Package 'hutan'

June 11, 2020

Title A Collection of Tools for Phylogenetic Tree Manipulation

Version 0.5.1

Date 2017-07-23

Description A collection of tools for phylogenetic tree manipulation. It is named after the Indonesian word for forest.

Depends R (>= 3.1.0)

License MIT + file LICENSE

LazyData true

Collate 'utility_functions.R'
'hutan.R'
'siphonophore_constraint.R'
'siphonophore_ml.R'

Imports ape (>= 3.3),
tidyverse,
magrittr,
geiger,
phytools

Suggests testthat,
roxygen2,
knitr,
rmarkdown

VignetteBuilder knitr

RoxygenNote 7.1.0

R topics documented:

`ancestral_edges` *For a node in a tree, get a vector of the edges between the node and the root*

Description

For a node in a tree, get a vector of the edges between the node and the root

Usage

```
ancestral_edges(phy, node)
```

Arguments

phy	A phylo object
node	A node number

Value

A vector of edge numbers

are_bipartitions_compatible

Check if two bipartitions drawn from trees with the same tips are compatible with each other. Each bipartition is defined as a vector of the names of the tips on one side of the bipartition.

Description

Check if two bipartitions drawn from trees with the same tips are compatible with each other. Each bipartition is defined as a vector of the names of the tips on one side of the bipartition.

Usage

```
are_bipartitions_compatible(bi1, bi2, phy)
```

Arguments

bi1	The first bipartition.
bi2	The second bipartition.
phy	A phylo object describing a tree that includes all tips under investigation. This is used to infer the other half of each bipartition.

Value

TRUE if bi1 is compatible with bi2, otherwise FALSE.

bipartition_for_edge

Get a bipartition, described as a vector of tip numbers, from a specified tree and edge number.

Description

Get a bipartition, described as a vector of tip numbers, from a specified tree and edge number.

Usage

```
bipartition_for_edge(phy, edge)
```

Arguments

phy	A phylo object that specifies the tree.
edge	The number of the edge that defines the bipartition.

Value

A vector of tip nodes (specified by numbers) that define one half of the bipartition (the other half is the set of tip nodes that are not in this vector).

`bipartition_for_edge_by_label` *Get a bipartition, described as a vector of tip labels, from a specified tree and edge number.*

Description

Get a bipartition, described as a vector of tip labels, from a specified tree and edge number.

Usage

```
bipartition_for_edge_by_label(edge, phy)
```

Arguments

`edge`	The number of the edge that defines the bipartition.
`phy`	A phylo object that specifies the tree.

Value

A vector of tip nodes (specified by labels) that define one half of the bipartition (the other half is the set of tip nodes that are not in this vector).

`calc_diffs` *For each internal node, calculate the difference in state values at the two child nodes*

Description

For each internal node, calculate the difference in state values at the two child nodes

Usage

```
calc_diffs(phy, states)
```

Arguments

`phy`	An ape::phylo object
`states`	A vector with length equal to nodes with the state at each node

Value

A vector of values corresponding to each internal node with the value difference

`compatible_edges` *Identify the edges in one phylo object that are compatible with the edges in another phylo object. Requires the same tip labels for each tree.*

Description

Identify the edges in one phylo object that are compatible with the edges in another phylo object. Requires the same tip labels for each tree.

Usage

```
compatible_edges(phy1, phy2)
```

Arguments

phy1	The tree under consideration
phy2	The tree to be compared to

Value

A boolean vector corresponding to the edges in phy1. Each element is FALSE if the edge is iscompatible with phy2, or TRUE if compatible.

`connecting_edges` *For two nodes in a tree, get a vector of the edges that connect them*

Description

For two nodes in a tree, get a vector of the edges that connect them

Usage

```
connecting_edges(phy, node_a, node_b)
```

Arguments

phy	A phylo object
node_a	Number of the first node
node_b	Number of the second node

Value

A vector of edge numbers

cut_tree — *Cuts a single tree on the branch subtending a specified node*

Description

Cuts a single tree on the branch subtending a specified node

Usage

```
cut_tree(phy, x)
```

Arguments

phy	The tree to be cut, as an ape phylo object
x	An internal node number. The tree phy will be cut on the branch that subtends this nodes.

Value

A list of phylo objects that are the subtrees

decompose_tree — *Decomposes a single tree into a series of subtrees designated by internal node numbers*

Description

Decomposes a single tree into a series of subtrees designated by internal node numbers

Usage

```
decompose_tree(phy, x)
```

Arguments

phy	The tree to be decomposed, as an ape phylo object
x	A vector of internal node numbers. The tree phy will be cut on each branch that subtends each of these nodes.

Value

A list of phylo objects

descendants *Get all the descendants of a given node in a tree.*

Description

Get all the descendants of a given node in a tree.

