Package ‘apTreeshape’

February 19, 2015

Date 2009-03-13

Version 1.4-5

Title Analyses of Phylogenetic Treeshape

Author Nicolas Bortolussi, Eric Durand, Michael Blum
<michael.blum@imag.fr>, Olivier Francois
<olivier.francois@imag.fr>

Maintainer Michael Blum <michael.blum@imag.fr>

Depends R (>= 2.0.0), ape

Imports quantreg

Description apTreeshape is mainly dedicated to simulation and analysis of phylogenetic tree topologies using statistical indices. It is a companion library of the 'ape' package. It provides additional functions for reading, plotting, manipulating phylogenetic trees. It also offers convenient web-access to public databases, and enables testing null models of macroevolution using corrected test statistics. Trees of class ``phylo'' (from 'ape' package) can be converted easily.

License GPL (>= 2)

Encoding latin1

Repository CRAN

Date/Publication 2012-11-09 07:03:26

NeedsCompilation no

R topics documented:

aldous.test 2
all.equal.treeshape 4
as.phylo.treeshape 5
as.treeshape 6
carnivora.treeshape 7
cladesize 8

colless . . . 9
cutreeshape . . . 11
cytochromc . . . 12
hivtree.treeshape . . . 12
index.test . . . 13
likelihood.test . . . 15
maxlik.betasplit . . . 17
plot.treeshape . . . 18
primates . . . 19
rhodopsin . . . 19
rtreeshape . . . 20
sackin . . . 22
shape.statistic . . . 23
shift.test . . . 25
smaller.clade.spectrum . . . 26
spectrum.treeshape . . . 27
subtree.test . . . 28
summary.treeshape . . . 30
tipsubtree . . . 31
treeshape . . . 32
universal.treeshape . . . 34

Index **35**

`aldous.test` *Visualizing balance via scatter diagrams*

Description

A graphical test to decide if tree data fit the Yule or the PDA models.

Usage

```
aldous.test(tree, xmin=20, ...)
```

Arguments

`tree`	An object of class `"treeshape"`.
`xmin`	An object of class `"numeric"` that defines the range of the x-axis. The minimal parent clade size displayed in the graphical representation (default: `xmin=20`).
`...`	further arguments to be passed to `plot()`.

Details

A binary tree contains a set of splits

$$(m, i) = (size\tilde{}of\tilde{}parent\tilde{}clade, size\tilde{}of\tilde{}smaller\tilde{}daughter\tilde{}clade)$$

which can be plotted as a scatter diagram. Aldous' proposal for studying tree balance is that, given a large phylogenetic tree, one should estimate the median size of the smaller daughter clade as a function of the parent clade and use this function as a descriptor of balance or imbalance of the tree. It is convenient to make a log-log plot and to ignore small parent clades. The scatter diagram shows lines giving the approximate median values of the size of smaller daughter clade predicted by the beta-splitting model for two values of beta, the value for the Yule ($\beta = 0$) and PDA ($\beta = -1.5$) models. In other words, if the null model were true, then the scatter diagram for a typical tree would have about half the points above the line and half below the line, throughout the range.

The green line represents the median regression estimated from the tree data.

Value

The function provides a graphical display of results.

Author(s)

Michael Blum <michael.blum@imag.fr>
Nicolas Bortolussi <nicolas.bortolussi@imag.fr>
Eric Durand <eric.durand@imag.fr>
Olivier Francois <olivier.francois@imag.fr>

References

Aldous, D. J. (1996) *Probability Distributions on Cladograms*. pp.1-18 of *Random Discrete Structures* eds D. Aldous and R. Pemantle, IMA Volumes Math. Appl. 76.

Aldous, D. J. (2001) Stochastic Models and Descriptive Statistics for Phylogenetic Trees, from Yule to Today. *Statistical Science*, **16**, 23 – 24.

Examples

```
library(quantreg)
aldous.test(rbiased(2000, p=.5))

## Test with a huge balanced tree:
aldous.test(rbiased(2000, p=.5))
```

`all.equal.treeshape` *Compare two objects of class treeshape*

Description

This function makes a global comparison of two phylogenetic trees.

Usage

```
## S3 method for class 'treeshape'
all.equal(target, current, names=FALSE, height=FALSE, ...)
```

Arguments

`target`	An object of class `"treeshape"`.
`current`	An object of class `"treeshape"`.
`names`	An object of class `"logical"`, checking if the names of the tips should be tested. If `FALSE` (default), the names of the tips are not compared.
`height`	An object of class `"logical"`, checking if the heights of the nodes should be tested. If `FALSE` (default), the height of internal nodes are not compared.
`...`	further arguments passed to or from other methods.

Details

This function is meant to be an adaptation of the generic function `all.equal` for the comparison of phylogenetic trees. A phylogenetic tree can have many different representations. Permutations between the left and the right daughter clade of a node do not change the corresponding phylogeny, and `all.equal.treeshape` returns `TRUE` on two permutated trees.

Value

Returns the logical `TRUE` if the tree objects are similar up to a permutation of their tips. Otherwise, it returns `FALSE`. Heights and labels can be taken into account.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier Francois <<olivier.francois@imag.fr>>

See Also

`all.equal` for the generic R function.

Examples

```
## Trees with permutations
data(carnivora.treeshape)
tree=carnivora.treeshape
tree$merge[8,]=c(tree$merge[8,2],tree$merge[8,1])
all.equal(tree, carnivora.treeshape)

## Trees with different heights
merge=matrix(NA, 3, 2)
merge[,1]=c(-3,-1,2); merge[,2]=c(-4,-2,1);tree1=treeshape(merge)
merge[,1]=c(-1,-3,1); merge[,2]=c(-2,-4,2);tree2=treeshape(merge)

plot(tree1, tree2)
all.equal(tree1, tree2)
all.equal(tree1, tree2, height=TRUE)

## Trees with different names
tree3=treeshape(tree1$merge, c("a", "b", "c", "d"))
tree4=treeshape(tree1$merge, c("1", "2", "3", "4"))
plot(tree3, tree4)
all.equal(tree3, tree4)
all.equal(tree3, tree4, names=TRUE)
```

`as.phylo.treeshape` *Conversion among tree objects*

Description

`as.phylo` is a generic function - described in the APE package - which converts an object into a tree of class "`phylo`". This method is an adataption of this generic method to convert objects of class "`treeshape`" in objects of class "`phylo`".

