named vector containing the states at internal nodes - names are node numbers; or a list containing ancestral state estimates (ace), variances on the estimates (var), and/or 95-percent confidence intervals (CI95).

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

ace, anc.Bayes, anc.ML, pic

**Examples**

```r
  tree<-pbtree(n=50)
  x<-fastBM(tree) # simulate using fastBM
  fastAnc(tree,x) # estimate states
```

---

**Description**

This function conducts fast quantitative trait simulation on a phylogeny under several different models: Brownian motion (default), BM with a trend (for mu!=0), bounds (for bounds!=c(-Inf,Inf)), and OU.

**Usage**

```r
  fastBM(tree, a=0, mu=0, sigR=1, bounds=c(-Inf,Inf), internal=FALSE, nsim=1, ...)
```

**Arguments**

- `tree` is a phylogenetic tree in "phylo" format.
- `a` a value for ancestral state at the root node.
- `mu` an optional value for the mean of random normal changes along branches of the tree - can be used to simulate a trend if mu!=0.
- `sigR` instantaneous variance of the BM process.
- `bounds` a vector with the lower and upper bounds (respectively) for bounded Brownian simulation - by default simulation is unbounded.
fastMRCA

Description

This function returns the most recent common ancestor (node number) for a pair of taxa; or, in the case of fastHeight, the height above the root of the MRCA of a pair of taxa.

Usage

fastMRCA(tree, sp1, sp2)
fastHeight(tree, sp1, sp2)

Arguments

tree a phylogenetic tree as an object of class "phylo".
sp1 species name.
sp2 species name.
Details

This function is mostly redundant with findMRCA (or findMRCA(...,type="height") in the case of fastHeight) but for very large trees will be considerably faster.

Value

The node number of the MRCA or the height above the root (for fastHeight).

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

findMRCA, mrca

Examples

tree<-pbtree(n=2000)
anc<-fastMRCA(tree,"t1","t15")

---

**findMRCA**

*Get the MRCA of a set of taxa*

Description

This function returns the most recent common ancestor (node number) for a set of taxa. If tips=NULL will be redundant with mrca (for type="node") or vcv.phylo, but much slower (for type="height").

Usage

findMRCA(tree, tips=NULL, type=c("node","height"))

Arguments

tree a phylogenetic tree as an object of class "phylo".
tips a vector containing a set of tip labels.
type either "node" to return the node of the MRCA; or "height" to return the height above the root of the MRCA of tips.
Details

If tips==NULL will return the result of a normal function call to mrca. If tips=NULL will return a matrix equal to vcv.phylo.

Value

The node number of the MRCA, or a matrix of node numbers (if tips==NULL) - for type="node"; or the height of the MRCA, or a matrix of heights (if tips==NULL) - for type="height".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

findMRCA, mrca

Examples

tree<-pbtree(n=20)
ancc<-findMRCA(tree,c("t1","t10","t15"))

fitBayes

Evolutionary model fitting with intraspecific variability using Bayesian MCMC

Description

This function uses Bayesian MCMC to sample terminal states (species means) as well as evolutionary parameters.

Usage

fitBayes(tree, x, ngen=10000, model="BM", method="reduced", control=list())

Arguments

tree an object of class "phylo".
x a vector of phenotypic values for individuals; names(x) should contain the species names (not individual IDs).
ngen a integer indicating the number of generations for the MCMC.
model an evolutionary model: either "BM" or "lambda".
### Method

A method: either "reduced" or "full".

### Control

A list of control parameters containing the following elements: sig2: starting value for $\sigma^2$ (BM rate); lambda: starting value for the $\lambda$ parameter; a: starting for the state at the root node; xbar: starting values for the states at the tips; intV: starting value for the intraspecific variance (reduced method); or v: starting value for the vector of intraspecific variances for all species (full method); pr.mean: means for the prior distributions in the following order - sig2, lambda (if applicable), a, xbar, intV or v (if applicable), note that the prior probability distribution is exponential for sig2 and normal for a and y; pr.var: variances on the prior distributions, same order as pr.mean.

### Value

A matrix with number of rows ngen/control$sample+1 containing the posterior sample and likelihoods. Matrix columns are labeled by species (for species means and variances), or by the corresponding evolutionary parameter.

### Author(s)

Liam Revell <liam.revell@umb.edu>

### References


### See Also

anc.Bayes, brownie.lite, evol.rate.mcmc

### fitDiversityModel

*Fit diversity-dependent phenotypic evolution model*

### Description

This function fits a diversity-dependent phenotypic evolution model (based on Mahler et al. 2010).

### Usage

```r
fitDiversityModel(tree, x, d=NULL, showTree=TRUE, tol=1e-6)
```
Arguments

- **tree**: an object of class "phylo".
- **x**: a vector with tip values for a continuously distributed trait.
- **d**: a vector containing the inferred historical diversity at each node in the tree - if d=NULL (the default) function will treat the diversification as if it occurred in a single geographic area.
- **showTree**: optional logical value indicating whether to plot the tree transformation implied by the model.
- **tol**: some small value by which d is incremented during rescaling of psi for optimization. If R thinks your matrices are singular during optimization, try increasing tol slightly.

Value

A list with the following components:

- **logL**: log-likelihood of the fitted model.
- **sig0**: estimated starting value for the rate at the root of the tree.
- **psi**: the estimated rate of change in the rate associated with the addition of a lineage.
- **vcv**: a matrix with the variances and covariance of the estimated parameters (from the Hessian).

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

- `brownie.lite.estDiversity`
- `evol.rate.mcmc`
**fitPagel**

*Function to test for correlated evolution of binary traits*

**Description**

This function fits Pagel’s (1994) model for the correlated evolution of two binary characters.

**Usage**

```
fitPagel(tree, x, y, NNNI)
```

**Arguments**

- `tree`: an object of class "phylo".
- `x`: a vector of phenotypic values for a binary trait for the species in `tree`.
- `y`: a second binary character for the species in `tree`.
- `NNN`: optional arguments. Currently includes `method`, which can be set to "ace" to use the `ace` function in ape for optimization, or to "fitDiscrete" (if the geiger package is installed) to use geiger’s `fitDiscrete` for optimization.

**Value**

An object of class "fitPagel" which contains the optimized matrices under an independence & a dependence model, log-likelihoods, a likelihood ratio, and a P-value for the independence model based on a chi-squared test.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

ace, make.simmap
gammatest

Description

Conducts $\gamma$-test of Pybus & Harvey (2000).

Usage

gammatest(x)

Arguments

x list from function call of ltt in which ltt(...)\[gamma\]fI.

Details

Do not use for object returned by ltt(...)\[gamma\]tI.

Value

A list containing:

- gamma optionally, a value for the $\gamma$-statistic.
- p two-tailed P-value for the $\gamma$-test.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

ltt

Examples

tree<-pbtree(n=200)
z<-ltt(tree,\[gamma\]=FALSE)
g<-gammatest(z)
Simulate a DNA alignment on the tree under a model

Description

Simulates DNA sequence on tree under the specified model. Uses sim.history internally.

Usage

genSeq(tree, l=1000, Q=NULL, rate=1, format="DNAbin", ...)

Arguments

tree object of class "phylo".
l length of desired sequences.
Q transition matrix for the simulation. Row and column names c("a", "c", "g", "t") (although not necessarily in that order, should be provided. If NULL, a single rate is assumed.
rate multiplier for Q, or a vector for Gamma rate heterogeneity.
format format of the output object. Can be "DNAbin", "phyDat", or "matrix".
... optional arguments.

Value

An object of class "DNAbin" or "phyDat", or a matrix of nucleotides.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


Examples

## simulate gamma rate heterogeneity
tree<-pbtree(n=26,tip.label=LETTERS)
gg<-rgamma(n=100,shape=0.25,rate=0.25)
X<-genSeq(tree,l=100,rate=gg)
getCladesofSize  
*Get all subtrees larger than or equal to a specified size*

**Description**

This function gets all subtrees that cannot be further subdivided into two reciprocally monophyletic subtrees of size $\geq$ clade.size.

**Usage**

```r
getCladesofSize(tree, clade.size=2)
```

**Arguments**

- `tree` is an object of class "phylo".
- `clade.size` subtree size.

**Value**

An object of class "multiPhylo".

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

`extract.clade`, `getDescendants`

getDescendants  
*Get descendant node numbers*

**Description**

This function returns the set of node & tip numbers descended from node.

**Usage**

```r
getDescendants(tree, node, curr=NULL)
```
**Arguments**

- `tree`: A phylogenetic tree as an object of class "phylo".
- `node`: An integer specifying a node number in the tree.
- `curr`: The set of previously stored node numbers - used in recursive function calls.

**Value**

The set of node and tip numbers for the nodes and tips descended from `node` in a vector.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

- `paintSubTree`

---

**Description**

The function `getExtant` takes a tree as input and returns a vector containing the names of all the tips that have a height above the root that is equal (to a degree of numerical precision determined by `tol`) to the height of the highest tip. `getExtinct` returns the complement.

**Usage**

```r
getExtant(tree, tol=1e-8)
getExtinct(tree, tol=1e-8)
```

**Arguments**

- `tree`: A phylogeny stored as an object of class "phylo" with some tips that are non-contemporaneous (i.e., end before the present).
- `tol`: A tolerance value to account for numerical imprecision.

**Value**

A vector with the tip names of extant or extinct species in the tree.
getSisters

Get the sister node number, label, or set of nodes for a node or tip

Description

This function takes a tree and node or tip number of label and returns the number or label of the sister or sisters to that node or tip.

Usage

getSisters(tree, node, mode=c("number","label"))

Arguments

tree: object of class "phylo".
node: a node number, tip number, node label, or tip label.
mode: an optional string indicating whether to return the node or tip number(s) or the node or tip label(s), if available.

Value

If mode="number" this function returns an integer or vector containing the node number of numbers of the sister node or tip. If mode="label" that this function returns a list containing up to two vectors: one for the node numbers of labels of sister nodes (if available); and the other containing the tip labels of the sister tips.

Author(s)

Liam Revell <liam.revell@umb.edu>

References

**getStates**  
*Get the states at nodes or tips from a mapped tree*

**Description**
This function gets the states from the nodes or tips of a mapped tree (e.g., `make.simmap`).

**Usage**
```
getStates(tree, type=c("nodes","tips"))
```

**Arguments**
- **tree**: is a modified object of class "phylo" or "multiPhylo".
- **type**: mode indicating whether to get states at the nodes (type="nodes") or the tips (type="tips") of the tree.

**Value**
A named vector (for "phylo") or matrix (for "multiPhylo").

**Author(s)**
Liam Revell <liam.revell@umb.edu>

**References**

**See Also**
- `describe.simmap`
- `make.simmap`
- `read.simmap`
- `sim.history`

---

**ladderize.simmap**  
*Ladderize a tree with a mapped discrete character*

**Description**
This function "ladderizes" an object of class "phylo" with a mapped discrete character. For more information, please see `ladderize`.

**Usage**
```
ladderize.simmap(tree, right=TRUE)
```
Arguments

- `tree` an object of class "phylo" with a mapped discrete character.
- `right` a logical specifying how the tree should be ladderized.

Value

An object of class "phylo" with a mapped discrete character.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

`make.simmap`, `ladderize`

---

**lambda.transform**

*Lambda transformation of matrix*

Description

Function multiplies the off-diagonals of a square matrix by lambda. Used internally in `phyl.pca` and other functions.

Usage

`lambda.transform(lambda, C)`

Arguments

- `lambda` scalar, usually (but not necessarily) on the interval 0,1.
- `C` matrix probably returned by `vcv.phylo`.

Details

Do not use unless you know what you’re doing.

Value

A matrix.
likMlambda

Author(s)

Liam Revell <liam.revell@umb.edu>

References


---

**likMlambda**

*Likelihood for joint lambda*

**Description**

Computes the likelihood.

**Usage**

`likMlambda(lambda, X, C)`

**Arguments**

- `lambda`: scalar, usually on the interval 0,1.
- `X`: data, in a matrix.
- `C`: matrix probably returned by `vcv.phylo`.

**Details**

Do not use unless you know what you’re doing.

**Value**

A scalar.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**

locate.fossil

Locate a fossil lineage in a tree using continuous characters

Description

This function uses ML to place a fossil lineage into a tree using continuous traits.

Usage

`locate.fossil(tree, X, ...)`

Arguments

- `tree`: an object of class "phylo".
- `X`: a matrix with continuous character data.
- `...`: optional arguments including `time.constraint` which can be a scalar (positive height above the root of the fossil or negative time before present) or a vector (age range of fossil, either positive or negative); `edge.constraint`, which is equivalent to `constraint` in `locate.yeti`; `plot`, `rotate`, and `quiet`, which have the same interpretation (and defaults) as the equivalent arguments in `locate.yeti`.

Value

Optimized tree as an object of class "phylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


locate.yeti

Locate a cryptic, recently extinct, or missing taxon on a tree

Description

This function uses ML (or REML) to place a recently extinct, cryptic, or missing taxon on an ultrametric (i.e., time-calibrated) phylogeny.

Usage

locate.yeti(tree, X, ...)

Arguments

tree an object of class "phylo".
X a matrix with continuous character data.
... optional arguments including: method ("ML" or "REML", defaults to "ML"); search ("heuristic" or "exhaustive", defaults to "heuristic"); constraint, a vector containing the daughter node numbers from tree$edge for each edge to try; plot a logical argument specifying whether or not to plot the likelihood profile on edges (defaults to FALSE); rotate a logical indicating whether or not to rotate the data based on the input tree; and quiet, which is logical and has an obvious interpretation.

Value

Optimized tree as an object of class "phylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


ls.tree  

Least squares branch lengths for a given tree

Description
Computes the least squares branch lengths conditioned on a topology and distance matrix. Internal function for optim.phylo.ls.

Usage
ls.tree(tree, D)

Arguments
tree  
phylogeny.
D    
distance matrix.

Details
Do not use unless you know what you’re doing.

Value
A tree with branch lengths.

Author(s)
Liam Revell <liam.revell@umb.edu>

References

ltt  

Creates lineage-through-time plot (including extinct lineages)

Description
This function computes LTT plot with extant and extinct lineages, and optionally conducts $\gamma$-test of Pybus & Harvey (2000). The object returned by ltt can be plotted or re-plotted using plot.