Usage

```
descendants(phy, a, keep_node = FALSE)
```

Arguments

phy	A phylo object that specifies the tree.
a	The number of a node in phy.
keep_node	If FALSE, do not include a in the result.

Value

A vector of nodes (specified by number) that are descendants of a. Includes internal and tip nodes.

difference_from_calibrated *Assesses how much phy deviates from an ultrametric tree*

Description

Assesses how much phy deviates from an ultrametric tree

Usage

```
difference_from_calibrated(phy, model = "discrete", ...)
```

Arguments

phy	A phylo object
model	The model used for fitting. "discrete" is used by default for speed
...	Additional chronos arguments

Value

The sum of absolute changes in branch length required to make an ape::chronos time calibrated tree, normalized by the total branch length of the calibrated tree. The higher the value, the more the tree deviates from the calibrated tree.

`distance_from_tip` *For each node in the tree (including internal nodes and tips) get the shortest distance to a descendant node. Values for tip nodes should be 0.*

Description

For each node in the tree (including internal nodes and tips) get the shortest distance to a descendant node. Values for tip nodes should be 0.

Usage

```
distance_from_tip(phy)
```

Arguments

phy	A phylo object

Value

A vector with elements corresponding to each node

`extend_terminal_branches` *Extends each terminal branch by specified length*

Description

Extends each terminal branch by specified length

Usage

```
extend_terminal_branches(phy, x)
```

Arguments

phy	A phylogeny in ape::phylo format
x	Amount to extend each branch by

Value

A phylogeny in ape::phylo format

`flip_bipartition` *Given a tree and a bipartition, described as a vector of tip labels on one side of of the bipartition, return the same bipartition but defined by the tip labels on the other side of the bipartition.*

Description

Given a tree and a bipartition, described as a vector of tip labels on one side of of the bipartition, return the same bipartition but defined by the tip labels on the other side of the bipartition.

Usage

```
flip_bipartition(phy, bi)
```

Arguments

`phy`	A phylo object that specifies the tree.
`bi`	The bipartition.

Value

A vector of tip nodes (specified by labels) that define one half of the bipartition (the other half is the set of tip nodes that are provided as bi).

`generate_constaint_tree` *Generates a tree with a single resolved bipartition between two sets of tip names. Useful for generating constraint trees.*

Description

Generates a tree with a single resolved bipartition between two sets of tip names. Useful for generating constraint trees.

Usage

```
generate_constaint_tree(tips1, tips2)
```

Arguments

`tips1`	A vector of tip names for clade 1
`tips2`	A vector of tip names for clade 2

Value

A phylo object

Examples

```
library( ape )
tips1 = c("a", "b", "c")
tips2 = c("d", "e", "f")
ctree = generate_constaint_tree( tips1, tips2 )
```

`get_bipartitions` *Get a list of all the bipartitions in a tree.*

Description

Get a list of all the bipartitions in a tree.

Usage

```
get_bipartitions(phy)
```

Arguments

phy	A phylo object that specifies the tree.

Value

A list of bipartitions for the tree. The order of the list corresponds to the edges in phy$edge. Bipartitions are specified as a vector of the tip labels that make up one half of the bipartition.

`get_corresponding_nodes` *Given two trees phy1 and phy2 with the same topology and tip labels, get a vector that indicates which node numbers in phy2 correspond to the nodes in phy1*

Description

Given two trees phy1 and phy2 with the same topology and tip labels, get a vector that indicates which node numbers in phy2 correspond to the nodes in phy1

Usage

```
get_corresponding_nodes(phy1, phy2)
```

Arguments

phy1	A phylo object
phy2	A phylo object

Value

A numeric vector in the order of nodes in phy1, providing corresponding node numbers from phy2

`hutan` *hutan: A collection of tools for phylogenetic tree manipulation.*

Description

The hutan package provides functions for common phylogenetic tree manipulation tasks, and uses these facilitate some more specialized tasks. It is named after the Indonesian word for forest.

`is_compatible_with_set` *Check if bipartition bi is compatible with the bipartitions in bi_list. Each bipartition is defined as a vector of the names of the tips on one side of the bipartition.*

Description

Check if bipartition bi is compatible with the bipartitions in bi_list. Each bipartition is defined as a vector of the names of the tips on one side of the bipartition.

Usage

```
is_compatible_with_set(bi, bi_list, phy)
```

Arguments

`bi`	The query bipartition.
`bi_list`	A list of the bipartitions to be compared against.
`phy`	A phylo object describing a tree that includes all tips under investigation. This is used to infer the other half of each bipartition.

Value

TRUE if bi is compatible with all bipartition in bi_list, otherwise FALSE.

