

Package ‘FossilSim’

February 21, 2019

Type Package

Title Simulation of Fossil and Taxonomy Data

Version 2.1.1

Description Simulating taxonomy and fossil data on phylogenetic trees under mechanistic models of speciation, preservation and sampling.

License GPL-3

LazyData TRUE

RoxygenNote 6.1.1

Imports ape, TreeSim,

Suggests paleotree, knitr, rmarkdown, devtools

VignetteBuilder knitr

Language en-GB

NeedsCompilation no

Author Rachel Warnock [aut, cph],
Joelle Barido-Sottani [aut, cre, cph],
Walker Pett [aut, cph],
O'Reilly Joseph [aut, cph]

Maintainer Joelle Barido-Sottani <joelle.barido-sottani@m4x.org>

Repository CRAN

Date/Publication 2019-02-21 18:00:04 UTC

R topics documented:

beast.fbd.format	3
count.fossils	4
count.fossils.binned	4
fossils	5
fossils.to.BEAST.constraints	5
fossils.to.BEAST.start.tree	6
fossils.to.paleotree.record	7
fossils.to.pyrate	8

FossilSim	9
get.tip.descs	11
paleotree.record.to.fossils	11
place.fossils	12
plot.fossils	13
plot.taxonomy	15
prune.fossil.tips	16
prune.fossils	17
rangeplot.asymmetric	18
reconcile.fossils.taxonomy	19
remove.stem.fossils	20
remove.stem.lineages	20
sampled.tree.from.combined	21
SAtree	22
SAtree.from.fossils	22
sim.anagenetic.species	23
sim.cryptic.species	24
sim.extant.samples	24
sim.fbd.age	25
sim.fbd.rateshift.taxa	26
sim.fbd.taxa	27
sim.fossils.environment	28
sim.fossils.intervals	30
sim.fossils.poisson	32
sim.gradient	33
sim.interval.ages	34
sim.taxonomy	35
sim.tip.samples	36
sim.trait.values	37
species.end	38
species.start	39
subsample.fossils.oldest	39
subsample.fossils.oldest.and.youngest	40
subsample.fossils.uniform	41
subsample.fossils.youngest	41
summary.taxonomy	42
taxonomy	43
tree.max	44
Index	45

beast.fbd.format	<i>Transforms a tree and fossils into a sampled tree in beast-usable format and writes it in Newick format. Designed to work with FBD.</i>
------------------	--

Description

Transforms a tree and fossils into a sampled tree in beast-usable format and writes it in Newick format. Designed to work with FBD.

Usage

```
beast.fbd.format(tree, fossils, rho = 1, sampled_tips = NULL, ...)
```

Arguments

tree	Complete tree.
fossils	fossils dataframe.
rho	Sampling probability of extant tips. Default 1, will be disregarded if sampled_tips is not null.
sampled_tips	List of tip labels corresponding to sampled extant tips.
...	Additional parameters will be passed to ape::write.tree

Value

Output of write.tree.

Examples

```
# simulate tree
t = ape::rtree(6)

# simulate fossils
f = sim.fossils.poisson(rate = 2, tree = t)

# output for BEAST
beast.fbd.format(t, f) # output on the console
beast.fbd.format(t, f, file="example.tre") # output in file
```

<code>count.fossils</code>	<i>Count the total number of fossils</i>
----------------------------	--

Description

Count the total number of fossils

Usage

```
count.fossils(fossils)
```

Arguments

`fossils` Fossils object.

Value

Number of extinct samples.

<code>count.fossils.binned</code>	<i>Count the total number of fossils per interval</i>
-----------------------------------	---

Description

Count the total number of fossils per interval

Usage

```
count.fossils.binned(fossils, interval.ages)
```

Arguments

`fossils` Fossils object.

`interval.ages` Vector of stratigraphic interval ages, starting with the minimum age of the youngest interval and ending with the maximum age of the oldest interval.

Value

Vector of extinct samples corresponding to each interval. Note the last value corresponds to the number of samples > the maximum age of the oldest interval.

`fossils`*Fossils object*

Description

Create a fossil record object. The input is taken to be a dataframe or list.