Usage
ltt(tree, plot=TRUE, drop.extinct=FALSE, log.lineages=TRUE, gamma=TRUE, ...)

Arguments

tree is a phylogenetic tree in "phylo" format, or an object of class "multiphylo" containing a list of phylogenetic trees.

plot a logical value indicating whether or not to create LTT plot.
drop.extinct logical value indicating whether or not to drop extinct tips from the tree.
log.lineages logical value indicating whether LTT plot should be on log-linear (default) or linear-linear scale.
gamma logical value indicating whether or not to compute eqngamma from Pybus & Harvey (2000; Proc. Roy. Soc. B).

... other arguments to be passed to plotting methods. See plot.default.

Details

Although it is calculated here, it's unclear how to interpret the γ-statistic if not all the tips in the tree are contemporaneous.

Value

An object of class "ltt" which includes the following components:

times a vector of branching times.
ltt a vector of linages.
gamma optionally, a value for the gamma-statistic.
p two-tailed P-value for the gamma-test.

If tree is an object of class "multiphylo", then an object of class "multiltt" is returned consisting of a list of object of class "ltt".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

gammatest, ltt95
Examples
trees <- pbtree(n=100, scale=100, nsim=10)
obj <- ltt(trees, plot=FALSE)
plot(obj, log="y", log.lineages=FALSE, main="lineage through time plots")
tree <- pbtree(b=1, d=0.25, t=4)
obj <- ltt(tree, gamma=FALSE)
obj

\texttt{ltt95}

\textit{Creates a (1-\alpha)-percent CI for a set of LTTs}

Description
This function computes LTT plots for a set of trees & plots a (1-\alpha)-percent CI by various methods.

Usage
\texttt{ltt95(trees, alpha=0.05, log=FALSE, method=c("lineages","times"),
mode=c("median","mean"), ...)
## S3 method for class 'ltt95'
plot(x, ...)

Arguments
- \texttt{trees} is an object of class "multiPhylo" containing a list of phylogenetic trees.
- \texttt{alpha} confidence level.
- \texttt{log} logical value indicating whether or not to plot on the log-scale.
- \texttt{method} plot the CI on the number of lineages given time ("lineages"); or on times given a number of lineages ("times").
- \texttt{mode} plot the median or mean LTT.
- \texttt{x} object of class "ltt95" for plotting method.
- ... optional arguments to be used by \texttt{ltt95} or the plotting method. So far, \texttt{res} gives the number of time-steps (defaults to \texttt{res=100}. \texttt{xaxis(standard,"negative", or "flipped") determines the scale (time from the root, time back from the present, or time from the present) of the x-axis of the plot.

Details
This function creates a plot and invisibly returns an object of class "ltt95".

Author(s)
Liam Revell <liam.revell@umb.edu>
References


See Also

ltt

make.era.map

Create "era" map on a phylogenetic tree

Description

This function creates a temporal map on the tree based on limits provided by the user.

Usage

make.era.map(tree, limits, ...)

Arguments

tree
limits
... a phylogenetic tree as an object of class "phylo".
a vector containing the temporal limits, in time since the root node of the tree,
optional arguments.
for the mappings.

Value

A modified phylogenetic tree of class "phylo" with the following additional elements:

maps
mapped.edge

a list of named vectors containing the times spent in each state on each branch,
a matrix containing the total time spent in each state along each edge of the tree,
in the order in which they occur.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

make.simmap, read.simmap, plotSimmap
Examples

```r
make.simmap

# Example

tree <- pbtree(n=1000, scale=100)
tree <- make.era.map(tree, c(0, 25, 50, 75))
plotSimmap(tree, pts=FALSE, ftype="off")
```

**make.simmap**  
*Simulate stochastic character maps on a phylogenetic tree or trees*

**Description**

This function performs stochastic mapping using several methods.

For Q="empirical", it first fits a continuous-time reversible Markov model for the evolution of x and then simulates stochastic character histories using that model and the tip states on the tree. This is the same procedure that is described in Bollback (2006), except that simulation is performed using a fixed value of the transition matrix, Q, instead of by sampling Q from its posterior distribution.

For Q="mcmc", it first samples Q nsim times from the posterior probability distribution of Q using MCMC, then it simulates nsim stochastic maps conditioned on each sampled value of Q.

For Q set to a matrix, it samples stochastic mappings conditioned on the fixed input matrix.

**Usage**

```r
make.simmap(tree, x, model="SYM", nsim=1, ...)
```

**Arguments**

- `tree`  
a phylogenetic tree as an object of class "phylo", or a list of trees as an object of class "multiPhylo".

- `x`  
a vector containing the tip states for a discretely valued character, or a matrix containing the prior probabilities of tip states in rows.

- `model`  
a character string containing the model - options as in ace.

- `nsim`  
number of simulations. If tree is an object of class "multiPhylo", then nsim simulations will be conducted per tree.

- `...`  
optional arguments. So far, pi gives the prior distribution on the root node of the tree - options are "equal", "estimated", or a vector with the frequencies. If pi="estimated" then the stationary distribution is estimated by numerically solving pi*Q=0 for pi, and this is used as a prior on the root. Defaults to pi="equal" which results in the root node being sampled from the conditional scaled likelihood distribution at the root. message tells whether or not to print a message containing the rate matrix, Q and state frequencies. Defaults to message=TRUE. For optional argument Q="mcmc" the mean value of Q from the posterior sample is printed. tol gives the tolerance for zero elements in Q. (Elements less than tol will be reset to tol). Q can be a string ("empirical" or "mcmc"), or a fixed value of the transition matrix, Q. If "empirical" than a single value of Q, the most likely value, is used for all simulations. If "mcmc",
then nsim values of $Q$ are first obtained from the posterior distribution for $Q$ using Bayesian MCMC, then a simulated stochastic character map is generated for each value of $Q$. vQ a single numeric value or a vector containing the (normal) sampling variances for the MCMC. The order of vQ is assumed to be in the order of the index.matrix in ace for the chosen model. prior a list containing alpha and beta parameters for the gamma prior distribution on the transition rates in $Q$. Note that alpha and beta can be single values or vectors, if different priors are desired for each value in $Q$. As for vQ, the order of prior is assumed to be the order of index.matrix in ace. prior can also be given the optional logical value use.empirical which tells the function whether or not to give the prior distribution the empirical mean for $Q$. If TRUE then only prior$beta$ is used and prior$alpha$ is set equal to prior$beta$ times the empirical mean of $Q$. burnin and samplefreq are burn-in and sample frequency for the MCMC, respectively.

Details

Uses code modified from ace (by Paradis et al.) to perform Felsenstein’s pruning algorithm & compute the likelihood.

As of phytools>=0.2-33 x can be a vector of states or a matrix containing the prior probabilities of tip states in rows. In this case the column names of x should contain the states, and the row names should contain the tip names.

Note that there was a small (but potentially significant) bug in how node states were simulated by
make.simmap in versions of phytools<=0.2-26. Between phytools 0.2-26 and 0.2-36 there was also a bug for asymmetric models of character change (e.g., model="ARD"). Finally, between phytools 0.2-33 and phytools 0.2-47 there was an error in use of the conditional likelihoods for the root node, which caused the root node of the tree to be sampled incorrectly. All of these issues should be fixed in the present version.

Q="mcmc" and Q set to a fixed value were introduced to phytools >= 0.2-53. As of the present version of phytools, this method is still somewhat experimental & should be used with caution.

If tree is an object of class "multiPhylo" then nsim stochastic maps are generated for each input tree.

Value

A modified phylogenetic tree of class "phylo" (or a modified "multiPhylo" object, for nsim > 1) with the following additional elements:

maps a list of named vectors containing the times spent in each state on each branch, in the order in which they occur.

mapped.edge a matrix containing the total time spent in each state along each edge of the tree.

Q the assumed or sampled value of $Q$.

logL the log-likelihood of the assumed or sampled $Q$.

Author(s)

Liam Revell <liam.revell@umb.edu>
References


See Also

`brownie.lite`, `brownieREML`, `countSimmap`, `describe.simmap`, `evol.vcv`, `plotSimmap`, `read.simmap`, `write.simmap`

map.overlap

*Proportional overlap between two mapped character histories on a tree*

Description

This function computes the fraction of a stochastic character mapping that is shared between two differently mapped trees.

Usage

```r
map.overlap(tree1, tree2, tol=1e-6)
```

Arguments

- `tree1`: a modified "phylo" object (see `read.simmap`.
- `tree2`: a modified "phylo" object with the same topology and branch lengths, but different map from `tree1`.
- `tol`: an optional tolerance value.

Value

A numerical value on the interval 0-1.

Author(s)

Liam Revell <liam.revell@umb.edu>

References

map.to.singleton

See Also

make.simmap, read.simmap

map.to.singleton

Converts a mapped tree to a tree with singleton nodes

Description

The function map.to.singleton takes an object of class "phylo" with a mapped discrete character and converts it to a tree with singleton nodes, in which edge has only one state. The states for each edge are stored in names(tree$edge.length). plotTree.singletons plots a tree with singleton nodes. Finally, drop.tip.singleton drops tips from the tree leaving ancestral nodes for all remaining tips as singletons.

Usage

map.to.singleton(tree)
plotTree.singletons(tree)
drop.tip.singleton(tree, tip)

Arguments

tree a modified object of class "phylo" with a mapped discrete character or (for plotTree.singletons a tree with singleton nodes.
tip a tip label or vector of tip labels.

Value

An object of class "phylo" with singleton nodes. plotTree.singletons plots a tree. If names(tree$edge.length)!=NULL it will use a different color from palette for each mapped state.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

collapse.singles, drop.tip, make.simmap
matchNodes

Matches nodes between two trees

Description

This function returns a matrix in which the first column contains all the internal nodes for tr1 and the second column contains the matching nodes from tr2, inasmuch as they can be identified. For method="descendants", pairs of matching nodes are defined by sharing all descendant leaves in common. For method="distances", nodes are considered to matched if the share the same set of distances (or proportional distances, for optional argument corr=TRUE) to all tips.

Usage

matchNodes(tr1, tr2, method=c("descendants","distances"), ...)

Arguments

- tr1: first tree.
- tr2: second tree.
- method: method to use to match nodes between trees. "descendants" uses the tip species descended from each node; "distances" uses the distances from the nodes to the tips. Any unambiguous shortening of "descendants" or "distances" is allowed.
- ...: optional arguments which may or may not be used depending on method. tol is a tolerance value for the difference from exact matching that is allowed for method="distances". corr, which is FALSE by default, indicates whether to match nodes under method="distances" using the correlation (corr=TRUE) or the absolute similarity of distances.

Details

Primarily designed to be used internally by fastAnc.

Value

A matrix in which the first column contains the nodes of tr1 with the second column containing matching nodes in tr2, with the criterion for matching defined by method.

Author(s)

Liam Revell <liam.revell@umb.edu>

References

mergeMappedStates

Merge two or more mapped states into one state

Description

This function merges two or mapped states on the tree to get one new state. For instance, one could merge the states "C", "G", and "T" to get the state "not-A".

Usage

mergeMappedStates(tree, old.states, new.state)

Arguments

tree a modified object of class "phylo" or "multiPhylo" with a mapped discrete character.
old.states states to merge.
new.state name for new state.

Value

A modified object of class "phylo" or "multiPhylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

make.simmap, read.simmap
midpoint.root

Midpoint root a phylogeny

Description
This function midpoint roots a rooted or unrooted tree (Farris 1972).

Usage
midpoint.root(tree)

Arguments
tree an object of class "phylo".

Details
Midpoint rooting involves locating the midpoint of the longest path between any two tips and putting the root in that location. This function performs the same operation as midpoint in the phangorn package, but uses no phangorn code internally.

Value
A phylogenetic tree in "phylo" format.

Author(s)
Liam Revell <liam.revell@umb.edu>

References

See Also
reroot, root
**minRotate**

Rotates all nodes of the tree to minimize the difference in order with a vector.

**Description**

This function rotates all the nodes of the tree to try and minimize the difference between the order of the tips and the rank-order of a numeric vector `x`.

**Usage**

```r
minRotate(tree, x)
```

**Arguments**

- `tree`: tree.
- `x`: numeric vector.

**Details**

Primarily designed to be used internally by `phylo.to.map`.

**Value**

A object of class "phylo".

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**minSplit**

Finding the minimum (median) split in the posterior sample

**Description**

This function takes a phylogenetic tree and a list of splits and identifies the split with the smallest summed or summed squared distances to all the other splits. Used to be called `min.split()` but was changed to avoid conflict with the generic `min`.

**Usage**

```r
minSplit(tree, split.list, method="sum", printD=FALSE)
```
mrp.supertree

Arguments

tree  a phylogeny stored as an object of class "phylo".
split.list  either a matrix with two named columns, "node" and "bp"; a $mcmc matrix from evol.rate.mcmc(); or the entire raw output from evol.rate.mcmc().
method  an optional string indicating the criterion to minimize: options are "sum" and "sumsq".
printD  logical specifying whether to print distances to screen (FALSE by default).

Value

A list with the following components:

node  node for the minimum split.
bp  location on the branch leading to node of the minimum split.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

evol.rate.mcmc, posterior.evolverate

Description

This function estimates the MRP (matrix representation parsimony) supertree from a set of trees (Baum 1992; Ragan 1992).

Usage

mrp.supertree(trees, method=c("pratchet","optim.parsimony"), ...)

Arguments

trees  an object of class "multiPhylo" (i.e., a list of trees).
method  an argument specifying whether to optimize the tree using pratchet or optim.parsimony.
...  optional arguments - mostly to be passed to pratchet or optim.parsimony.
Details

Function uses \texttt{pratchet} or \texttt{optim.parsimony} from the "phangorn" package (Schliep 2011) and \texttt{prop.part} from the "ape" package (Paradis et al. 2004). See \texttt{pratchet} or \texttt{optim.parsimony} for optional arguments, which vary slightly depending on the method. All optional arguments of these methods are available to the user with one exception. The argument tree in \texttt{optim.parsimony} is supplied instead as \texttt{start}. In addition to being an object of class "phylo", \texttt{start} can also be assigned the string values of "NJ" or "random", in which case either a neighbor-joining or random tree will be used as the starting tree for optimization.