Usage

```
## S3 method for class 'treeshape'
as.phylo(x, ...)
```

Arguments

`x`	An object of class"`treeshape`".
`...`	further arguments to be passed to or from other methods.

Value

An object of class "`phylo`".

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier Francois <<olivier.francois@imag.fr>>

See Also

as.phylo
as.treeshape

Examples

```
data(primates)
plot(primates)

library(ape)

primates.phylo=as.phylo(primates)
plot(primates.phylo)
```

---

as.treeshape *Conversion among tree objects*

Description

as.treeshape is a generic function which converts an tree object into an object of class "treeshape". There is currently a single method for this generic function.

Usage

```
## S3 method for class 'phylo'
as.treeshape(x, model, p, ...)
```

Arguments

x	An object to be converted into another class. Currently, it must be an object of class "phylo".
model	The model to use when the tree to convert is not binary. If NULL (default), the tree is not converted. One of "biased", "pda", "aldous" or "yule" character string.
p	The parameter for the model "biased".
...	Further arguments to be passed to or from other methods.

Details

as.treeshape can convert trees that are not binary. When trying to convert a tree with polytomies, this function may either reject the tree (if model=NULL) or simulate the tree. The polytomy is replaced by a randomized subtree with n tips where n is the size of the polytomy. The subtree is simulated using the PDA, Yule, Aldous or biased model.

Value

An object of class "treeshape" or an object of classes "treeshape" and "randomized.treeshape" if the original tree was not binary.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier Francois <<olivier.francois@imag.fr>>

See Also

as.phylo.treeshape

Examples

```
library(ape)
data(bird.orders)
## Data set from APE
plot(bird.orders)

## "treeshape" conversion
tree=as.treeshape(bird.orders)
plot(tree)
summary(tree)
```

---

carnivora.treeshape *Phylogeny of carnivores.*

Description

An object of class "treeshape" containing the phylogeny of 12 biological familes in the Carnivora.

Usage

```
carnivora.treeshape
```

`cladesize` *Compute the number of children of a randomly chosen node*

Description

`cladesize` takes a random internal node in a tree, and computes its number of descendants (clade size).

Usage

```
cladesize(tree)
```

Arguments

`tree`	An object of class `"treeshape"`.

Details

This function can be used to check whether a tree fits the Yule or the PDA models. Under the Yule model, the probability distribution of the random clade size is equal to

$$P(Kn = k) = \frac{2n}{(n-1)k(k+1)}$$

for $k = 2, 3, \ldots, n-1$ and

$$P(Kn = n) = \frac{1}{n-1}$$

(where n is the number of tips of the tree and Kn is the number of descendents of an internal node of the tree). Under the PDA model, the asymptotic distribution (when the number of tips grows to infinity) of the random clade size is equal to

$$P(K = k+1) = \frac{\binom{2k}{k}}{(k+1)(2^k)^2}$$

.

Value

An object of class `numeric` representing the clade size of a random node of a tree.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Oliver François <<olivier.francois@imag.fr>>

References

Blum, M., Francois, O. and Janson, S. The mean, variance and limiting distribution of two statistics sensitive to phylogenetic tree balance; manuscript available from http://www-timc.imag.fr/Olivier.Francois/bfj.pdf.

Examples

```
# Histogram of random clade sizes
main="Random clade sizes for random generated trees"
xlabel="clade size"
hist(sapply(rtreeshape(100,tip.number=40,model="yule"),FUN=cladesize),
      freq=FALSE,main=main,xlab=xlabel)
```

`colless` *Compute the Colless' shape statistic on tree data*

Description

`colless` computes the Colless' index of a tree and provides standardized values according to the Yule and PDA models.

Usage

```
colless(tree, norm = NULL)
```

Arguments

`tree`	An object of class `"treeshape"` on which the Colless' index is computed.
`norm`	A character string equals to `NULL` (default) for no normalization or one of `"pda"` or `"yule"`.

Details

The Colless' index Ic computes the sum of absolute values $|L - R|$ at each node of the tree where L (resp. R) is the size of the left (resp. right) daughter clade at the node.

The mean and standard deviation of the Colless's statistic Ic have been computed by Blum et al (2005). Under the Yule model the standardized index

$$Iyule = \frac{Ic - n * \log(n) - n(\gamma - 1 - \log(2))}{n}$$

converges in distribution (γ is the Euler constant). The limiting distribution is non Gaussian and is characterized as a functional fixed-point equation solution. Under the PDA model, the standardization is different

$$Ipda = \frac{Ic}{n^{3/2}}$$

and converges in distribution to the Airy distribution (See Flajolet and Louchard (2001)). Standardized indices are useful when one wishes to compare trees with different sizes. The `colless` function returns the value of the unnormalized index (default) or one of the standardized statistics (Yule or PDA).

Value

An object of class `numeric` which is the Colless' index of the tree.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier Francois <<olivier.francois@imag>>

References

Mooers, A. O. and Heard, S. B. (1997) Inferring Evolutionnary Process from Phylogenetic Tree Shape. *The Quarterly Review of Biology*, **72**, 31 – 54.

Blum, M., Francois, O. and Janson, S. The mean, variance and limiting distribution of two statistics sensitive to phylogenetic tree balance; manuscript available from http://www-timc.imag.fr/Olivier.Francois/bfj.pdf.

Flajolet, P. and Louchard, O. (2001) Analytic Variations on the Airy Distribution. *Algorithmica*, **31**, 361 – 377.