`is_monophyletic` *Test if a set of tips, specified as a vector of tip labels, forms a monophyletic group in a given tree. The test is unrooted, i.e. the group can span the root.*

Description

Test if a set of tips, specified as a vector of tip labels, forms a monophyletic group in a given tree. The test is unrooted, i.e. the group can span the root.

Usage

```
is_monophyletic(phy, x)
```

Arguments

phy	The tree under consideration
x	A vector of the labels of the tips in question

Value

A boolean, TRUE if the tips form a monophyletic group.

`picx` *Estimate the extended phylogenetic independent contrast. Rather than normalize differences across nodes by branch lengths, differences are normalized be the expected difference obtained from replicate simulations. This allows for greater model flexibility.*

Description

Estimate the extended phylogenetic independent contrast. Rather than normalize differences across nodes by branch lengths, differences are normalized be the expected difference obtained from replicate simulations. This allows for greater model flexibility.

Usage

```
picx(
  x,
  phy,
  var.contrasts = FALSE,
  model_method = "BM",
  model_parameters = NA,
  n_replicates = 200
)
```

Arguments

x	A numeric vector with one trait value per tip
phy	An ape::phylo object
var.contrasts	logical, indicates whether the expected variances of the contrasts should be returned
model_method	The model of trait evolution. Can be one of c("BM", "OU")
model_parameters	Model parameters from fitContinuous. Will estimate if not provided.
n_replicates	The number of simulations used to estimate the expected differences

Value

A vector of phylogenetically independent contrasts

`safe.drop.tip` *Drops specified tips from a phylogeny. Like ape's drop.tip(), but it works when only a single tip is to be retained.*

Description

Drops specified tips from a phylogeny. Like ape's drop.tip(), but it works when only a single tip is to be retained.

Usage

```
safe.drop.tip(phy, tip)
```

Arguments

`phy`	The tree, as an ape phylo object
`tip`	A vector of tip numbers to be removed.

Value

The reduced tree, as a phylo object

`sim_diffs` *For each internal node, simulate the difference between state values at the node's children*

Description

For each internal node, simulate the difference between state values at the node's children

Usage

```
sim_diffs(phy, model_parameters, model_method = "BM")
```

Arguments

`phy`	An ape::phylo object
`model_parameters`	Parameter estimates for evolutionary model
`model_method`	The model of trait evolution. Can be one of c("BM","OU")

Value

The simulated child differences for each internal node

siphonophore_constraint *Siphonophores constraint phylogeny.*

Description

An unresolved phylogeny that constrains the group Agalmatidae sensu stricto + Bargmannia to be monophyletic, corresponding to the published SOWH test

Usage

```
siphonophore_constraint
```

Format

An ape phylo object

Source

http://dx.doi.org/10.1080/10635150500354837

siphonophore_ml *Siphonophores phylogeny.*

Description

A maximum likelihood phylogeny of siphonophores

Usage

```
siphonophore_ml
```

Format

An ape phylo object

Source

http://dx.doi.org/10.1080/10635150500354837

`slide_root_edges` *Repartitions lengths along edges that descend from root node so that they are equal. Useful after rooting operations that result in branches with 0 length*

Description

Repartitions lengths along edges that descend from root node so that they are equal. Useful after rooting operations that result in branches with 0 length

Usage

```
slide_root_edges(phy)
```

Arguments

phy	A phylo object

Value

A phylo object with modified edge lengths

`tips` *Get tips and labels of a phylo object.*

Description

Get tips and labels of a phylo object.

Usage

```
tips(phy)
```

Arguments

phy	A phylo object.

Value

A vector of all the tips, annotated with their names

tip_descendants — *Get all the tips that are descendants of a given node in a tree.*

Description

Get all the tips that are descendants of a given node in a tree.

Usage

```
tip_descendants(phy, a)
```

Arguments

phy	A phylo object that specifies the tree.
a	The number of a node in phy.

Value

A vector of tip nodes (specified by number) that are descendants of a. If a is a tip, it is the sole element of this vector.

zero_constrained — *Generates the "zero-constrained" tree described by Susko 2014 (http://dx.doi.org/10.1093/molbev/msu039)*

Description

Generates the "zero-constrained" tree described by Susko 2014 (http://dx.doi.org/10.1093/molbev/msu039)

Usage

```
zero_constrained(phy_resolved, phy_constraint, epsilon = 1e-06)
```

Arguments

phy_resolved	A fully resolved phylogeny stored as a phylo object, e.g. an ML tree.
phy_constraint	A partially resolved constraint tree.
epsilon	The value to replace the branch length with

Value

A phylo object containing a tree that is the same as phy_resolved, except that the length of edges that are incompatible with phy_constraint are replaced with epsilon.

Examples

```
data( siphonophore_ml )
data( siphonophore_constraint )
zc <- zero_constrained( siphonophore_ml, siphonophore_constraint )
```