Value

A "phylo" or "multiPhylo" object that is the MP or set of MP MRP trees.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

\texttt{exhaustiveMP, optim.parsimony, pratchet}

\begin{center}
\begin{tabular}{ll}
\texttt{multi.mantel} & \textit{Multiple matrix regression (partial Mantel test)}
\end{tabular}
\end{center}

Description

This function conducting a multiple matrix regression (partial Mantel test) and uses Mantel (1967) permutations to test the significance of the model and individual coefficients. It also returns the residual and predicted matrices.

Usage

\texttt{multi.mantel(Y, X, nperm=1000)}
Arguments

Y  single "dependent" square matrix. Can be either a symmetric matrix of class "matrix" or a distance matrix of class "dist".
X  a single independent matrix or multiple independent matrices in a list. As with Y can be a object of class "matrix" or class "dist".
nperm  number of Mantel permutations.

Value

A list with the following components:

r.squared  multiple R-squared.
coefficients  model coefficients, including intercept.
tstatistic  t-statistics for model coefficients.
fstatistic  F-statistic for the overall model.
probt  vector of probabilities, based on permutations, for tstatistic.
probf  probability of F, based on Mantel permutations.
residuals  matrix of residuals.
predicted  matrix of predicted values.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


multic  

Returns a list with phylogenetic VCV matrix for each mapped state

Description

This function takes a modified "phylo" object as input and returns a set of so-called phylogenetic covariance matrices as a list: one for each mapped state.

Usage

`multic(tree)`
multiRF

Arguments

tree a phylogeny with mapped discrete state in a modified object of class "phylo" (e.g., see read.simmap.

Value

A list of matrices.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also
evolcvv.lite, read.simmap, vcvPhylo, vcv.phylo

Description

Computes the Robinson-Foulds (Robinson & Foulds 1981) distance between a set of trees in an object of class "multiPhylo". If trees contains a large number of phylogenies (say 100 or 1,000) this will be much faster than calling Rf.dist in the phangorn package for all pairwise comparisons because it avoids repeating some internal calculations. Nonetheless for large numbers multiRF is slow, and will use lots of memory.

Usage

multiRF(trees)

Arguments

trees object of class "multiPhylo" consisting of two or more fully bifurcating, unrooted trees. If trees are rooted, they will be unrooted.

Details

Computes the Robinson-Foulds distance between all phylogenies in an object of class "multiPhylo". Uses prop.part internally for most of the heavy lifting.
Value

A distance matrix.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


nodeHeights

*Compute the heights above the root of each node*

Description

nodeHeights computes the height above the root for all nodes in the tree. nodeheight computes the height above the root for a single node.

Usage

nodeHeights(tree)
nodeheight(tree, node)

Arguments

tree a phylogeny as an object of class "phylo".
node for nodeheight, the node for which we want to compute a height above the root.

Details

The function nodeHeights also gives a handy way to get the total length of the tree from the root to the highest tip which will be given by max(nodeHeights(tree)). Generally speaking, nodeHeights will be faster if the heights of all or a large proportion of nodes is needed, whereas nodeheight will be faster if the height of one or a small number of nodes are needed.

Value

Either a matrix of the same dimensions as tree$edge containing the height above the root of each node in edge (for nodeHeights); or a single positive number (for nodeheight).

Author(s)

Liam Revell <liam.revell@umb.edu>
References


See Also

vcvPhylo

Examples

tree<-rtree(10)
X<-nodeHeights(tree)

optimum.phylo.ls

**Phylogeny inference using the least squares method**

Description

This function performs phylogeny inference using least-squares.

Usage

optim.phylo.ls(D, stree=NULL, set.neg.to.zero=TRUE, fixed=FALSE, tol=1e-10, collapse=TRUE)

Arguments

- **D**: a distance matrix.
- **stree**: an optional starting tree for the optimization.
- **set.neg.to.zero**: a logical value indicating whether to set negative branch lengths to zero (default TRUE).
- **fixed**: a logical value indicating whether to estimate the topology - if TRUE only the branch lengths will be computed.
- **tol**: a tolerance value used to assess whether the optimization has converged.
- **collapse**: a logical indicating whether to collapse branches with zero length.

Details

Function uses nni from the "phangorn" package (Schliep 2011) to conduct NNIs for topology estimation. Since topology optimization is performed using NNIs, converge to the true least-squares topology is not guaranteed. It is consequently probably wise to start with a very good tree - such as a NJ tree.
An object of class "phylo" that (may be) the least-squares tree with branch lengths; also returns the sum of squares in attr(tree, "Q-score").

Author(s)
Liam Revell <liam.revell@umb.edu>

References

See Also
exhaustiveMP, nni

Order the columns of mapped.edge to match across trees

This function takes a modified object of class "multiPhylo" with a mapped discrete character (e.g., see readSimmap and sorts the columns of each tree$mapped.edge to have the same state ordering. This is handy if we want to, for instance, run brownie.lite on a set of mapped trees, and then average the fitted parameter values across trees. The function also works for a single tree.

Usage
orderMappedEdge(trees, ordering=NULL)

Arguments
trees object of class "phylo" or "multiPhylo".
ordering ordering for the columns of $mapped.edge. If NULL, then an alphabetical order is assumed. Options are "alphabetical", "numerical", or any specific ordering of the mapped traits (e.g., c("A","B","C").
Value

A modified object of class "phylo" or "multiPhylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


Description

This function maps or "paints" an arbitrary, i.e., user-specified, discrete character history on the tree. paintSubTree paints the clade downstream of node with a particular state; whereas paintBranches paints only a specified branch.

Usage

paintSubTree(tree, node, state, anc.state="1", stem=FALSE)
paintBranches(tree, edge, state, anc.state="1")

Arguments

tree: a phylogenetic tree as an object of class "phylo" or a modified object with mapped character traits.
node: an integer specifying the node number tipward of which the function should paint the derived state.
edge: an integer or vector of integers specifying the node or tip numbers of the edges that should be painted in paintBranches.
state: a string (or numeric value) specifying the state to paint on the tree tipward of node.
anc.state: the ancestral state to use; will only be applied if there are presently no character values mapped on the tree.
stem: logical or numeric value indicating whether to use the derived state on the stem leading to node (or not, if stem=FALSE), or, alternatively, what fraction of the stem should be assigned to the derived clade. Note that for tip clades stem=FALSE is not allowed.
Value

A modified phylogenetic tree of class "phylo" with the following additional elements:
- maps: a list of named vectors containing the times spent in each state on each branch, in the order in which they occur.
- mapped.edge: a matrix containing the total time spent in each state along each edge of the tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

make.simmap, read.simmap, plotSimmap, sim.history

Description

Primarily internal function for posterior.evolrate; can be used to graft a clade into a receptor tree, at the "sticky tip" labeled with "NA".

Usage

paste.tree(tr1, tr2)

Arguments

- tr1: receptor tree.
- tr2: donor clade.

Details

The donor clade needs to have a root edge, even if it is zero length.

Value

A tree.

Author(s)

Liam Revell <liam.revell@umb.edu>
References


Examples

```r
tr1 <- rtree(10)
tr2 <- rtree(10)
tr1$tip.label[1] <- "NA"
tr2$root.edge <- 0
tr3 <- paste.tree(tr1, tr2)
```

Description

This function simulates stochastic birth-death trees. Simulation can be performed conditioning on \( n \), on \( t \), or on both simultaneously. If the both, then (for optional argument `method="rejection"`) rejection sampling is performed whereby trees are simulated given \( b \) and \( t \) until a tree containing \( n \) taxa is found. The giving-up point can be set using the optional argument `max.count`. Simulations can also be performed in continuous time (the default) or discrete time; the difference being that wait times in the continuous-time simulation come from the exponential distribution; whereas waiting times in discrete-time simulations come from the geometric distribution. In addition, discrete-time simulations allow for the possibility that multiple speciation events can occur at (exactly) the same time, so long as they are on separate branches. Finally, sometimes for stopping criterion \( n \) in discrete-time there will be a number of tips different from \( n \). This indicates that the last event contained more than one speciation event, and a warning is printed.

`method="direct"` is presently experimental. It does not really perform direct sampling; however waiting times & birth or death events are sampled first - with only wait-times consistent with \( n \) and \( t \) being retained. This rejection sampling occurs one layer earlier than for `method="rejection"`. This results in a significant (several-fold) speed-up of the code and enables sampling conditioned on \( n \) and \( t \) simultaneously for much higher \( b \) and \( d \). At the present time, `extant.only=TRUE` does not work for this mode, nor does `type="discrete"`.

Note that if `ape=FALSE`, then the function will run faster, and the tree is theoretically compatible with the ape "phylo" standard; however some downstream errors with functions such as `bind.tree` have been observed.

Usage

```r
pbtree(b=1, d=0, n=NULL, t=NULL, scale=NULL, nsim=1,
      type=c("continuous", "discrete"), ...)
```
Arguments

b  birth rate or speciation rate for type="continuous"; the probability of speciating per time-step for type="discrete".

d  death rate or extinction rate for type="continuous"; the probability of going extinct per time-step for type="discrete".

n  desired number of species (i.e., taxa-stop criterion).

t  total time for simulation (i.e., time-stop criterion).

scale  if set, rescales tree to have total length scale.

nsim  number of simulated trees to return.

type  string to indicate whether to simulate trees in continuous or discrete time. If the former, then wait times between speciation events are drawn from an exponential distribution; whereas if the latter then wait times comes from a geometric distribution.

...  optional arguments including ape, a logical value indicating whether to return nodes in a 'ape' compatible ordering (default is TRUE); extant.only a logical value indicating whether or not to return only extant species (defaults to FALSE); max.count a numeric value indicating the maximum number of iterations to run is sampling conditioned on both n and t (defaults to 1e5); method gives the method used for simultaneously conditioning on n and t - options are "rejection" and "direct"; tip.label, a vector of tip labels (only works for n!=NULL); and, finally, quiet, a logical value indicating whether or not to suppress certain message (defaults to FALSE).

Value

A tree or list of trees as an object of class "phylo" or "multiPhylo", respectively.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


Examples

# simulate a pure-birth tree with 1000 tips, scaled to a length of 1.0
tree<-pbtree(n=1000,scale=1)
# simulate a pure-birth tree conditioning on n & t
tt<-log(50)-log(2)
tree<-pbtree(n=50,t=tt)
**pgls.Ives**

*Phylogenetic regression with intraspecific sampling error*

**Description**

This function fits the phylogenetic regression model with within-species sampling error following Ives et al. (2007).

**Usage**

```r
pgls.Ives(tree, X, y, Vx=NULL, Vy=NULL, Cxy=NULL, lower=c(1e-8,1e-8))
```

**Arguments**

- `tree`: a phylogeny as an object of class "phylo".
- `X`: a named vector containing a single independent variable (multiple independent variables to be added in future). X can contain the species means, or a single long vector containing the sample of values for each species. In the latter case the names(X) will be repeating - all samples from the same species should have the same name.
- `y`: vector the dependent variable. Can be species means or individual values, as for X.
- `Vx`: sampling variances for X. If NULL, then the within-species variance is computed from the data assuming that individual samples, not species means, have been provided in X.
- `Vy`: sampling variances for y. If NULL, then the within-species variance is computed from the data assuming that individual samples, not species means, have been provided in y.
- `Cxy`: sampling covariances between X and y. This will also be computed from the data if Cxy==NULL. Note than in this case - but not for the calculation of Vx and Vy, the same number of observations and the same ordering must be provided for X and y. If this is not the case, then it is assumed that different individuals have been sampled for X and y and thus Cxy is assumed to be zero for all species.
- `lower`: vector specifying the lower bounds for estimation for sig2x and sig2y, respectively. (Must be >0.)

**Details**

Presently only the bivariate regression model is implemented. Uses optim(...,method="L-BFGS-B") for optimization. Note that some problems have been reported with the optimization algorithm for this model, which is simple and thus may fail to find the ML solution.
phenogram

Value

A list with the following components:

- beta: a vector or matrix of regression coefficients.
- sig2x: fitted BM rate for x.
- sig2y: fitted BM rate for y.
- a: fitted ancestral states for x and y.
- logL: log-likelihood.
- convergence: a value for convergence. convergence=0 is good; see optim for more details.
- message: a message for convergence.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

brownie.lite, phylosig, phyl.resid

Description

Function plots a traitgram (Evans et al. 2009), that is, a projection of the phylogenetic tree in a space defined by phenotype (on the y axis) and time (on the x). If a discrete character is mapped on the tree this will also be plotted.

Usage

phenogram(tree, x, fsize=1.0, ftype="reg", colors=NULL, axes=list(), add=FALSE, ...)
Arguments

tree an object of class "phylo", with or without a mapped discrete character.
x a vector containing the states at the tips or the states at all the tips and the internal nodes of the tree.
FSIZE relative font size for tip labels.
ftype font type - options are "reg", "i" (italics), "b" (bold), or "bi" (bold-italics).
colors colors for plotting the mapped character (if available) in tree. If no character is mapped on the tree, then a single color for all the branches of the tree can be provided.
axes list of axis dimensions. Items are time and trait.
add optional logical value indicating whether to add to an open plot. If TRUE, then new axes will not be plotted.
... optional arguments including xlim, ylim, log, main, sub, xlab, ylab, asp, type, lty, lwd, offset, and digits are as in plot.default or par. Note that axes overrides xlim and ylim. spread.labels is a logical value indicating whether or not to minimize tip label overlap (default is TRUE); spread.cost is a numeric vector indicating the relative penalty to be used for label overlap and deviance, respectively (if spread.labels=TRUE); spread.range is the range over which to (potentially) spread the labels - note that if labels do not overlap, not all of that range will be used; finally, link is a numeric value by which to offset the tip labels, linking them to the tips with a dashed line (default is 0, if spread.labels=FALSE, or 10-percent of the total tree length otherwise). The optional argument offsetFudge "fudges" the computation of label offset in scaling xlim. It is 1.37, which is the correct fudge in the Windows R GUI, but this may need to be changed in other systems. hold indicates whether (or not) the output to the graphical device should be held using dev.hold before plotting (defaults to hold=TRUE). quiet suppresses some system messages if set to quiet=TRUE.

Details

For spread.labels=TRUE numerical optimization is performed to optimize the distribution of the labels vertically, where the solution depends on the vector spread.cost containing the cost of overlap (first) and the cost of deviation from the vertical position of the tip. Note that because this is done via numerical optimization, plotting may hang briefly while the best solution is found (especially for large trees).