See Also

`sackin`

Examples

```
## Colless' index for a randomly generated PDA tree (unnormalized value)
tpda<-rtreeshape(1,tip.number=70,model="pda")
colless(tpda[[1]],norm="pda")

## Histogram of Colless' indices for randomly generated Yule trees
main="Colless' indices for randomly generated Yule trees"
xlab="Colless' indices"
hist(sapply(rtreeshape(300,tip.number=50,model="yule"),FUN=colless,norm="yule"),
      freq=FALSE,main=main,xlab=xlab)

## Change the number of tips
hist(sapply(rtreeshape(300,tip.number=100,model="yule"),FUN=colless,norm="yule"),
      freq=FALSE,main=main,xlab=xlab)
```

cutreeshape *Cut objects of class "treeshape"*

Description

Prunes or cuts an object of class "treeshape" from a specificed internal node, either by specifying a top or bottom direction. This function returns either the top part or the bottom part of a tree.

Usage

```
cutreeshape(tree, node, type)
```

Arguments

tree	An object of class "treeshape".
node	An integer representing the node at which the tree will be pruned or cut. node should be in the range $1, \ldots, treesize - 1$.
type	A character string equals to either "top" or "bottom".

Details

If the type argument is "top", the tree is pruned from node. The resulting tips correspond to the ancestral branches present at the same height as the given node. New tip labels are assigned to the tips.

If the type specified is "bottom", the subtree under node is returned. The tips are not renamed (they keep their former names) and the specified node is the root of the new tree.

Value

An object of class "treeshape"

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier François <<olivier.francois@imag.fr>>

See Also

tipsubtree

Examples

```
## Data set provided with the library. Type help(cytochromc) for more infos.
data(carnivora.treeshape)
data(hivtree.treeshape)

## Examples of "bottom" cutting:
bottom.tree=cutreeshape(carnivora.treeshape, 3, "bottom")
plot(carnivora.treeshape, bottom.tree)
bottom.tree=cutreeshape(carnivora.treeshape, 8, "bottom")
plot(carnivora.treeshape, bottom.tree)

## Examples of "top" pruning:
top.tree=cutreeshape(hivtree.treeshape, 158, "top")
plot(hivtree.treeshape, top.tree)
```

---

cytochromc *Phylogeny of the cytochrome C family.*

Description

An object of class "treeshape" containing the phylogeny of 50 species of cytochrome c.

---

hivtree.treeshape *Phylogenetic Tree of 193 HIV-1 Sequences*

Description

This data set describes an estimated clock-like phylogeny of 193 HIV-1. group M sequences sampled in the Democratic Republic of Congo. This data is the conversion of the data from the APE package into the class "treeshape".

Usage

```
data(hivtree.treeshape)
```

Format

hivtree.treeshape is an object of class "treeshape".

Source

This is a data example from Strimmer and Pybus (2001).

References

Strimmer, K. and Pybus, O. G. (2001) Exploring the demographic history of DNA sequences using the generalized skyline plot. *Molecular Biology and Evolution*, **18**, 2298 – 2305.

Examples

```
data("hivtree.treeshape")
summary(hivtree.treeshape)
plot(hivtree.treeshape)
```

`index.test` *Perform a test on the Yule or PDA hypothesis based on the Colless or the Sackin statistic*

Description

`colless.test` performs a test based on the Colless' index on tree data for the Yule or PDA model hypothesis.
`sackin.test` does the same with the Sackin's index.

Usage

```
colless.test(tree, model = "yule", alternative = "less", n.mc = 500)
sackin.test(tree, model = "yule", alternative = "less", n.mc = 500)
```

Arguments

`tree`	An object of class `"treeshape"`.
`model`	The null hypothesis of the test. One of `"yule"` (default) or `"pda"`.
`alternative`	A character string specifying the alternative hypothesis of the test. One of `"less"` (default) or `"greater"`.
`n.mc`	An integer representing the number of random trees to be generated and required to compute a p-value from a Monte Carlo method.

Details

A test on tree data that either rejects the Yule or the PDA models. This test is based on a Monte Carlo estimate of the p-value. Replicates are generated under the Yule or PDA models, and their Colless' (Sackin's) indices are computed. The empirical distribution function of these statistics is then computed thanks to the "ecdf" R function. The p-value is then deduced from its quantiles. The less balanced the tree is and the larger its Colless's (Sackin's) index. The alternative "less" should be used to test whether the tree is more balanced (less unbalanced) than predicted by the null model. The alternative "greater" should be used to test whether the tree is more unbalanced than predicted by the null model. The computation of p-values may take some time depending on the number of replicates |n.mc| and the size of the simulated tree.

Value

model	the null model.
statistic	the test statistic.
p.value	the p-value of the test.
alternative	the alternative hypothesis of the test.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr >>
Olivier Francois <<olivier.francois@imag.fr>>

References

Mooers, A. O., Heard, S. B. (Mar., 1997) Inferring Evolutionnary Process from Phylogenetic Tree Shape. *The Quarterly Review of Biology*, **72**, 31 – 54.

Blum, M., Francois, O. and Janson, S. The mean, variance and limiting distribution of two statistics sensitive to phylogenetic tree balance; manuscript available from http://www-timc.imag.fr/Olivier.Francois/bfj.pdf.

See Also

subtree.test
colless
sackin

Examples

```
## Test on a randomly generated Yule tree with 30 tips
a<-rtreeshape(1,30,model="yule")
a<-a[[1]]

## Is it more balanced than a Yule tree ?
colless.test(a,alternative="less",model="yule")
## Is it less balanced than a PDA tree ?
colless.test(a,model="pda",alternative="greater")

## Test on the phylogenetic tree hiv.treeshape: is it more balanced than
##      predicted by the Yule model?
data(hivtree.treeshape)
## The tree looks compatible with the null hypothesis
colless.test(hivtree.treeshape, alternative="greater", model="yule")

## What happen when we look at the top the tree?
colless.test(cutreeshape(hivtree.treeshape, 160, "top"),
      alternative="greater", model="yule")
```

```
colless.test(cutreeshape(hivtree.treeshape, 160, "top"),
      alternative="greater", model="pda")

## Test with the Sackin's index: is the HIV tree less balanced than
##      predicted by the PDA model?
sackin.test(hivtree.treeshape,alternative="greater",model="pda")
## The p.value equals to 1...
```

`likelihood.test` *Test the Yule model vs PDA (uniform) model.*

Description

`likelihood.test` uses the function `shape.statistic` to test the Yule model against the PDA model. The test is based on a Gaussian approximation for the log-ratio of likelihoods.

Usage

```
likelihood.test(tree, model = "yule", alternative="two.sided")
```

Arguments

`tree`	An object of class `"treeshape"` on which the test is performed.
`model`	The null hypothesis of the test. It must be equal to one of the two character strings `"yule"` or `"pda"`.
`alternative`	A character string specifying the alternative hypothesis of the test. Must be one of `"two.sided"` (default), `"less"` or `"greater"`.

Details

A test on tree data that either rejects the Yule of the PDA model. The test is based on the ratio of the likelihood of the PDA model to the likelihood of the Yule model (shape.statistic). The less balanced the tree is the larger its shape statistic is. The alternative "less" should be used to test whether the tree is less unbalanced than predicted by the null model. The alternative "greater" should be used to test whether the tree is more unbalanced than predicted by the null model.

Under the Yule model, the test statistic has approximate Gaussian distribution of $mean = 1.204 * n - \log n - 1 - 2$ and $variance = 0.168 * n - 0.710$, where n is the number of tips of the tree. The Gaussian approximation is accurate for n greater than 20.

Under the PDA model, the test statistic has approximate Gaussian distribution of $mean \sim 2.03 * n - 3.545 * \sqrt{n-1}$ and $variance \sim 2.45 * (n-1) * \log n - 1$, where n is the number of tips of the tree. The Gaussian approximation is however accurate for very large n (n greater than 10000(?)). The values of the means and variances have been obtained from an analogy with binary search tree models in computer science.

The function includes corrections for small sizes under the PDA model, and uses empirical values of variances estimated throught Monte Carlo replicates as follows

$$variance \sim 1.570 * n * \log n - 5.674 * n + 3.602 * \sqrt{n} + 14.915$$

Value

`likelihood.test` returns a list which includes:

`model`	the null model used by the test
`statistic`	the test statistic
`p.value`	the p.value of the test
`alternative`	the alternative hypothesis of the test

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier François <<olivier.francois@imag.fr>>

References

Fill, J. A. (1996), On the Distribution of Binary Search Trees under the Random Permutation Model. *Random Structures and Algorithms*, **8**, 1 – 25.

See Also

`shape.statistic`

Examples

```
## Generate a Yule tree with 150 tips. Is it likely to be fitting the PDA model?
likelihood.test(ryule(150),model="pda")
## The p.value is close from 0. We reject the PDA hypothesis.

## Test on the Carnivora tree: is it likely to be fitting the Yule model?
data(carnivora.treeshape)
likelihood.test(carnivora.treeshape)
## The p.value is high, so it's impossible to reject the Yule hypothesis.
```

`maxlik.betasplit` *Maximum likelihood of beta in the Beta-splitting model*

Description

The function finds the beta value that maximizes the likelihood in the Beta-splitting model. Beta=0 corresponds to Yule trees, Beta<0 corresponds to trees more unbalanced than Yule trees and Beta>0 corresponds to trees more balanced than Yule trees. Confidence intervals can also be provided.

Usage

```
maxlik.betasplit(phylo, up = 10, remove.outgroup = FALSE,
confidence.interval = "none", conf.level = 0.95, size.bootstrap = 100)
```

Arguments

`phylo`	An object of class `"treeshape"` or `"phylo"` on which the likelihood is computed.
`up`	numeric. The Beta value that maximizes the likelihood is searched between -2 and `up`
`remove.outgroup`	logical. Should one or two outgroups be removed before computing the likelihood
`confidence.interval`	The method to be used for computing confidence intervals. See *Details*.
`conf.level`	numeric. Probability associated with the confidence interval
`size.bootstrap`	number. Number of bootsrap replicates

Details

The beta-splitting model has been introduced by Aldous to simulate trees with different tree balance.

Beta=0 corresponds to Yule trees.

Beta<0 corresponds to trees more unbalanced than Yule trees, Beta=-3/2 corresponds to the PDA model.

Beta>0 corresponds to trees more balanced than Yule trees.

By default, `confidence.interval="none"` and no confidence interval is computed.

When `confidence.interval="bootstrap"`, a confidence interval is found with a resampling technique. Shall be used when the number of tips is small (<50).

When `confidence.interval="profile"`, a confidence interval is found with a profile likelihood technique. Shall be used when the number of tips is large (>50).

Value

`max_lik`	The Beta value that maximizes the likelihood
`conf_interval`	A confidence interval for `max_lik`

Author(s)

michael.blum@imag.fr

References

Aldous, D. J. (1996) Probability Distributions on Cladograms pp.1-18 of Random Discrete Structures eds D. Aldous and R. Pemantle, IMA Volumes Math. Appl. 76.

Aldous, D. J. (2001) Stochastic Models and Descriptive Statistics for Phylogenetic Trees, from Yule to Today. Statistical Science, *16*, 23 - 24.

Blum, M.G.B. and Francois, O. Which random processes describe the Tree of Life? A large-scale study of phylogenetic tree imbalance. Systematic Biology *55*, 685-691, 2006.

See Also

sackin, sackin.test, colless, colless.test

Examples

```
tree.pda<-rtreeshape(n=1, tip.number=50, model="pda")[[1]]
maxlik.betasplit(tree.pda,confidence.interval="none")
maxlik.betasplit(tree.pda,confidence.interval="bootstrap")
maxlik.betasplit(tree.pda,confidence.interval="profile")
```

---

plot.treeshape *Plot phylogenetic treeshapes.*

Description

Plot method for objects of class "treeshape".

Usage

```
## S3 method for class 'treeshape'
plot(x, y, ...)
```

Arguments

x	An object of class "treeshape".
y	An object of class "treeshape".
...	Further arguments to be passed to plot().

Details

If two trees are specified, they are plotted on the same window. This option is provided is order to facilitate the comparison between two trees.