Value

Plots a traitgram, optionally with a mapped discrete character.

Author(s)

Liam Revell <liam.revell@umb.edu>
References


Examples

tree<-pbtree(n=20,scale=2)
x<-fastBM(tree)
phenogram(tree,x)
# or, simulate a discrete character history
tree<-sim.history(tree,Q=matrix(c(-1,1,1,-1),2,2),anc="1")
# simulate in which the rate depends on the state
x<-sim.rates(tree,c(1,10))
phenogram(tree,x)
# now use spread.labels

tree<-pbtree(n=40)
x<-fastBM(tree)
phenogram(tree,x,spread.labels=TRUE,spread.cost=c(1,0))

phyl.cca

Phylogenetic canonical correlation analysis

Description

This function performs phylogenetic canonical correlation analysis (e.g., Revell & Harrison 2008; Bioinformatics).

Usage

phyl.cca(tree, X, Y, lambda=1.0, fixed=TRUE)

Arguments

tree a phylogenetic tree in "phylo" format.
X a data matrix with traits in columns.
Y data matrix with traits in columns, to be correlated with X.
lambda optionally, a (fixed) value for lambda.
fixed optionally, a logical value indicating whether or not to estimate lambda using likelihood.

Details

(Optional) joint optimization of λ is performed using optimize on the interval (0,1).
Value

A list with the following components:

- **cor**: canonical correlations.
- **xcoef**: coefficients for the canonical variables for X.
- **ycoef**: coefficients for the canonical variables for Y.
- **xscores**: matrix with the canonical scores for X.
- **yscores**: matrix with the canonical scores for Y.
- **chisq**: vector of $\chi^2$ values.
- **p**: P-values for the hypothesis test that the $i$th and all subsequent correlations are zero.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

- `phyl.pca`

---

Description

This function conducts a phylogenetic paired t-test, roughly following Lindenfors et al. (2010; *J. Evol. Biol*.). This is not a phylogenetic ANOVA, in which we want to compare the means of different sets of species on the tree. Instead, we are interested in the difference between two characters, or two measures of a character within a species, and we want to know if this difference is significantly different from zero controlling for the phylogenetic non-independence of species.

Usage

```r
phyl.pairedttest(tree, x1, x2=NULL, se1=NULL, se2=NULL, lambda=1.0, h0=0.0, fixed=FALSE)
```
Arguments

- **tree**: a phylogeny as an object of class "phylo".
- **x1**: data vector for first trait, or matrix with two traits in columns.
- **x2**: data vector for second trait (or null if x1 is a matrix).
- **se1**: standard errors for x1.
- **se2**: standard errors for x2.
- **lambda**: starting value for Pagel’s lambda (or fixed value, if fixed=TRUE).
- **h0**: null hypothesis (to be tested) for the mean difference between x1 and x2.
- **fixed**: logical value specifying whether or not to optimize lambda.

Details

Likelihood optimization is performed using `optim` with method="L-BFGS-B" with box constraints on lambda (0,1).

Value

A list with the following components:

- **d-bar**: phylogenetic mean difference.
- **se**: standard error of d-bar.
- **sig2**: estimated evolutionary variance (of the difference).
- **lambda**: fitted (or fixed) value of lambda.
- **logL**: log-likelihood of the fitted model.
- **t.d-bar**: t-value ((d-bar-h0)/se where se is computed from the Hessian).
- **P.d-bar**: P-value.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


Description

This function performs phylogenetic PCA (e.g., Revell 2009, *Evolution*).

Usage

```r
phyl.pca(tree, Y, method="BM", mode="cov", ...)  
## S3 method for class 'phyl.pca'
biplot(x, ...)
```

Arguments

- **tree**: phylogeny as an object of class "phylo".
- **Y**: data matrix with traits in columns.
- **method**: method to obtain the correlation structure: can be "BM" or "lambda".
- **mode**: is the mode for the PCA: can be "cov" or "corr".
- **x**: object of class "phyl.pca" for `biplot.phyl.pca`
- **...**: for S3 plotting method `biplot.phyl.pca`, other arguments to be passed to `biplot`.

Details

If method="lambda" \( \lambda \) is optimized on the interval (0,1) using `optimize`. S3 methods (print, summary, and `biplot`) are modified from code provided by Joan Maspons and are based on the same methods for objects of class "prcomp". Function `biplot` now permits the argument choices to be supplied, which should be a vector of length two indicated the two PC axes to be plotted.

Value

An object of class `phyl.pca` which is a list with some or all of the following components:

- **Eval**: diagonal matrix of eigenvalues.
- **Evec**: matrix with eigenvectors in columns.
- **S**: matrix with scores.
- **L**: matrix with loadings.
- **lambda**: fitted value of lambda (method="lambda" only).
- **logL**: log-likelihood for lambda model (method="logL" only).

Author(s)

Liam Revell <liam.revell@umb.edu>, Joan Maspons
References


See Also

`phyl.cca, phyl.resid`

---

**Description**

This function fits one or multiple phylogenetic regressions (depending on the number of columns in `y`) and computes the residuals. Designed for phylogenetic size correction using GLS regression (e.g., Revell 2009: *Evolution*).

**Usage**

`phyl.resid(tree, x, Y, method="BM")`

**Arguments**

- `tree` a phylogenetic tree in "phylo" format.
- `x` vector containing the single independent variable (e.g., size), or matrix with multiple independent variables in columns.
- `Y` vector or matrix with one or multiple dependent variables in columns.
- `method` method to obtain the correlation structure: can be "BM" or "lambda".

**Details**

Optionally fits \( \lambda \) for each regression model. Likelihood optimization of \( \lambda \) is performed for `method="lambda"` using `optimize` on the interval (0,1). This function is redundant with `residuals(gls(\ldots,correlation=corPagel(\ldots)))`, but some users may find this method simpler, and it provides a good way to cross-check your results & make sure that you are using `gls` correctly.

**Value**

A list with the following components:

- `beta` a vector or matrix of regression coefficients.
- `resid` a vector or matrix of residuals for species.
- `lambda` a vector of lambda values (`method="lambda"` only).
- `logL` a vector of log-likelihoods (`method="lambda"` only).
**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

`phyl.pca`, `gls`

---

### phyl.RMA

*Phylogenetic reduced major axis (RMA) regression*

**Description**

This function performs phylogenetic RMA regression.

**Usage**

```r
call = phyl.RMA(x, y, tree, method="BM", lambda=NULL, fixed=FALSE, h0=1.0)
```

**Arguments**

- `x`: vector with names.
- `y`: vector with names.
- `tree`: a phylogenetic tree in “phylo” format.
- `method`: method to obtain the correlation structure: can be "BM" or "lambda".
- `lambda`: value of lambda for fixed \( \lambda \).
- `fixed`: logical value indicating whether or not \( \lambda \) should be optimized using likelihood.
- `h0`: null hypothesis for beta. Defaults to 1.0. Note that a null hypothesis of 0.0 is not allowed.

**Details**

Optionally jointly estimates lambda if `method="lambda"`. Likelihood optimization of lambda is performed using `optimize` on the interval (0,1). Note that some statistician think there is *never* a condition in which a reduced-major-axis regression should be used.
Value

A list with the following components:

- **RMA.beta**: a vector of RMA regression coefficients.
- **V**: a VCV matrix for the traits.
- **lambda**: fitted value of lambda (method="lambda" only).
- **logL**: log-likelihood (method="lambda" only).
- **test**: a vector containing results for hypothesis tests on beta.
- **resid**: a vector of residuals for y given x.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

- `phyl.cca`, `phyl.pca`, `phyl.resid`

Description

Primarily an internal function for `phyl.pca`; this can be used to compute the phylogenetic trait variance-covariance matrix given a phylogenetic VCV, lambda, and a data matrix. Should not be confused with `vcv.phylo` in the "ape" package (although one of the objects returned is the output of `vcv.phylo`).

Usage

`phyl.vcv(X, C, lambda)`

Arguments

- **lambda**: value for λ transformation.
- **X**: data matrix.
- **C**: matrix containing the height above the root of each pair of species in the tree. Typically this will have been produced by calling `vcv.phylo`.
phylanova

Details

Do not use unless you know what you’re doing.

Value

A list containing three elements, as follows: C, the matrix \( vcv\oper({phylo}) \) transformed by \( \lambda \); R, the among trait variance-covariance matrix for the data in \( X \); and alpha, a vector of ancestral states at the root node of the tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


Description

This function performs the simulation-based phylogenetic ANOVA of Garland et al. (1993) and (optionally) conducts all posthoc comparisons of means among groups (also obtaining the P-values by phylogenetic simulation).

Usage

\texttt{phylanova(tree, x, y, nsim=1000, posthoc=TRUE, p.adj="holm")}

Arguments

- \texttt{tree} a phylogenetic tree in "phylo" format.
- \texttt{x} a vector containing the groups.
- \texttt{y} a vector containing the response variable (continuously valued).
- \texttt{nsim} an integer specifying the number of simulations (including the observed data).
- \texttt{posthoc} a logical value indicating whether or not to conduct posthoc tests to compare the mean among groups.
- \texttt{p.adj} method to adjust P-values for the posthoc tests to account for multiple testing. Options same as \texttt{p.adjust}.

Details

Uses a little bit of code from phy.anova in the "geiger" package as well as \texttt{pairwise.t.test}.
Value

A list containing the following elements:

- F  
  F from observed data.
- Pf  
  P-value for F from simulation.
- T  
  matrix of t-values.
- Pt  
  matrix of multiple test corrected P-values from posthoc t-tests.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

anova, pairwise.t.test

---

**phylo.to.map**  
Plot tree with tips linked to geographic coordinates

**Description**

Function plots a tree and tips pointing to coordinates on a global map.

**Usage**

```r
phylo.to.map(tree, coords, rotate=TRUE, ...)  
# S3 method for class 'phylo.to.map'
plot(x, type=c("phylogram","direct"), ...)
```

**Arguments**

- `tree`  
an object of class "phylo".
- `coords`  
a matrix containing the latitude (in column 1) and the longitude of all tip species in the tree. The row names should be the same as `tree$tip.label`.
- `rotate`  
a logical value indicating whether or not to rotate nodes of the tree to better match longitudinal positions.
for plot.phylo.to.map, an object of class "phylo.to.map".

type a string indicating whether to map the tips of the tree onto a geographic map
from a square phylogram (type="phylogram") or to project the tree directly
onto the map (type="direct").

... optional arguments. For phylo.to.map, which creates an object of class "phylo.to.map"
and (optionally) plots that object, optional arguments include: database and
regions (see map); xlim and ylim, which control the plot area for the map;
fs for the font size of plot labels; split which controls the proportion of
vertical space for the tree (first) and map, in a vector; psize the size of the
plotted points on the map; mar and asp (see par. For plot.phylo.to.map, the
options are the same - excluding database and regions.

Value

Plots a phylogeny and map and returns an object of class "phylo.to.map" invisibly.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


Description

Converts tree to backbone or vice versa

Usage

phylo.toBackbone(x, trans, ...)
backbone.toPhylo(x)

Arguments

x an object of class "phylo" (for phylo.toBackbone); or an object of class backbone.toPhylo
(for backbone.toPhylo).

trans data frame containing the attributes necessary to translate a backbone tree to an
object of class "backbonePhylo". The data frame should contain the following
variables: tip.label: the tip labels in the input tree (not all need be included);
clade.label: labels for the unobserved subtrees; N: number of species in each
subtree; and depth: desired depth of each subtree. depth for each terminal
taxon in x cannot be greater than the terminal edge length for that taxon.

... optional arguments.
phyloDesign

Value
Either an object of class "phylo" or an object of class "backbonePhylo", depending on the method.

Author(s)
Liam Revell <liam.revell@umb.edu>

References

See Also
plot.backbonePhylo

Description
Primarily an internal function for optim.phylo.ls, this function creates a design matrix for least squares phylogenetic analysis.

Usage
phyloDesign(tree)

Arguments
tree    phylogenetic tree.

Details
This function returns a matrix containing the edges in the tree (in columns) and pairs of tip node numbers (in rows). Values in the matrix are either 1 if the edge is on the shortest path between the two tips; and 0 otherwise. Probably do not use unless you know what you’re doing.

Value
A matrix.

Author(s)
Liam Revell <liam.revell@umb.edu>
phylomorphospace

Creates phylomorphospace plot

Description

This function creates a phylomorphospace plot (a projection of the tree into morphospace) for two characters following Sidlauskas (2008; Evolution). It will also plot a discrete character mapped on tree.

Usage

phylomorphospace(tree, X, A=NULL, label=c("radial","horizontal","off"), control=list(), ...)

Arguments

tree

a phylogenetic tree in "phylo" format, or a modified "phylo" object with a mapped discrete character.

X

an n x 2 matrix of tip values for two characters in n species.

A

an optional m x 2 matrix (for m nodes) of values for two traits at internal nodes in the tree - if not supplied, these values will be estimated using fastAnc.

label

string indicating whether to plot the tip labels in the same direction as the terminal edge (label="radial"), horizontally (label="horizontal"), or not at all "off". label=TRUE and label=FALSE are also acceptable, for compatibility with phytools <= 0.3-03.

control

a list containing the following optional control parameters: col.edge: a vector of edge colors; and col.node: a vector of node colors.

... optional arguments for plotting, including xlim, ylim, xlab, ylab, lwd, colors, fsize, and node.by.map. colors is only used when there is a mapped discrete character on the tree, in which case control$col.edge is ignored. fsize is relative to the default, which is textxy(...,cx=0.75). node.by.map is a logical value (default to FALSE which tells the function whether or not to plot the node colors using the colors of the mapped discrete character. Setting this option to TRUE will cause control$col.node to be ignored. node.size is a vector containing the point size relative to the default (see par for plotted internal nodes and tips, respectively. Defaults to node.size=c(1,1,3). If only one number is provided it will be recycled. axes is a logical value indicating whether or not axes should be plotted (see plot.default). Finally, add indicates whether to add the phylmorphospace to the current plot.

References

Value

This function creates a phylomorphospace plot

Author(s)

Liam Revell <liam.revell@umb.edu>

References


Examples

tree<-pbtree(n=25)
X<-fastBM(tree,nsim=2)
phylomorphospace(tree,X,xlab="trait 1",ylab="trait 2")

---

**phylomorphospace3d**

*Creates tree-dimensional phylomorphospace plot*

Description

This function creates a phylomorphospace plot for three characters using the 3D visualization package, 'rgl'.