Value

A null value is returned. Results are displayed on graphical window.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier Francois <<olivier.francois@imag.fr>>

See Also

plot for the basic plotting function in R

Examples

```
## Visual representation of the universal tree of life provided in data
data(universal.treeshape)
plot(universal.treeshape)

## Visual representation of two trees at the same time
data(carnivora.treeshape)
plot(carnivora.treeshape, cutreeshape(carnivora.treeshape, 8, "bottom"))
```

---

primates — *Phylogeny of the primates.*

Description

An object of class "treeshape" containing the phylogeny of the primates.

---

rhodopsin — *Phylogeny of rhodopsin proteins.*

Description

An object of class "treeshape" containing the phylogeny of 64 rhodopsin proteins.

`rtreeshape` *Generate a list of random binary trees according to a given model*

Description

This function generates a tree or a list of trees of class "treeshape" according to the Yule, PDA, Biased models. Speciation-specified models are also allowed.

Usage

```
rtreeshape(n, tip.number, p = 0.3, model="", FUN="")
```

Arguments

n	The number of trees to generate.
tip.number	The number of tips of the trees to generate. Can be a vector.
p	Only used when model="biased". It represents the bias factor of the tree to generate.
model	A character string equals to "yule", "pda", "aldous" or "biased".
FUN	A two variables (n and i) function.

Details

The "FUN" and "model" arguments cannot be specified at the same time. An error will be returned if both arguments are specified.

If tip.number is a vector, n trees will be generated for each size contained by tip.number

Q enables you to build trees of class "treeshape" according to the *Markov branching* model described by D. Aldous. Qn(i) is the probability that the left daughter clade of an internal node with n descendents contains i tips. The Qn(i) need not sum to one. Still, be carefull when you specify this distribution: computational errors may occur for complicated distributions and/or large trees.

The Yule model, also known as Markov model, can be described as follows. At any time, all the extant branches have the same probability to split into two subspecies.
The PDA model (Proportional to Distinguishable Arrangements) is not a model of growing tree. Instead, each tree with n tips has the same probability to be generated under this model. There is $(2n-3)!!$ possible trees with n tips.
The Biased model is a model of growing tree. When a species with speciation rate *r* splits, one of its descendent species is given the rate *pr* and the other is given the speciation rate $1-pr$ where p is a probability parameter. The Biased model was introduced by Kirkpatrick and Slatkin (1993). The Aldous' Branching (AB) model is defined by the following symmetric split distribution $q(n,i) = n/(2*h(n-1))*(1/(i(n-i)))$, where $h(n)$ is the nth harmonic number. The AB model is hardly motivated by biological considerations.

Value

A list of objects of class "treeshape" NULL if n=0

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier Franï¿½ois <<olivier.francois@imag.fr>>

References

Mooers, A. O. and Heard, S. B. (Mar., 1997), Inferring Evolutionnary Process from Phylogenetic Tree Shape. *The Quarterly Review of Biology*, **72**, 31-54, for more details about the Yule and PDA models.

Aldous, D. J. (1996), *Probability Distributions on Cladograms*. pp.1-18 of *Random Discrete Structures* eds D. Aldous and R. Pemantle, IMA Volumes Math. Appl. 76.

Kirkpatrick, M. and Slatkin, M. (1993) Searching for evolutionary patterns in the shape of a phylogenetic tree. *Evolution*, **47**, 1171 – 1181.

Examples

```
## Summary of a PDA tree with 100 tips:
summary(rtreeshape(n=1, tip.number=100, model="pda")[[1]])
## Summary of a Yule tree with 100 tips:
summary(rtreeshape(n=1, tip.number=100, model="yule")[[1]])

## Generate trees with different sizes
trees=rtreeshape(n=2, tip.number=c(10,20), model="yule")
length(trees)
plot(trees[[1]])
plot(trees[[2]])

## Histogram of Colless' indices for a list of 1000 PDA trees with 60 tips
hist(sapply(rtreeshape(1000,60,model="pda"),FUN=colless,norm="pda"),freq=FALSE)

## Histogram of shape statistics for a list of 1000 Yule trees with 100 tips
##      (takes some time to compute)
main="Histogram of shape statistics for a list of 1000 Yule trees"
hist(sapply(rtreeshape(1000,100,model="yule"),FUN=shape.statistic,norm="yule"),
      freq=FALSE, main=main)
## It should be a gaussian with mean 0 and standard deviation 1.
x<-seq(-4,4,by=0.01)
lines(x,dnorm(x))

## Building a tree using Markov splitting model
Q <- function(n,i) (i==1)

tree=rtreeshape(n=1, tip.number=10, FUN=Q)
plot(tree[[1]])
```

sackin *Compute the Sackin's index of a tree*

Description

sackin computes the Sackin's index on tree and normalizes it.

Usage

```
sackin(tree, norm = NULL)
```

Arguments

tree	An object of class "treeshape" on which the Sackin's index is computed.
norm	A character string equals to "null" (default) for no normalization, "pda" for the PDA normalization or "yule" for the Yule normalization.

Details

The Sackin's index is computed as the sum of the number of ancestors for each tips of the tree. The less balanced a tree is and the larger its Sackin's index. It can be normalized in order to obtain a statistic that does not depend on the tree size, and so compare trees with different sizes. The normalization depends on the reference model (Yule or PDA). Under the Yule model, the normalized index is

$$Iyule = \frac{Is - 2n * \sum_{j=2}^{n} \frac{1}{j}}{n}$$

where Is is the non-normalized Sackin's index for a n-tips tree. Under the PDA model, the normalized index is

$$Ipda = \frac{Is}{n^{3/2}}$$

See details on the Colless index.

Value

An object of class numeric which contains the Sackin's index of the tree.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier François <<olivier.francois@imag.fr>>

References

Mooers, A. O., Heard, S. B. (Mar., 1997) Inferring Evolutionnary Process from Phylogenetic Tree Shape. *The Quarterly Review of Biology*, **72**, 31 – 54, for more details about the Sackin'index and its signification about the balance of phylogenetic trees.

Blum, M., Francois, O. and Janson, S. The mean, variance and limiting distribution of two statistics sensitive to phylogenetic tree balance; manuscript available from http://www-timc.imag.fr/Olivier.Francois/bfj.pdf.

See Also

colless
sackin.test

Examples

```
## Index of Sackin of a PDA tree :
tpda<-rtreeshape(1,tip.number=70,model="pda")
tpda<-tpda[[1]]
sackin(tpda,norm="pda")

## Histogram of the Sackin's indices for randomly generated Yule trees,
##      with no normalization
main="Histogram of Sackin's indices for randomly generated Yule trees"
xlab="Sackin's index"
hist(sapply(rtreeshape(300,tip.number=50,model="yule"),FUN=sackin,norm="yule"),
      freq=FALSE, main=main, xlab=xlab)

## Change the size of the trees:
hist(sapply(rtreeshape(300,tip.number=100,model="yule"),FUN=sackin,norm="yule"),
      freq=FALSE, main=main, xlab=xlab)
```

---

shape.statistic *Computes the log of the likelihood ratio (yule/pda)*

Description

shape.statistic computes the logarithm of the ratio of the likelihoods under the Yule model and the PDA model of the given tree.

Usage

```
shape.statistic(tree, norm=NULL)
```

Arguments

tree	An object of class "treeshape".
norm	A character string equals to NULL for no normalization, "yule" for the Yule model normalization or "pda" for the pda normalization.

Details

The log of the likelihood ratio is proportional to

$$\sum (\log N(v) - 1),$$

for all internal node v (where $N(v)$ is the number of internal nodes descending from the node v). The ratio of the likelihoods enables to build the most powerful test of the Yule model against the PDA one. (Neyman-Pearson lemma).

Under the PDA model, the log ratio has approximate Gaussian distribution of $mean \sim 2.03 * n - 3.545 * \sqrt{n-1}$ and $variance \sim 2.45 * (n-1) * \log n - 1$, where n is the number of tips of the tree. The Gaussian approximation is accurate for very large n (n greater than 10000(?)). The normalization of the ratio uses tabulated empirical values of variances estimated from Monte Carlo simulations. The normalization uses the formula:

$$variance \sim 1.570 * n * \log n - 5.674 * n + 3.602 * \sqrt{n} + 14.915$$

Under the Yule model, the log ratio has approximate Gaussian distribution of $mean = 1.204 * n - \log n - 1 - 2$ and $variance = 0.168 * n - 0.710$, where n is the number of tips of the tree. The Gaussian approximation is accurate for n greater than 20.

Value

An object of class `numeric` containing the shape statistic of the tree.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier François <<olivier.francois@imag.fr>>

References

Fill, J. A. (1996), On the Distribution of Binary Search Trees under the Random Permutation Model. *Random Structures and Algorithms*, **8**, 1 – 25, for more details about the normalization and proofs.

Examples

```
data(universal.treeshape)
tree <- universal.treeshape
plot(tree)
summary(tree)

likelihood.test(tree,  model = "yule", alternative = "two.sided")
likelihood.test(tree,  model = "pda", alternative = "two.sided")

## Histogram of shape statistics for a list of Yule trees
##      (may take some time to compute)
main="Histogram of shape statistics"; xlab="shape statistic"
hist(sapply(rtreeshape(1000,tip.number=100,model="yule"),FUN=shape.statistic,
```

```
      norm="yule"), freq=FALSE, main=main, xlab=xlab)

## Does it fit the Gaussian distribution with mean=0 and sd=1 ?
x<-seq(-3,3,by=0.001)
lines(x,dnorm(x))
```

`shift.test` *Testing diversification rate variation in phylogenetic trees*

Description

A statistical test of diversification rate shift within an evolutionary process.

Usage

```
shift.test(tree, node, lambda1, lambda2, nrep, silent)
```

Arguments

`tree`	An object of class `treeshape`.
`node`	An integer value that represents the internal node at which the test is done. It ranges from 1 to the number of tips minus 2.
`lambda1`	A positive numerical value (object of class `numeric`) that represents an ancestral diversification rate in the lineage ending at `node`.
`lambda2`	A positive numerical value (object of class `numeric`) that represents the shifted diversification rate after the speciation event at `node`.
`nrep`	Number of Monte-carlo replicates in the computation of the P-value.
`silent`	A boolean indicating whether the test is silent or verbose.

Details

This function implements a test of diversification rate shift based on the Delta1 statistic (Moore et al. 2004). We introduced some simplifications of the test statistic using basic results in branching process.

Value

The function provides textual results and a P-value for the null hypothesis of no diversification rate shift against a (lambda2/lambda1)-fold shift at the node under consideration.

Author(s)

Eric Durand <eric.durand@imag.fr>
Olivier Francois <olivier.francois@imag.fr>

References

Moore, B.R., Chan, K.M.A, and Donoghue M.J. (2004) *Detecting diversification rate variation in supertrees*. In Bininda-Emonds, O. R. P. (ed.) *Phylogenetic Supertrees: Combining Information to Reveal the Tree of Life*, pp. 487-533. Computational Biology, volume 3 (Dress, A., series ed.).

Examples

```
## Detecting diversification rate variation in bird families (135 tips)
 data(bird.families)
 tree.birds <- as.treeshape(bird.families, model = "yule")
 class(tree.birds) <- "treeshape"
 pv <- sapply(1:135, FUN = function(i) shift.test(tree.birds, i, lambda1 = 1, lambda2 = 100, nrep = 1000, silent =

## Significant shifts detected at nodes = 67 and 78
 pv[c(67,78)]
 shift.test(tree.birds, node = 67, lambda1 = 1, lambda2 = 100, nrep = 10000, silent = TRUE)
 shift.test(tree.birds, node = 78, lambda1 = 1, lambda2 = 100, nrep = 10000, silent = TRUE)

  ## visualize the shifts
 par(mfrow=c(2,1))
 plot(cutreeshape(tree.birds, ancestor(tree.birds, 67) , "bottom"))
 plot(cutreeshape(tree.birds, 78 , "bottom"))
```

`smaller.clade.spectrum`

Compute the smaller clade spectrum of a tree.