Usage

```
phylomorphospace3d(tree, X, A=NULL, label=TRUE, control=list(),
method=c("dynamic","static"), ...)
```

Arguments

tree a phylogenetic tree in "phylo" format.
X an n x 3 matrix of tip values for two characters in n species.
A an optional m x 3 matrix (for m nodes) of values for two taits at internal nodes in the tree - if not supplied, these values will be estimated using `anc.ML`.
label logical value indicating whether to print tip labels next to terminal nodes in the plot (presently doesn’t do anything, but labels can be dropped using control).
spin=TRUE,axes=TRUE,box=TRUE,simple.axes=FALSE,lwd=1,ftype="reg"
control

A list containing the following optional control parameters:

- `spin`: a logical value indicating whether to animate the plot when created;
- `axes`: a logical indicating whether to plot the axes;
- `box`: a logical value indicating whether to plot in box;
- `simple.axes`: logical value indicating whether to replace box and axes with simpler axes;
- `lwd`: line widths;
- `ftype`: font type ("off" turns off labels altogether);
- `col.edge`: a vector of colors of length `nrow(tree$edge)`.

method

A string either "dynamic" for a dynamic (animated) plot created using `rgl`; or "static" for a flat 3D plot created using `scatterplot3d` and base graphics. The latter has the advantage of being very easy to export in standard format.

... optional arguments to be passed to `scatterplot3d`. Most options not available. angle is an important option that does work here.

Value

This function creates a three dimensional phylomorphospace plot. The function returns a function from `spin3d` (for method="dynamic"); or a series of functions from `scatterplot3d` (for method="static").

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

`fancytree`, `phenogram`, `phylomorphospace`

Examples

tree<-.pbtree(n=26,tip.label=LETTERS)
X<-.fastBM(tree,nsim=3)
## Not run:
phylomorphospace3d(tree,X,control=list(spin=FALSE))

## End(Not run)
phylomorphospace3d(tree,X,method="static")
Compute phylogenetic signal with two methods

Description

This function computes phylogenetic signal using two different methods. It can also conduct the hypothesis tests for significant phylogenetic signal, and estimate phylogenetic signal incorporating sampling error following Ives et al. (2007).

Usage

phylosig(tree, x, method="K", test=FALSE, nsim=1000, se=NULL, start=NULL, control=list())

Arguments

tree  
a phylogenetic tree in "phylo" format.
x  
vector containing values for a single continuously distributed trait.
method  
method to compute signal: can be "K" or "lambda".
test  
logical indicating whether or not to conduct a hypothesis test of "K" or "lambda".
nsim  
for method="K", number of simulations in randomization test.
se  
named vector containing the standard errors for each species.
start  
vector of starting values for optimization of (respectively) sigma^2 and lambda. Only used in method="lambda" and se!=NULL.
control  
list of control parameters for multidimensional optimization, implemented in optim. Only used in method="lambda" and se!=NULL.

Details

A optimization is performed using optimize with the range of lambda set between 0 and the theoretical upper limit of lambda (determined by the relative height of the most recent internal node on the tree).

Value

If (method="K"), a list with the following components:

K  
value of the K-statistic.
sig2  
rate of evolution, σ^2, for estimation with sampling error.
logL  
log-likelihood, for estimation with sampling error.
P  
optionally, the P-value from the randomization test.

If (method="lambda"), a list with the following components:

lambda  
fitted value of lambda.
plot.backbonePhylo

<table>
<thead>
<tr>
<th>sig2</th>
<th>rate of evolution, for estimation with sampling error.</th>
</tr>
</thead>
<tbody>
<tr>
<td>logL</td>
<td>log-likelihood.</td>
</tr>
<tr>
<td>logL0</td>
<td>log-likelihood for lambda=0.0.</td>
</tr>
<tr>
<td>P</td>
<td>P-value of the likelihood ratio test.</td>
</tr>
<tr>
<td>convergence</td>
<td>value for convergence, for estimation with sampling error only. (See optim).</td>
</tr>
<tr>
<td>message</td>
<td>message from optim, for estimation with sampling error only.</td>
</tr>
</tbody>
</table>

Author(s)
Liam Revell <liam.revell@umb.edu>

References

Examples
```r
tree <- pbtree(n=100)
x <- fastBM(tree)
phylosig(tree,x,method="lambda",test=TRUE)
```

plot.backbonePhylo Plots backbone tree with triangles as clades

Description
Function plots a backbone tree (stored as an object of class "backbonePhylo") with triangles as subtrees.

Usage
```r
## S3 method for class 'backbonePhylo'
plot(x, ...)  
```

Arguments
- **x**
  - an object of class "backbonePhylo".
- **...**
  - optional arguments. So far includes only vscale, which is used to rescale the vertical dimension in plotting.
Value

Plots a tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

phylo.toBackbone

Examples

```r
## first create our backbone tree with
## random subtree diversities
tree <- phytools:::lambdatree(pmtree(n=10), lambda=0.5)
## for old versions of geiger, use lambdatree
## create a translation table
## leaving a couple of single-taxon clades for fun
tip.label <- sample(tree$tip.label, 8)
clade.label <- LETTERS[1:8]
N <- ceiling(runif(n=8, min=1, max=20))
## set crown node depth to 1/2 the maximum depth
depth <- sapply(tip.label, function(x) 0.5*y$edge.length[which(tree$edge[, 2] == which(y$tip.label == x))], y = tree)
trans <- data.frame(tip.label, clade.label, N, depth)
rownames(trans) <- NULL
rm(tip.label, clade.label, N, depth)
## here's what trans looks like
trans
## convert
tt <- phylo.toBackbone(tree, trans)
## plot
plot(tt)
```

---

**plotBranchbyTrait**  
*Plot branch colors by a quantitative trait or value*

Description

Function plots a tree with branches colored by the value for a quantitative trait or probability, by various methods. Unlike most other tree plotting functions in phytools, this function calls `plot.phylo` (not `plotSimmap`) internally.
Usage

plotBranchbyTrait(tree, x, mode=c("edges","tips","nodes"), palette="rainbow", legend=TRUE, xlims=NULL, ...)

Arguments

tree an object of class "phylo".

x either a vector of states for the edges, tips, or nodes of the tree (for mode="edges", "tips", and "nodes", respectively).

mode string indicating plotting mode. mode="edges", the default, requires that the mapping state of each edge in the tree should be provided. mode="tips" takes the tip values and estimates the state at each internal node. The mapped character value along each branch is the average of the nodes subtending that branch. mode="nodes" similar to "tips", except that the node values are provided instead of estimated.

palette color palette to translate character values to color. Options are presently "rainbow" (the default), "heat.colors", and "gray".

legend can be a logical value (TRUE or FALSE) or a numeric value greater than 0. In the latter case the numeric value gives the length of the plotted legend, which also acts as a scale bar for the branch lengths of the tree.

xlims range for the translation map between trait values and the color map. Should be inclusive of all the values in x.

... other optional arguments to be passed to plot.phylo - pretty much all arguments are available. In addition, there plotBranchbyTrait has the following additional optional arguments: tol a small tolerance value to be added to the range of x; prompt for legend=TRUE, a logical value indicating whether to prompt for the position of the legend (or not) - the default is to put the legend in the lower left hand size of the plot; title for legend=TRUE, the title of the legend; and digits for legend=TRUE, the number of digits in the quantitative scale of the legend.

Details

Note that if prompt=TRUE, the function will prompt for the position of the legend.

Value

Plots a phylogeny.

Author(s)

Liam Revell <liam.revell@umb.edu>

References

**plotSimmap**

*Plot stochastic character mapped tree*

**Description**

Function plots one or multiple stochastic character mapped trees.

**Usage**

```
plotSimmap(tree, colors=NULL, fsize=1.0, ftype="reg", lwd=2, pts=FALSE,
node.numbers=FALSE, mar=NULL, add=FALSE, offset=NULL, direction="rightwards",
type="phylogram", setEnv=TRUE, part=1.0, xlim=NULL, ylim=NULL,
node.numbers="intermediate", tips=NULL, maxY=NULL, hold=TRUE)
```

**Arguments**

- **tree**: a modified object of class "phylo" or "multiPhylo" containing a stochastic mapping or set of mappings (e.g., see `read.simmap` & `make.simmap`).
- **colors**: a vector with names translating the mapped states to colors - see Examples.
- **fsize**: relative font size for tip labels.
- **ftype**: font type - options are "reg", "i" (italics), "b" (bold), or "bi" (bold-italics).
- **lwd**: line width for plotting.
- **pts**: logical value indicating whether or not to plot filled circles at each vertex of the tree, as well as at transition points between mapped states. Default is FALSE.
- **node.numbers**: a logical value indicating whether or not node numbers should be plotted.
- **mar**: vector containing the margins for the plot to be passed to `par`. If not specified, the default margins are [0.1,0.1,0.1,0.1].
- **add**: a logical value indicating whether or not to add the plotted tree to the current plot (TRUE) or create a new plot (FALSE, the default).
- **offset**: offset for the tip labels. Primarily to be used internally by `densityMap`.
- **direction**: plotting direction. Options are "rightwards" (the default) and "leftwards". Note that for some reason that is not totally clear, node.numbers=TRUE does not work for direction="leftwards".
- **type**: plot type. Can be "phylogram" or "fan". Only a subset of options are presently available for type="fan".
- **setEnv**: logical value indicating whether or not to set the environment .PlotPhyloEnv. Setting this to TRUE will allow (so far, only for type="phylogram" compatibility with ape function nodelabels.
- **part**: value between 0 and 1 for type="fan" indicating what fraction of the full circular tree to use as plotting area. For instance, part=0.5 will plot a half fan phylogeny. It also affects the axis scaling used.
- **xlim**: x-limits for the plot.
ylim  y-limits for the plot.
nodes  node placement following Felsenstein (2004; pp. 574-576). Can be "intermediate", "centered", "weighted", or "inner". So far only works for type="phylogram".
tips  labeled vector containing the vertical position of tips. Normally this will be 1:N for N tips in the tree.
maxY  maximum value of y to use before rotating a tree into fan configuration. This will only make a difference if different from Ntip(tree).
hold  logical argument indicating whether or not to hold the output to the graphical device before plotting. Defaults to hold=TRUE.

Details
The underscore character "_" is automatically swapped for a space in tip labels, as in plot.phylo.

Value
Plots a tree.

Author(s)
Liam Revell <liam.revell@umb.edu>

References

See Also
densityMap, make.simmap, read.simmap

Examples
# simulate a mapped tree
Q<--matrix(c(-2,1,1,-2,1,1,-2),3,3)
rownames(Q)<-colnames(Q)<-letters[1:3]
tree<-sim.history(pbtree(n=100,scale=1),Q)
cols<-setNames(c("blue","red","green"),letters[1:3])
# plot the mapping
plotSimmap(tree,cols,ftype="i",fsize=0.7)
plotThresh

Description

This function uses the object returned by \texttt{ancThresh} to plot the posterior probabilities of ancestral states under the threshold model. It is also called internally by \texttt{ancThresh}.

Usage

\begin{verbatim}
plotThresh(tree, x, mcmc, burnin=NULL, piecol, tipcol="input", legend=TRUE, ...)
\end{verbatim}

Arguments

\begin{itemize}
\item \texttt{tree} \hspace{1cm} phylogenetic tree.
\item \texttt{x} \hspace{1cm} a named vector containing discrete character states; or a matrix containing the tip species, in rows, and probabilities of being in each state, in columns.
\item \texttt{mcmc} \hspace{1cm} list object returned by \texttt{ancThresh}.
\item \texttt{burnin} \hspace{1cm} number of generations (not samples) to exclude as burn in; if \texttt{NULL} then 20 percent of generations are excluded as burn-in.
\item \texttt{piecol} \hspace{1cm} a named vector containing the colors for the posterior probabilities plotted as pie charts at internal nodes.
\item \texttt{tipcol} \hspace{1cm} a string indicating whether the tip colors should be based on the input data ("input") or sampled tip liabilities ("estimated"). These will only differ if there is uncertainty in the tip states.
\item \texttt{legend} \hspace{1cm} logical value or text to be plotted in the legend.
\item ... \hspace{1cm} other arguments to be passed to \texttt{plot.phylo} - \texttt{label.offset} should be >0 so that tip labels and species names do not overlap.
\end{itemize}

Value

Plots a tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


plottree

See Also

ancThresh, plot.phylo

---

plottree

Plots rooted phylogenetic tree

Description

This function plots a rooted phylogram. Arguments in ... are passed to `plotSimmap`, with the exception of optional argument color which is used to determine the plotted color of the branch lengths of the tree.

Usage

`plottree(tree, ...)`

Arguments

- `tree` a phylogenetic tree in "phylo" format; or multiple trees as an object of class "multiPhylo".
- `...` optional arguments.

Value

This function plots a rooted phylogram.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

`plot.phylo`, `plotSimmap`

Examples

```r
  tree<-pbtree(n=25)
  plottree(tree, color="blue", ftype="i")
```
plotTree.wBars  Plot a tree with bars at the tips

Description

Function plots a phylogeny in phylogram or fan style with bars at the tips representing the values for a phenotypic trait.

Usage

plotTree.wBars(tree, x, scale=1, width=NULL, type="phylogram", method="plotTree", tip.labels=FALSE, ...)

Arguments

tree  an object of class "phylo".
x  a named vector of trait values (normally > 0).
scale  scaling factor for the tip bars (relative to the total tree height).
width  width of the tip bars.
type  plot type. Can be "phylogram" or "fan".
method  plotting method to use. Can be "plotTree" (for plotTree) or "plotSimmap" (for plotSimmap).
tip.labels  argument indicating whether or not tip labels should be plotted. Defaults to tip.labels=FALSE.
...  optional arguments to be passed to codeplotTree or plotSimmap.

Value

Plots a tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

plotSimmap, plotTree
posterior.evolrate  Analysis of the posterior sample from evol.rate.mcmc

Description

This function takes a phylogenetic tree, an average split position, and a raw MCMC output from evol.rate.mcmc and returns a posterior sample of evolutionary rates rootward ($\sigma(1)^2$) and tipward ($\sigma(2)^2$) from the average split.

Usage

posterior.evolrate(tree, ave.shift, mcmc, tips, showTree=FALSE)

Arguments

tree         a phylogenetic tree in "phylo" format.
ave.shift    mean or median shift-point from the posterior sample (see minSplit).
mcmc         matrix $mcmc$ from evol.rate.mcmc (probably with burnin excluded).
tips         list of tips in state sig(1)^2 for each sampled generation of MCMC.
showTree     optional logical value indicating whether or not to plot the stretched and shrunken tree generated by the pre-processing algorithm implemented in this function (default is FALSE).

Value

A matrix containing the posterior sample of evolutionary rates and shift-points between rates.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

evol.rate.mcmc, minSplit
**print.backbonePhylo**  \(\text{Print method for backbone phylogeny}\)

**Description**

Print method for an object of class "backbonePhylo".

**Usage**

```r
## S3 method for class 'backbonePhylo'
print(x, ...)
```

**Arguments**

- `x`: an object of class "backbonePhylo".
- `...`: optional arguments.

**Value**

Prints to screen.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

- `phylo.toBackbone`

---

**ratebystate**  \(\text{Method for investigating the rate of one trait as a function of the state of another}\)

**Description**

This function attempts to ask if the rate of a continuous character, \(y\), depends on the state of a separate continuous trait, \(x\). This is accomplished by regressing the squared contrasts in \(y\) on the branch or node ancestral estimates of \(x\).
ratebystate

Usage

ratebystate(tree, x, y, nsim=100, corr=c("pearson","spearman"), ...)

Arguments

tree              phylogenetic tree.
x                a continuous character - the dependent variable in the model.
y                a second continuous trait - the response variable.
nsim              number of simulations for hypothesis testing.
corr              correlation method to use. Same as in cor.
...  optional arguments which include sim.method ("fastBM" or "sim.corrs"; see fastBM and sim.corrs); method ("by.node" or "by.branch" indicating whether to assume the rate varies as a function of the node state or the mean branch state); message - a logical value indicating whether or not to return corr and method; finally logarithm - indicating whether or not to fit a model in which the variance of Brownian evolution in y changes as a multiplicative function of x. The default is logarithm=FALSE.

Value

This function returns a list with up to the following four elements:

beta          value of the regression coefficient for square of the contrasts in y regressed on the ancestral or branch-wise estimated states for x.
r               correlation coefficient for corr=corr.
corr          string giving the value of corr.
method        string giving the value of method.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

fastAnc.pic
rateshift

Find the temporal position of one or more rate shifts

Description

Function finds the location of one or more rate shifts.

Usage

rateshift(tree, x, nrates=1, niter=10, ...)

Arguments

tree  object of class "phylo".

x      vector of phenotypic trait values for species. names(x) should contain the species names and match tree$tip.label.

nrates number of rates.

niter  number of iterations of optimization routine to ensure convergence.

...   optional arguments.

Value

An object of class "rateshift".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

brownie.lite
**read.newick**  
*Robust Newick style tree reader*

**Description**

This function reads a Newick style tree from file.

**Usage**

```r
tree <- "((Human,Chimp),Gorilla),Monkey;"
phy <- read.newick(text = tree)
```

**Arguments**

- `file` name of text file with single Newick style tree or multiple trees, one per line.
- `text` character string containing tree.

**Details**

This function is almost completely redundant with `read.tree`; however it is 'robust' in that it does not fail if the tree contains so-called 'singles' (nodes with only one descendant).

**Value**

An object of class "phylo", possibly containing singles (see `collapse.singles`).

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

`read.tree, read.nexus`

**Examples**

```r
tree <- "((Human,Chimp),Gorilla),Monkey;"
phy <- read.newick(text = tree)
```
read.simmap  

Read SIMMAP style trees from file

Description

This reads one or multiple SIMMAP style trees from file.

Usage

read.simmap(file='', text, format="nexus", rev.order=TRUE, version=1)

Arguments

file  
name of text file with one or multiple SIMMAP v1.0 or v1.5 style trees.

text  
character string containing the tree. If version=1.5 this argument is ignored. (This format tree can only be read from file in the present version.)

format  
format of the trees: either "phylip" or "nexus" - the latter is the default output from SIMMAP. If version=1.5 this argument is ignored.

rev.order  
a logical value indicating whether the states and times along each branch is given (from root to tip) in right-to-left order (if TRUE) or in left-to-right order. If version=1.5 this argument is ignored.

version  
version of SIMMAP for input tree. If the tree(s) was/were simulated in SIMMAP v1.0 or written to file by \link{make.simmap} then version=1.0; if the tree(s) was/were simulated using SIMMAP v1.5 then version=1.5.

Details

This function now accepts trees in both SIMMAP v1.0 and SIMMAP v1.5 format. In addition, it can read a more flexible format than is produced by SIMMAP (for instance, multi-character mapped states and more than 7 mapped states). Uses some modified code from \code{read.nexus} from the "ape" package to read the NEXUS block created by SIMMAP. Also creates the attribute "map.order" which indicates whether the stochastic map was read in from left to right or right to left. This attribute is used by default by \code{write.simmap} to write the tree in the same order.

Value

A modified object of class "phylo" (or list of class "multiPhylo") with the following additional elements:

maps  
a list of named vectors containing the times spent in each state on each branch, in the order in which they occur.

mapped.edge  
a matrix containing the total time spent in each state along each edge of the tree.

Author(s)

Liam Revell <liam.revell@umb.edu>
References


See Also

`brownie.lite`, `evol.vcv`, `read.tree`, `read.nexus`

---

`reorder.backbonePhylo`  
*Reorders a backbone phylogeny*

Description

Function reorders an object of class "backbonePhylo".

Usage

```r
## S3 method for class 'backbonePhylo'
reorder(x, order="cladewise", ...)
```

Arguments

- `x`:
  - an object of class "backbonePhylo".
- `order`:
  - order. See `reorder.phylo` for possible orderings.
- `...`:
  - optional arguments.

Value

An object of class "backbonePhylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

`phylo.toBackbone`
reorderSimmap  

Reorder edges of a simmap tree

Description

Function returns a reordered modified "phylo" object by using reorder.phylo but then sorting the additional elements $mapped.edge and $maps to have the same order as $edge.

Usage

reorderSimmap(tree, order="cladewise")

Arguments

tree  
a modified object of class "phylo".

order  
either "cladewise" or "pruningwise" (see reorder.phylo).

Value

A modified object of class "phylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

reorder.phylo, plotSimmap

rep.phylo  

Replicate a tree or set of trees

Description

S3 method rep for object of class "phylo" or "multiPhylo". repPhylo is just an alias for rep.phylo and rep.multiPhylo.
reroot

Usage

reroot(tree, node.number, position)

Arguments

tree: object of class "phylo".
node.number: a numeric value.
position: a numeric value.

Value

An object of class "multiPhylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

c.phylo, rep

Examples

tree<-pbtree(n=100)
trees<-rep(tree,100)

reroot tree along an edge

Description

This function re-roots a phylogenetic tree at an arbitrary position along an edge.

Usage

reroot(tree, node.number, position)
### Arguments

- **tree**: A phylogenetic tree in "phylo" format.
- **node.number**: Number of the node descending from the target branch in `tree$edge` - this can also be a tip in which case the node number is the index number of the tip in `tree$tip.label`.
- **position**: Position along the target edge at which to re-root the tree.

### Details

This function had an error for rootings along edges descended from the root node for `phytools<=0.2-47`. This should be fixed in the present version. Now uses `paste.tree.root` and `splitTree` internally. Earlier versions also had an error related to node labels. This should be fixed in `phytools>=0.4-47`.

### Value

A phylogenetic tree in "phylo" format.

### Author(s)

Liam Revell <liam.revell@umb.edu>

### References


### See Also

- `splitTree`
- `paste.tree`
- `root`

---

### Description

This function uses the re-rooting method of Yang et al. (1995) to get the marginal ancestral state estimates for each internal node of the tree using likelihood. This method get the conditional scaled likelihoods for the root node (which is the same as the marginal ancestral state reconstruction for that node) and successively moves the root to each node in the tree. The function can also return the posterior probabilities for the tip nodes of the tree.

### Usage

```r
erootingMethod(tree, x, model=c("ER","SYM"), ...)
```
rerootingMethod

Arguments

- **tree**: an object of class "phylo".
- **x**: a vector of tip values for species, or a matrix containing the prior probability that the tip is in each state. If `x` is a vector, then `names(x)` should be the species names. If `x` is a matrix of prior probabilities, then rownames should be species names, column names should be states for the discrete character, and rows of the matrix should sum to 1.0.
- **model**: any reversible model. `model=c("ER","SYM")` recommended.
- **...**: optional arguments. Presently the logical argument `tips`. If `tips=TRUE`, then the function will also compute the empirical Bayes posterior probabilities of the tips following Yang (2006).

Details

This function calls code modified from `ace` in the (Paradis et al. 2004) internally in the calculation of normalized conditional likelihoods.

Value

A list containing the following elements:

- **loglik**: the log-likelihood.
- **Q**: the fitted transition matrix between states.
- **marginal.anc**: the marginal ancestral state reconstructions for each node (and, optionally, each tip).

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

`ace`, `make.simmap`
rescaleSimmap  

Rescale SIMMAP style tree

Description

This function scales a tree with a mapped discrete character to an arbitrary total height, preserving the relative time spent in each state along each edge.

Usage

rescaleSimmap(tree, ...)

Arguments

tree a phylogenetic tree in modified "phylo" format with a discrete character mapping (e.g., see read.simmap or make.simmap).

... other arguments, such as depth.

Details

Replaces rescaleTree (now rescale.phylo) in the 'geiger' package for SIMMAP style trees.

Value

A phylogenetic tree in modified "phylo" format.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

make.simmap, read.simmap
rotateNodes  

Rotates a node or set of nodes in a phylogenetic tree

Description

This function is a wrapper for rotate which rotates a set of nodes or all nodes.

Usage

rotateNodes(tree, nodes, polytom=c(1,2), ...)

Arguments

tree  object of class "phylo".

nodes  either a single node number to rotate, a vector of node numbers, or the string "all".

polytom  a vector of mode numeric and length two specifying the two clades that should be exchanged in a polytomy (see rotate).

...  optional arguments.

Details

Also addresses the problem that the product of multiple rotations from rotate can be non-compliant with the implicit "phylo" standard because the tip numbers in tree$edge are not in numerical order 1:n for n tips.

Value

An object of class "phylo" (i.e., a phylogenetic tree).

Author(s)

Liam Revell <liam.revell@umb.edu>

References

roundBranches  Rounds the branch lengths of a tree

Description

This function rounds the branch lengths of a tree or trees and reconciles any mappings (as in \texttt{read.simmap}) with the rounded branch lengths.

Usage

\texttt{roundBranches(tree, digits)}

Arguments

- \textit{tree}  an object of class "phylo" or "multiPhylo".
- \textit{digits} number of digits for rounding. Passed to \texttt{round}.

Value

A tree with branch lengths, or modified "phylo" or "multiPhylo" object with a mapped discrete character.

Author(s)

Liam Revell \texttt{<liam.revell@umb.edu>}

References


roundPhylogram  Plot a round phylogram

Description

Function plots one or multiple round phylograms.

Usage

\texttt{roundPhylogram(tree, fsize=1.0, ftype="reg", lwd=2, mar=NULL, offset=NULL, direction="rightwards", type="phylogram", xlim=NULL, ylim=NULL)}
**Arguments**

- **tree**: an object of class "phylo" or "multiPhylo" containing one or multiple phylogenies.
- **fsize**: relative font size for tip labels.
- **ftype**: font type - options are "reg", "i" (italics), "b" (bold), or "bi" (bold-italics).
- **lwd**: line width for plotting.
- **mar**: vector containing the margins for the plot to be passed to `par`. If not specified, the default margins are [0.1,0.1,0.1,0.1].
- **offset**: offset for the tip labels.
- **direction**: plotting direction. Only the option `direction="rightwards"` is presently supported.
- **type**: plot type. Can be "phylogram" or "cladogram". If `type="cladogram"` then the branch lengths are not necessary (and, indeed, are not used).
- **xlim**: x-limits for the plot.
- **ylim**: y-limits for the plot.

**Details**

The underscore character "_" is automatically swapped for a space in tip labels, as in `plotSimmap`.

**Value**

Plots a tree.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

`plotSimmap`, `plotTree`
rstate

Pick a random state according to a vector of probabilities

Description

Primarily an internal function for make.simmap.

Usage

rstate(y)

Arguments

y vector of probabilities. Must have names & should probably add to 1.0.

Details

This function picks a random element in a vector according to the probability assigned that element. It returns the name. Uses rmultinom.

Value

A character or string.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


sampleFrom

Sample from a set of distributions

Description

Function samples from a set of normal distributions with parameters given in xbar and xvar.

Usage

sampleFrom(xbar=0, xvar=1, n=1, randn=NULL, type="norm")
setMap

Arguments

- xbar: a named vector of means.
- xvar: a named vector of variances.
- n: a vector containing the sample sizes of each species.
- randn: a range of sample sizes are to be random.
- type: "norm" is the only distribution implemented so far.

Value

A vector, with labels.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


setMap  

Set color map for objects of class "contMap" or "densityMap"

Description

Function to change the color map (ramp) in an object of class "contMap" or "densityMap".

Usage

setMap(x, ...)

Arguments

- x: an object of class "contMap" or "densityMap".
- ...: arguments to be passed to colorRampPalette. Also, the argument invert which (if invert=TRUE) will just flip the current color ramp.

Value

An object of class "contMap" or "densityMap".

Author(s)

Liam Revell <liam.revell@umb.edu>
References


See Also

`contMap, densityMap`

---

**sim.corrs**

Multivariate Brownian simulation with multiple correlations and rates

Description

This function conducts BM simulation on a tree with multiple rates and/or multiple evolutionary correlations between characters. If vcv is a single matrix, instead of a list of matrices, `sim.corrs` will simulate multivariate BM with a single rate matrix.

Usage

`sim.corrs(tree, vcv, anc=NULL, internal=FALSE)`

Arguments

- **tree**
  - is a phylogenetic tree in 'phylo' format; or a modified 'phylo' tree with a mapped discrete character.

- **vcv**
  - is a square covariance matrix or named list of matrices (one for each mapped state on the tree).