Description

`smaller.clade.spectrum` returns a n*2 matrix where n is the number of internal nodes of the tree. For each i in 1:n, the [i,1] element of the matrix is the size of the clade rooted at the 'i'th node of the tree. [i,2] is the size of the smaller daughter clade of the 'i'th node of the tree.

Usage

```
smaller.clade.spectrum(tree)
```

Arguments

`tree`	An object of class `"treeshape"`.

Value

A n*2 matrix (where n is the number of internal nodes of the tree) containing the size of the clades and the smaller clades. `smaller.clade.spectrum(tree)[1,1]` contains the number of tips of the tree. `smaller.clade.spectrum(tree)[i,1]` contains the number of tips of the subtree whose root is the node number `n-i+1`. `smaller.clade.spectrum(tree)[1,2]` contains the number of tips of the smaller daughter clade at the root.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier Francois <<olivier.franois@imag.fr>>

See Also

spectrum.treeshape

Examples

```
# computes the log-likelihood for Aldous' model
shape.new <- function(tree){
h <- function(n){sum(1/(1:n))}
mat <- smaller.clade.spectrum(tree)
parent <- mat[,1]
daughter <- mat[,2]
nh.n <- sapply(parent, FUN = function(x){x*h(x-1)} )
s <- sum(log(daughter/parent) + log(1 - daughter/parent) + log(nh.n))
return(s)}

# distribution over 200 replicates

tr <- rtreeshape(200, 100, FUN =
function(n,i){if((i>0)&(i<n))return(1/i/(n-i)) else return(0)})
res <- sapply( tr, FUN = shape.new)
hist(res)
```

---

spectrum.treeshape *Compute the spectrum of a tree*

Description

This function returns a sequence containing the number of subtrees of size n, n-1, ..., 3, 2 where n is the size of the tree. The 'k'th element of the sequence is the number of subtrees of size n-k+1 in the tree, where n is the number of tips of the tree.

Usage

```
spectrum.treeshape(tree)
```

Arguments

tree	An object of class "treeshape".

Value

A sequence of size n-1 (where n is the number of tips of the tree) containing the number of subtrees of each size. `spectrum.treeshape(tree)[1]` is the number of subtrees with n tips (equal to 1). `spectrum.treeshape(tree)[n-1]` is the number of cherries of the tree (subtrees with 2 tips).

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier Francois <<olivier.francois@imag.fr>>

See Also

`smaller.clade.spectrum`

Examples

```
## A random Yule tree with 30 tips
tr<-rtreeshape(n=1,tip.number=30,model="yule")
tr<-tr[[1]]
spectre=spectrum.treeshape(tr)
spectre

## Number of cherries of the tree : nrow(tr$merge)==29
spectre[29]
```

---

subtree.test — *Test the Yule or PDA hypothesis*

Description

`subtree.test` tests the likelihood of the Yule or the PDA hypothesis and computes the p-value of the test. The test is based on the number of subtrees of a given size in the tree.

Usage

```
subtree.test(tree, size , alternative = "two.sided")
```

Arguments

`tree`	An object of class `"treeshape"`.
`size`	The size of the subtrees to test for.
`alternative`	The alternative hypothesis of the test. It can be `"two.sided"` (default), `"less"` or `"greater"`.

Details

See references for the mathematical details of the test. It uses a Gaussian approximation to compute the p-value.

Value

A list containing the following arguments :

`statistic`	the value of the statistic used in the test
`p.value`	the p-value of the test
`alternative`	the alternative hypothesis used for the test

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier Francois <<olivier.francois@imag.fr>>

References

Blum, M. and François, O. (2005) Minimal clade size and external branch length under the neutral coalescent *Advances in Applied Probability* (2005), **37.3**.

See Also

`sackin.test`
`colless.test`

Examples

```
## Generate a random pda tree with 50 tips
tr<-rtreeshape(n=1,tip.number=50,model="pda")
tr<-tr[[1]]

## Test the yule hypothesis, using subtrees of size 2 (Cherries),
##      with the alternative hypothesis "less"
subtree.test(tr,size=2,alternative="less")
```

summary.treeshape *Print a summary of an object of class "treeshape"*

Description

This function prints a compact summary of a phylogenetic tree of class "treeshape")

Usage

```
## S3 method for class 'treeshape'
summary(object, ...)
```

Arguments

object	an object of class "treeshape".
...	further arguments passed to or from other methods.

Details

summary.treeshape prints the following information: the number of tips of the tree, its Colless' index, and the expected values and standard deviations of the Colless' index under the PDA and Yule models.The expected value of the Colless' index under the Yule model is given according to the formula: $n * \log n + n * (\gamma - 1 - \log 2)$ where n is the number of tips of the tree and γ the Euler's constant. The standard deviation under the Yule model is given by: $\sqrt{(3 - \frac{\pi^2}{6} - \log 2) * n}$. The expected value of the Colless' index under the PDA model is given according to the formula: $\sqrt{\pi} * n^{3/2}$. The standard deviation under the PDA model is given by: $\sqrt{\frac{10}{3} - \pi} * n^{3/2}$.

Value

A NULL value is returned, the results are simply printed.

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier François <<olivier.francois@imag.fr>>

See Also

summary
colless for more informations about the expected values under the Yule and PDA models.

Examples

```
## Summary of a PDA tree with 100 tips.
summary(rpda(100))
## Note that the standard deviation is very large.

## Summary of a Yule tree with 100 tips.
summary(ryule(100))
## The standard deviation under the Yule model is much smaller than under
##      the PDA model.

## Summary of the HIV tree.
data(hivtree.treeshape)
summary(hivtree.treeshape)
## The HIV tree is much closer from the Yule model than from the PDA model.
```

`tipsubtree` *Extract a subtree that contains pre-specified tip names or labels*

Description

`tipsubtree` returns an object of class `"treeshape"` that contains pre-specified tip names or labels. The name of the tips are conserved. It extracts the smallest tree that contains the common ancestors of the given tips.

Usage

```
tipsubtree(tree, tips, numeric=FALSE)
```

Arguments

`tree`	An object of class `"treeshape"`.
`tips`	A vector that contains the names of the tips one want to keep. Warning, the names are case-sensitive.
`numeric`	An object of class `"logical"`. If `FALSE`, the vector `tips` contains the names of the tips. If `TRUE`, it contains the labels of the tips.