- **anc**
  - optional vector of values for the root state.

- **internal**
  - logical value indicating whether to return states at internal nodes.

Value

A matrix containing the multivariate tip states for the \( n \) species in the tree (and nodes if `internal=TRUE`).

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

fastBM, make.simmap, read.simmap, sim.history, sim.rates

---

**sim.history**

*Simulate stochastic character history under some model*

**Description**

This function simulates a stochastic character history for a discretely valued character trait on the tree. The resultant tree is stored as a modified "phylo" object in stochastic character map (e.g., make.simmap) format.

**Usage**

```r
sim.history(tree, Q, anc=NULL, nsim=1, ...)
```

**Arguments**

- `tree`: a phylogenetic tree as an object of class "phylo".
- `Q`: a matrix containing the instantaneous transition rates between states. Note that normally this is the transpose of the matrix produced by `fitDiscrete` in the geiger package or `make.simmap` in phytools; that is to say the transition rate from \( i \rightarrow j \) should be given by \( Q[j,i] \). However, if your matrix is properly conformed (i.e., rows or columns sum to 0), then `sim.history` will attempt to transpose your matrix correctly & will return an informative message (if `message=TRUE`, see below).
- `anc`: an optional value for the state at the root node; if `NULL`, a random state will be assigned.
- `nsim`: number of simulations.
- `...`: other optional arguments. Currently only `message`, a logical value indicating whether or not to turn on informational messages (defaults to `message=TRUE`).

**Value**

A modified phylogenetic tree of class "phylo" (or a modified "multiPhylo" object, for `nsim > 1`) with the following additional elements:

- `maps`: a list of named vectors containing the times spent in each state on each branch, in the order in which they occur.
- `mapped.edge`: a matrix containing the total time spent in each state along each edge of the tree.
- `states`: a vector containing the tip states.
- `node.states`: a matrix containing the states at internal & terminal nodes (according to the dimensions of `edge`).
Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

make.simmap, read.simmap, plotSimmap, sim.rates

sim.ratebystate

Conduct simulation of state dependent rate variation

Description

This function attempts to simulate two characters under a model in which the rate of evolution for the second (y) depends on the states for the first x. See ratebystate for more details.

Usage

sim.ratebystate(tree, sig2x=1, sig2y=1, beta=c(0,1), ...)

Arguments

tree phylgenetic tree.
sig2x variance of the Brownian process of evolution for x.
sig2y variance of the Brownian process of evolution for y when x=min(x)=1 (for logarithm=FALSE) or x=0 (for logarithm=TRUE).
beta intercept and slope of the relationship between the value of x and the Brownian rate in y.
... optional arguments which include method ("by.node" or "by.branch" indicating whether to assume the rate varies as a function of the node state or the mean branch state); plot, a logical value indicating whether or not to plot a phenogram with the branches used for simulation of y after rescaling by the state of x; and logarithm, a logical value indicating whether or not simulate changes in the variance of Brownian evolution for y as an additive logarithm=FALSE or multiplicative function of x. The default is logarithm=FALSE.

Value

This function returns a matrix.

Author(s)

Liam Revell <liam.revell@umb.edu>
References


See Also

fastBM, ratebystate

---

**sim.rates**  
*Brownian simulation with multiple evolutionary rates*

**Description**

This function conducts BM simulation on a tree with multiple rates.

**Usage**

```r
sim.rates(mtree, sig2, anc=0, nsim=1, internal=F, plot=F)
```

**Arguments**

- `mtree` is a stochastic map format phylogenetic tree in modified "phylo" format (e.g., see `make.simmap`).
- `sig2` a named vector containing the rates for each state; names should be states in `mtree`.
- `anc` optional value for the root state.
- `nsim` number of simulations.
- `internal` logical value indicating whether to return states at internal nodes.
- `plot` logical value indicating whether or not to visualize the rate heterogeneity (default value is FALSE).

**Value**

A vector (for nsim=1) or matrix containing the tip states for the n species in the tree.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


**See Also**

fastBM, make.simmap, read.simmap, sim.history
Matrix comparison using the method of random skewers

**Description**

This function performs the random skewers matrix comparison method of Cheverud (1996; also see Cheverud & Marroig 2007 for more details). In addition, it includes a more robust hypothesis test in which random covariance matrices are simulated under a variety of models, and then the mean correlation between response vectors to random skewers are computed.

**Usage**

```r
skewers(x, y, nsim=100, method=NULL)
```

**Arguments**

- `x`: covariance matrix.
- `y`: covariance matrix.
- `nsim`: number of random vectors.
- `method`: method to generate a null distribution of the random skewers correlation between matrices. If `method=NULL` then the correlation will be compared to the correlation between random vectors; however this test has type I error substantially above the nominal level for ostensibly random matrices. Other values of `method` will be passed as `covmethod` to `genPositiveDefMat` for a more robust hypothesis test (see below). Recommended values include "unifcorrmat".

**Value**

A list with the following components:

- `r`: mean random skewers correlation.
- `p`: p-value from simulation.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**

splitplotTree

Plots a phylogeny in two columns

Description

Function plots a tree in two columns or windows.

Usage

splitplotTree(tree, fsize=1.0, ftype="reg", lwd=2, split=NULL, new.window=FALSE)

Arguments

tree
  an object of class "phylo".

fsize
  relative font size for tip labels.

ftype
  font type - options are "reg", "i" (italics), "b" (bold), or "bi" (bold-italics).

lwd
  line width for plotting.

split
  relative vertical position for splitting the tree (between 0 & 1).

new.window
  whether or not to plot the split tree in a new window. If FALSE then the tree will
  be plotted in two columns within the same plotting window.

Value

Plots a tree.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

plotTree, plotSimmap
splitTree

Split tree at a point

Description

Primarily an internal function for `posterior.evolrate`, this function splits the tree at a given point, and returns the two subtrees as an object of class "multiPhylo".

Usage

```
splitTree(tree, split)
```

Arguments

- `tree`: phylogenetic tree.
- `split`: split encoded as a list with two elements: `node`: the node number tipward of the split; and `bp`: the position along the branch to break the tree, measured from the rootward end of the edge.

Details

Probably do not use this unless you can figure out what you are doing.

Value

Two trees in a list.

Author(s)

Liam Revell <liam.revell@umb.edu>

References

starTree  

Create star phylogeny

Description

This function creates a star phylogeny.

Usage

starTree(species, branch.lengths=NULL)

Arguments

species  
a list of species.

branch.lengths  
an optional list of branch lengths in the same order as species.

Details

Creates a star phylogeny with (optionally) user specified branch lengths.

Value

An object of class "phylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

stree
strahlerNumber Computes Strahler number for trees and nodes

Description

The function strahlerNumber computes the Strahler number of all nodes and tips in the tree. For more information about Strahler numbers see http://en.wikipedia.org/wiki/Strahler_number. The function extract.strahlerNumber extracts all of the most inclusive clades of Strahler number i.

Usage

strahlerNumber(tree, plot=TRUE)
extract.strahlerNumber(tree, i, plot=TRUE)

Arguments

tree an object of class "phylo".
i order of Strahler number to extract for extract.strahlerNumber.
plot logical value indicating whether to plot the tree with Strahler numbers for node labels.

Value

Either a vector with the Strahler number for each tip and internal node; or (for extract.strahlerNumber) the set of (most inclusive) subtrees with Strahler number i as an object of class "multiPhylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References

**threshBayes**

*Threshold model using Bayesian MCMC*

**Description**

This function uses Bayesian MCMC to fit the quantitative genetics threshold model (Felsenstein 2012) to data for two discrete characters or one discrete and one continuous character.

**Usage**

```r
threshBayes(tree, X, types=NULL, ngen=1000, control=list())
```

**Arguments**

- `tree` an object of class "phylo".
- `X` a numeric matrix containing values for a numerically coded discrete character and a continuous character; or two discrete characters. The row names of `X` should be species names.
- `types` a vector of length `ncol(X)` containing the data types for each column of `X`, for instance `c("discrete","continuous")`.
- `ngen` a integer indicating the number of generations for the MCMC.
- `control` a list of control parameters for the MCMC. Control parameters include: `sample`, the sampling interval for the MCMC; `propvar`, a vector containing (in this order) proposal variances for the two rates (if the type is "discrete" this will be ignored), the two ancestral states, and the correlation; `propliab`, a single proposal variance for the liabilities; `pr.mean`, a vector for the mean of the prior probability distributions for each parameter, in the same order as `propvar`; `pr.liab`, currently ignored; `pr.var`, a vector with variances for the prior densities for each parameter, in the same order as `pr.mean` - note that for the rates we use an exponential distribution so the first two means are currently ignored; and `pr.vliab` currently ignored.

**Value**

This function returns a list with two elements: `par` a matrix containing the posterior sample for the model parameters (evolutionary rates, ancestral states, and correlation); `liab` a matrix containing the posterior sample of the liabilities. For continuous characters, the liabilities are treated as known and so the posterior samples are just the observed values.

**Author(s)**

Liam Revell <liam.revell@umb.edu>
References


See Also

`anc.Bayes`, `bmPlot`, `evol.rate.mcmc`

threshDIC

**Deviance Information Criterion from the threshold model**

Description

This function computes the Deviance Information Criterion from the MCMC object returned by `ancThresh`.

Usage

threshDIC(tree, x, mcmc, burnin=NULL, sequence=NULL, method="pD")

Arguments

- `tree`: phylogenetic tree.
- `x`: a named vector containing discrete character states; or a matrix containing the tip species, in rows, and probabilities of being in each state, in columns.
- `mcmc`: list object returned by `ancThresh`.
- `burnin`: number of generations (not samples) to exclude as burn in; if not supplied then 20 percent of generations are excluded.
- `sequence`: assumed ordering of the discrete character state. If not supplied and `x` is a vector then numerical-alphabetical order is assumed; if not supplied and `x` is a matrix, then the column order of `x` is used.
- `method`: method for computing the effective number of parameters (options are "pD" and "pV").

Value

A vector containing the mean deviance and deviance for the parameter means, the effective number of parameters, and the DIC.

Author(s)

Liam Revell <liam.revell@umb.edu>
threshState

References


See Also

ancThresh

threshState

*Computes value for a threshold character from a liability and thresholds*

Description

Primarily to be used internally by ancThresh; can also be used to simulate threshold traits.

Usage

threshState(x, thresholds)

Arguments

x liability.

thresholds a named vector containing the thresholds.

Value

A discrete character value.

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

ancThresh, threshDIC
to.matrix  

*Convert a character vector to a binary matrix*

**Description**

This function takes a vector of characters and computes a binary matrix. Primarily to be used internally by `make.simmap` and `rerootingMethod`.

**Usage**

```r
to.matrix(x, seq)
```

**Arguments**

- `x`: a vector of characters.
- `seq`: the sequence for the columns in the output matrix.

**Value**

A binary matrix of dimensions `length(x)` by `length(seq)`.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**


---

treeSlice  

*Slices the tree at a particular point and returns all subtrees*

**Description**

This function slices a tree at a particular height above the root and returns all subtrees or all non-trivial subtrees (i.e., subtrees with more than 1 taxon). Uses `extract.clade` in the "ape" package.

**Usage**

```r
treeSlice(tree, slice, trivial=FALSE)
```
untangle

Arguments

- **tree**: is a phylogenetic tree in "phylo" format.
- **slice**: a real number indicating the height above the root at which to slice the tree.
- **trivial**: a logical value indicating whether or not to return subtrees with a number of tips less than two (default is FALSE).

Value

An object of class "phylo" or "multiPhylo".

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

- extract.clade

Description

This function attempts to untangle the branches of a tree that are tangled in plotting with `plot.phylo`, `plotTree`, or `plotSimmap`. Note that method="read.tree" does not presently work for SIMMAP style trees.

Usage

```r
untangle(tree, method=c("reorder","read.tree"))
```

Arguments

- **tree**: tree as an object of class "phylo". Can be a SIMMAP style tree (e.g., `read.simmap`).
- **method**: method to use to attempt to untangle branches. method="reorder" uses two calls of `reorder.phylo` or `reorderSimmap`; method="read.tree" writes the tree to a text string and then reads it back into memory using `read.tree`.

Value

A tree with branch lengths, or modified "phylo" object with a mapped discrete character.
vcvPhylo

**Calculates cophenetic (i.e., phylogenetic VCV) matrix**

**Description**

This function returns a so-called *phylogenetic variance covariance matrix* (e.g., see `vcv.phylo`), but (optionally) including ancestral nodes and under different evolutionary models.

**Usage**

```
vcvPhylo(tree, anc.nodes=TRUE, ...)
```

**Arguments**

- `tree` object of class "phylo".
- `anc.nodes` logical value indicating whether or not to include ancestral nodes.
- `...` optional arguments including internal (synonym of `anc.nodes`) and model (can be "BM", "OU", or "lambda").

**Value**

A matrix.

**Author(s)**

Liam Revell <liam.revell@umb.edu>

**References**

write.simmap

Write a stochastic character mapped tree to file

Description

This function writes stochastic character mapped trees to file using the Newick style format of SIMMAP v1.0 (Bollback 2006). Note, can only write one tree at a time to file (hence the append option).

Usage

write.simmap(tree, file=NULL, append=FALSE, map.order=NULL)

Arguments

tree        a phylogenetic tree as a modified object of class "phylo". See make.simmap and read.simmap.
file        an optional filename.
append      a logical value indicating whether to append to file.
map.order   a optional value specifying whether to write the map in left-to-right or right-to-left order. Acceptable values are "left-to-right" or "right-to-left" or some abbreviation of either. If not provided, write.simmap will use attr(tree,"map.order") if available.

Value

A file or string (if file=NULL).

Author(s)

Liam Revell <liam.revell@umb.edu>

References


See Also

make.simmap, read.simmap, plotSimmap
writeAncestors

Write a tree to file with ancestral states and (optionally) CIs at nodes

Description

This function writes a tree to file with ancestral character states and (optionally) 95-percent confidence intervals stored as node value.

Usage

writeAncestors(tree, Anc=NULL, file="", digits=6, format=c("phylip","nexus"), ...)

Arguments

tree
    a phylogenetic tree or set of trees as an object of class "phylo" or "multiPhylo".