Value

An object of class `"treeshape"`

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier François <<olivier.francois@imag.fr>>

See Also

cutreeshape

Examples

```
## The universal tree of life provided in the data sets.
data(universal.treeshape)

## One might want to extract the tree containing the Animals, the Plants,
##      the Aquifex and the Microsporidia
tree1<-tipsubtree(universal.treeshape,tips=c("Animals", "Aquifex",
      "Microsporidia", "Plants"))
plot(universal.treeshape, tree1)

## Labels that do not appear in the tree are ignored
tree2<-tipsubtree(universal.treeshape,tips=c("Human", "Animals", "Aquifex",
      "Microsporidia", "Plants"))
plot(universal.treeshape, tree2)

tree3<-tipsubtree(universal.treeshape,tips=c(1,3,7), numeric=TRUE)
plot(universal.treeshape, tree3)
```

---

treeshape *Builds an object of class treeshape*

Description

treeshape builds a tree of class "treeshape" from a n*2 matrix, where n is the number of internal nodes of the tree. There are no informations about the heights of the branches in an object of class "treeshape". Formally, a "tree shape" is a phylogenetic tree where the label of the tips are ignored. Here, the label of the tips can be kept or ignored. If a names vector is provided, then the names of species are attached to the tips of the tree. Otherwise, tips are simply labeled with their numbers in the tree. Trees of class "treeshape" are always binary.

Usage

```
treeshape(nodes, names)
```

Arguments

nodes	nodes is a n*2 matrix containing the node structure of the tree.
names	names is a vector which contains the names of the tips.

Details

A tree of class "treeshape" is a fully dichotomous binary tree. The purpose of the class "treeshape" is to study the topology of phylogenetic trees. The heights of branches are not provided for a tree of that class because we mainly focus on the balance aspect of the trees. The 'i'th row of the nodes matrix represents the children of the node number i in the tree (nodes[i,1] being the left child, and nodes[i,2] being the right child). A positive value represents an internal node, while a negative one stands for a tip of the tree. The last row always represents the children of the root of the tree.

Value

An object of class "treeshape"

Author(s)

Michael Blum <<michael.blum@imag.fr>>
Nicolas Bortolussi <<nicolas.bortolussi@imag.fr>>
Eric Durand <<eric.durand@imag.fr>>
Olivier François <<olivier.francois@imag.fr>>

References

Semple, C. and Steel, M. (2003) Phylogenetics. *Oxford Lecture Series In Mathematics and its Applications*, **24**, for the mathematical definitions and tools for phylogenetic trees.

See Also

rtreeshape

Examples

```
## Nodes will define the nodes of a five tips tree
nodes<-matrix(nrow=4,ncol=2)
nodes[1,]<-c(-5,-4)
nodes[2,]<-c(1,-1)
nodes[3,]<-c(-3,2)
nodes[4,]<-c(-2,3)

## Now we can build the tree and plot it.
tree1<-treeshape(nodes)
plot(tree1)

## Computation of the sackin index for the tree :
sackin(tree1)

## Label will define the names of the tips
label=c("a", "b", "c", "d", "e")
tree2<-treeshape(nodes, label)
plot(tree1, tree2)
```

universal.treeshape *Universal phylogenetic tree of life*

Description

This data set describes the Universal Tree of Life described by CR Woese, in the `treeshape` class. "The universal phylogenetic tree not only spans all extant life, but its root and earliest branchings represent stages in the evolutionary process before modern cell types had come into being."

Usage

```
data(universal.treeshape)
```

Format

`universal.treeshape` is an object of class `"treeshape"`.

References

Woese, C. R. (July, 2000) *Interpreting the universal phylogenetic tree*, **97**, 392 – 8396 (Proc. Natl. Acad. Sci. USA).

Examples

```
## Example tree in "treeshape" format
data("universal.treeshape")

## Summary of the tree
summary(universal.treeshape)

## Visual representation of the tree
plot(universal.treeshape)
```

Index

∗Topic **datagen**
- rtreeshape, 20
- treeshape, 32

∗Topic **datasets**
- carnivora.treeshape, 7
- cytochromc, 12
- hivtree.treeshape, 12
- primates, 19
- rhodopsin, 19
- universal.treeshape, 34

∗Topic **hplot**
- plot.treeshape, 18

∗Topic **htest**
- aldous.test, 2
- index.test, 13
- likelihood.test, 15
- maxlik.betasplit, 17
- shift.test, 25
- subtree.test, 28

∗Topic **manip**
- all.equal.treeshape, 4
- as.phylo.treeshape, 5
- as.treeshape, 6
- cutreeshape, 11
- summary.treeshape, 30
- tipsubtree, 31

∗Topic **univar**
- cladesize, 8
- colless, 9
- sackin, 22
- shape.statistic, 23
- smaller.clade.spectrum, 26
- spectrum.treeshape, 27

aldous.test, 2
all.equal, *4*
all.equal.treeshape, 4
as.phylo, *6*
as.phylo.treeshape, 5, *7*
as.treeshape, *6*, 6
carnivora.treeshape, 7
cladesize, 8
colless, 9, *14*, *18*, *23*, *30*
colless.test, *18*, *29*
colless.test *(*index.test*)*, 13
cutreeshape, 11, *32*
cytochromc, 12

hivtree.treeshape, 12

index.test, 13

likelihood.test, 15

maxlik.betasplit, 17

plot, *19*
plot.treeshape, 18
primates, 19

rhodopsin, 19
rtreeshape, 20, *33*

sackin, *10*, *14*, *18*, 22
sackin.test, *18*, *23*, *29*
sackin.test *(*index.test*)*, 13
shape.statistic, *16*, 23
shift.test, 25
smaller.clade.spectrum, 26, *28*
spectrum.treeshape, *27*, 27
subtree.test, *14*, 28
summary, *30*
summary.treeshape, 30

tipsubtree, *11*, 31
treeshape, 32

universal.treeshape, 34