Anc
    a vector of ancestral states, a list containing the ancestral states and 95-percent confidence intervals (as from fastAnc or ace, or a list of such results.

file
    an optional string with the filename for output.

digits
    an integer indicating the number of digits to print for branch lengths and ancestral character values.

format
    a string indicating whether to output the result in simple Newick (i.e., "phylip") or Nexus format.

... additional arguments including x: a vector of character values, in which case ancestral states are estimated internally using fastAnc; and CI: a logical value indicating whether or not to estimate 95-percent confidence intervals.

Value

A file, string, or vector of strings.

Author(s)

Liam Revell <liam.revell@umb.edu>
writeNexus

References

See Also
ace, fastAnc, write.tree

----------
writeNexus  Write a tree to file in Nexus format

Description
This function writes one or multiple phylogenetic trees to file in NEXUS format. Redundant with ape::write.nexus.

Usage
writeNexus(tree, file="")

Arguments
tree          object of class "phylo" or "multiPhylo".
file          file name for output.

Value
Trees written to file.

Author(s)
Liam Revell <liam.revell@umb.edu>

References

See Also
write.simmap, write.nexus
Index

*Topic animation
branching.diffusion, 22

*Topic bayesian
anc.Bayes, 12
ancThresh, 16
ave.rates, 18
evol.rate.mcmc, 38
fitBayes, 50
make.simmap, 68
mergeMappedStates, 73
minSplit, 75
plotThresh, 110
posterior.evolrate, 113
rerootingMethod, 122
threshBayes, 139
threshDIC, 140

*Topic comparative method
add.color.bar, 7
add.simmap.legend, 9
anc.Bayes, 12
anc.ML, 13
anc.trend, 14
ancThresh, 16
ave.rates, 18
brownie.lite, 23
brownieREML, 24
dcontMap, 28
densityMap, 30
describe.simmap, 32
di2multi.simmap, 33
estDiversity, 37
evol.rate.mcmc, 38
evol.vcv, 39
evolvcv.lite, 41
fastAnc, 46
fitBayes, 50
fitDiversityModel, 51
fitPagel, 53
gammatest, 54
lambda.transform, 60
likMLambda, 61
make.era.map, 67
make.simmap, 68
map.overlap, 70
mergeMappedStates, 73
minSplit, 75
multi.mantel, 77
nodeHeights, 80
paintSubTree, 83
pgls.Ives, 87
phenogram, 88
phyl.cca, 90
phyl.pairedttest, 91
phyl.pca, 93
phyl.resid, 94
phyl.RMA, 95
phylANOVA, 97
phyломorphospace, 101
phyломorphospace3d, 102
phylosig, 104
plotBranchbyTrait, 106
plotSimmap, 108
plotThresh, 110
plotTree.wBars, 112
posterior.evolrate, 113
ratebystate, 114
rateshift, 116
read.simmap, 118
reorderSimmap, 120
rerootingMethod, 122
roundPhylogram, 126
sim.corrs, 130
sim.ratebystate, 132
sim.rates, 133
skewers, 134
threshBayes, 139
threshDIC, 140
threshState, 141
INDEX

write.simmap, 145
writeAncestors, 146
*Topic datasets
anoletree, 17
*Topic distance matrix
optim.phylo.ls, 81
*Topic diversification analysis
ltt, 64
ltt95, 66
*Topic diversification
gammatest, 54
*Topic inference
allFurcTrees, 11
exhaustiveMP, 42
locate.fossil, 62
locate.yeti, 63
ls.tree, 64
mrp.supertree, 76
optim.phylo.ls, 81
phyloDesign, 100
*Topic input/output
read.newick, 117
read.simmap, 118
write.simmap, 145
writeAncestors, 146
writeNexus, 147
*Topic least squares
ls.tree, 64
multi.mantel, 77
optim.phylo.ls, 81
pgls.Ives, 87
phyL.resid, 94
phyLANOVA, 97
phyloDesign, 100
*Topic math
expm, 43
lambda.transform, 60
likMLambda, 61
rstate, 128
*Topic maximum likelihood
anc.ML, 13
anc.trend, 14
brownie.lite, 23
estDiversity, 37
evol.vcv, 39
evolvcv.lite, 41
fastAnc, 46
fitDiversityModel, 51
locate.fossil, 62
locate.yeti, 63
pgls.Ives, 87
phyL.paired.t.test, 91
phyL.pca, 93
phyL.resid, 94
phyL.RMA, 95
phylosig, 104
rerootingMethod, 122
*Topic package
phytools-package, 5
*Topic parsimony
exhaustiveMP, 42
mrp.supertree, 76
*Topic phylogenetics
add.arrow, 6
add.color.bar, 7
add.everywhere, 8
add.random, 8
add.simmap.legend, 9
add.species.to.genus, 10
allFurcTrees, 11
anc.Bayes, 12
anc.ML, 13
anc.trend, 14
ancThresh, 16
applyBranchLengths, 18
ave.rates, 18
bind.tip, 19
bmPlot, 20
branching.diffusion, 22
brownie.lite, 23
brownieREML, 24
cladelabels, 25
collapse.to.star, 26
collapseTree, 27
collapseTree, 27
collapseTree, 27
countMap, 28
countSimmap, 29
densityMap, 30
describe.simmap, 32
di2multi.simmap, 33
drop.clade, 34
drop.leaves, 34
drop.tip.contMap, 35
drop.tip.simmap, 36
estDiversity, 37
evol.rate.mcmc, 38
evol.vcv, 39
writeNsimmap, 145
writeAncestors, 146
writeNexus, 147

*Topic **plotting**
add.color.bar, 7
add.ssimmap.legend, 9
bmPlot, 20
branching.diffusion, 22
collapseTree, 27
contMap, 28
densityMap, 30
fancyTree, 44
lItt, 64
lItt95, 66
phenogram, 88
phylo.to.map, 98
phylo.toBackbone, 99
phylomorphospace, 101
phylomorphospace3d, 102
plot.backbonePhylo, 105
plotBranchbyTrait, 106
plotSimmap, 108
plotTree, 111
plotTree.wBars, 112
print.backbonePhylo, 114
reorder.backbonePhylo, 119
roundPhylogram, 126
setMap, 129
splitplotTree, 135

*Topic **simulation**
bmPlot, 20
branching.diffusion, 22
fastBM, 47
genSeq, 55
make.ssimmap, 68
mergeMappedStates, 73
pbtree, 85
phylANOVA, 97
phylosig, 104
sim.corrs, 130
sim.history, 131
sim.ratebystate, 132
sim.rates, 133
threshState, 141

*Topic **statistics**
multi.mantel, 77
pgls.Ives, 87
phyl.cca, 90
phyl.pairedttest, 91
phyl.pca, 93
phyl.resid, 94
phyl.RMA, 95
phyl.vcv, 96
phylANOVA, 97
rstate, 128
sampleFrom, 128
skewers, 134
vcvPhylo, 144

*Topic **supertree**
mrp.supertree, 76

*Topic **utilities**
add.arrow, 6
add.everywhere, 8
add.random, 8
add.species.to.genus, 10
applyBranchLengths, 18
bind.tip, 19
cladelabels, 25
collapse.to.star, 26
countSimmap, 29
describe.ssimmap, 32
di2multi.ssimmap, 33
drop.clade, 34
drop.leaves, 34
drop.tip.contMap, 35
drop.tip.ssimmap, 36
export.as.xml, 43
fancyTree, 44
fastMRCA, 48
findMRCA, 49
getCladesOfSize, 56
getDescendants, 56
getExtant, 57
getSisters, 58
getStates, 59
ladderize.ssimmap, 59
likMLambda, 61
map.to.singleton, 71
matchNodes, 72
mergeMappedStates, 73
midpoint.root, 74
minRotate, 75
multiC, 78
multiRF, 79
nodeHeights, 80
orderMappedEdge, 82
paste.tree, 84
phyl.vcv, 96
reorderSimmap, 120
rep.phylo, 120
reroot, 121
rescaleSimmap, 124
rotateNodes, 125
roundBranches, 126
rstate, 128
splitTree, 136
starTree, 137
strahlernumber, 138
to.matrix, 142
treeSlice, 142
untangle, 143
vcvPhylo, 144
ace, 13–15, 37, 47, 53, 68, 69, 123, 146, 147
collapse.to.star, 26
collapseTree, 27
colorRampPalette, 129
countSimmap, 32
contMap, 7, 28, 35, 36, 44, 45, 130
cor, 115
countSimmap, 29, 32, 70
densityMap, 7, 29, 30, 35, 36, 44, 45, 108, 109, 130
describe.simmap, 32, 59, 70
dev.hold, 45, 89
di2multi, 33
di2multi.simmap, 33
drop.clade, 34
drop.leaves, 34
drop.tip, 35, 36, 45, 71
drop.tip.contMap, 35
drop.tip.densityMap(drop.tip.contMap), 35
drop.tip.simmap, 36, 36
drop.tip.singleton (map.to.singleton), 71
estDiversity, 37, 52
evol.rate.mcmc, 13, 19, 25, 38, 40, 51, 52, 76, 113, 140
evol.vcv, 24, 25, 39, 39, 41, 70, 119
evolvcv.lite, 41, 79
exhaustiveMP, 8, 12, 42, 77, 82
expm, 43
export.as.xml, 43
extract.clade, 36, 56, 142, 143
extract.clade.simmap (drop.tip.simmap), 36
extract.strahlernumber
(strahlernumber), 138
fancyTree, 44, 103
fastAnc, 13, 14, 28, 29, 46, 72, 101, 115, 146, 147
fastBM, 21, 22, 47, 115, 131, 133
fastHeight (fastMRCA), 48
fastMRCA, 48
findMRCA, 49, 49, 50
fitBayes, 50
fitDiversityModel, 38, 51
fitPage, 53
gammatest, 54, 65
INDEX

genSeq, 55
getCladesOfSize, 56
getDescendants, 56, 56
getExtant, 57
getExtinct (getExtant), 57
getSisters, 58
getStates, 59
gls, 95
ladderize, 59, 60
ladderize.simmap, 59
lambda.transform, 60
likMlambda, 61
locate.fossil, 62
locate.yeti, 62, 63
ls.tree, 64
ltt, 54, 64, 67
ltt95, 65, 66
make.era.map, 67
make.simmap, 23, 24, 29–33, 36, 37, 44, 53,
59, 60, 67, 68, 71, 73, 84, 108, 109,
123, 124, 128, 131–133, 142, 145
map, 99
map.overlap, 70
map.to.singleton, 71
matchNodes, 72
MatrixExp, 43
mergeMappedStates, 73
midpoint.root, 74
min, 75
minRotate, 75
minSplit, 19, 39, 75, 113
mrca, 49, 50
mrp.supertree, 42, 76
multi.mantel, 77
multiC, 78
multiRF, 79
nni, 82
nodeheight (nodeHeights), 80
nodeHeights, 58, 80
nodeLabels, 6, 26, 108
optim, 14, 15, 25, 40, 88, 92, 104, 105
optim.parsimony, 42, 76, 77
optim.phylol.s, 64, 81, 100
optimize, 90, 93–95, 104
orderMappedEdge, 82
p.adjust, 97
paintBranches (paintSubTree), 83
paintSubTree, 23, 57, 83
pairwise.t.test, 97, 98
palette, 71
parsimony, 42
paste.tree, 84, 122
pbtree, 21, 85
pgls.Ives, 87
phenogram, 21, 44, 45, 88, 103
phyDat, 42
phyI.cca, 90, 94, 96
phyI.pairedttest, 91
phyI.pca, 60, 91, 93, 95, 96
phyI.resid, 88, 94, 94, 96
phyI.RMA, 95
phyI.vcv, 96
phyIANOVA, 97
phylo.to.map, 75, 98
phylo.toBackbone, 99, 106, 114, 119
phyloDesign, 100
phylomorphospace, 45, 101, 103
phylomorphospace3d, 45, 102
phylosig, 88, 104
phytools (phytools-package), 5
phytools-package, 5
pic, 47, 115
plot, 64
plot.backbonePhylo, 100, 105
plot.contMap (contMap), 28
plot.default, 65, 89, 101
plot.densityMap (densityMap), 30
plot.ltt95 (ltt95), 66
plot.phylo, 9, 45, 106, 107, 109–111, 143
plot.phylo.to.map (phylo.to.map), 98
plotBranchbyTrait, 7, 106
plotSimmap, 9, 10, 27–29, 31, 45, 67, 70, 84,
108, 111, 112, 120, 127, 132, 135,
143, 145
plotThresh, 16, 110
plotTree, 27, 111, 112, 127, 135, 143
plotTree.singletons (map.to.singleton), 71
plotTree.wBars, 112
posterior.evolute, 18, 19, 34, 39, 76, 84,
113, 136
pratchet, 42, 76, 77
print.backbonePhylo, 114
prop.part, 77, 79
ratebystate, 114, 132, 133
rateshift, 116
read.newick, 117
read.nexus, 44, 117–119
read.simmap, 18, 23, 24, 29–33, 36, 39, 41,
44, 59, 67, 70, 71, 73, 79, 82, 84,
108, 109, 118, 124, 126, 131–133,
143, 145
read.tree, 117, 119, 143
reorder.backbonePhylo, 119
reorder.phylo, 119, 120, 143
reorderSimmap, 120, 143
rep, 121
rep.multiPhylo (rep.phylo), 120
rep.phylo, 120
repPhylo (rep.phylo), 120
reroot, 74, 121
rerootingMethod, 122, 142
rescaleSimmap, 124
rmultinom, 128
root, 74, 122
rotate, 125
rotateNodes, 125
round, 18, 126
roundBranches, 126
roundPhylogram, 126
rstate, 128
sampleFrom, 128
scatterplot3d, 103
setMap, 129
sim.corr, 48, 115, 130
sim.history, 36, 55, 59, 84, 131, 131, 133
sim.ratebystate, 132
sim.rates, 131, 132, 133
skewers, 134
splitplotTree, 135
splitTree, 27, 122, 136
starTree, 27, 137
strahlerNumber, 138
stree, 137
threshBayes, 17, 21, 139
threshDIC, 140, 141
threshState, 141
to.matrix, 142
treeSlice, 142
untangle, 9, 143
vcv.phylo, 49, 50, 60, 61, 79, 96, 144
vcvPhylo, 79, 81, 144
write.nexus, 147
write.simmap, 44, 70, 118, 145, 147
write.tree, 147
writeAncestors, 146
writeNexus, 147