Usage

```
fossils(data = NULL, from.taxonomy = FALSE)
```

```
as.fossils(data, from.taxonomy = FALSE)
```

```
is.fossils(data)
```

Arguments

`data` Dataframe or list of sampled fossils. See Details for the list of required fields. If NULL, the function creates an empty fossils object.

`from.taxonomy` Boolean indicating whether the fossils were sampled using a taxonomy object, as opposed to a tree object. Default = FALSE.

Details

The fossil record object contains 4 fields for each fossil with the following information:

- `sp` the label of the corresponding species. This label matches the edge labels in the corresponding phylo object or the species labels in the corresponding taxonomy object if additional taxonomic information was provided
- `edge` the label of the sampled node or tip in the phylogeny, i.e the node at the end of the edge along which the fossil was sampled
- `hmin` the age of the fossil or the youngest bound of the time interval in which the fossil was sampled
- `hmax` the oldest bound of the time interval in which the fossil was sampled. This is equal to `hmin` if exact sampling times are known

`fossils.to.BEAST.constraints`

Create a set of BEAST2 constraints to construct a DPPDIV style fixed extant topology FBD analysis

Description

If complete = FALSE, only the extant taxa are used to construct the taxon constraints, resulting in a DPPDIV style analysis in which the extant topology is fixed and fossils can float in the tree. The resulting output uses the stronglyMonophyletic taxon constraint on the root, this means that all fossil taxa will be sampled in the crown group, and never in a position below the root.

Usage

```
fossils.to.BEAST.constraints(fossils, tree,
  file = "BEASTconstraints.xml", complete = FALSE,
  tree.name = "beastTree")
```

Arguments

fossils	an object of class "fossils" that corresponds to fossil occurrences for the "tree" argument.
tree	an object of class "phylo", representing the tree upon which the fossil occurrences were simulated.
file	the name of the file to which the constraints will be written, defaults to "BEASTconstraints.xml".
complete	logical, if TRUE then taxon constraints are built for the complete tree, if FALSE then constraints are built for the crown clades only. Default value is FALSE.
tree.name	the name of the tree as used in the BEAST2 xml format.

Value

NULL.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
f = sim.fossils.poisson(0.1, t, root.edge = FALSE)
fossils.to.BEAST.constraints(f, t, complete = TRUE)
```

```
fossils.to.BEAST.start.tree
```

Create a suitable starting tree for a DPPDIV style FBD analysis in BEAST2

Description

If complete = FALSE, only the extant taxa are used to construct the tree, resulting in a DPPDIV style analysis in which the extant topology is fixed and fossils can float in the tree.

Usage

```
fossils.to.BEAST.start.tree(tree, fossils, complete = FALSE)
```

Arguments

tree	an object of class "phylo", representing the tree upon which the fossil occurrences were simulated.
fossils	an object of class "fossils" that corresponds to fossil occurrences for the "tree" argument.
complete	logical, if TRUE then the tree are built for the complete tree, if FALSE then the tree is built for the crown clades only.

Value

a string representing the starting tree in newick format.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
f = sim.fossils.poisson(0.1, t, root.edge = FALSE)
fossils.to.BEAST.start.tree(t,f, complete = FALSE)
```

fossils.to.paleotree.record

Transforms a fossils dataframe and either taxonomy or tree into a fossilRecordSimulation object from package paleotree.

Description

Transforms a fossils dataframe and either taxonomy or tree into a fossilRecordSimulation object from package paleotree.

Usage

```
fossils.to.paleotree.record(fossils, tree = NULL, taxonomy = NULL)
```

Arguments

fossils	fossils object
tree	phylo object containing the tree. If provided and taxonomy = NULL, all specification is assumed symmetric
taxonomy	taxonomy object. If both tree and taxonomy are provided, only taxonomy will be used.

Value

The converted paleotree record

See Also

[taxonomy](#), [fossils](#), [paleotree.record.to.fossils](#)

Examples

```
# simulate tree
t = ape::rtree(6)
# simulate fossils using taxonomy
s = sim.taxonomy(t, 0.5, 1, 0.5)
f = sim.fossils.poisson(2, taxonomy = s)
# transform format
record = fossils.to.paleotree.record(f, taxonomy = s)
```

fossils.to.pyrate *Generate output in the format used by the program PyRate*

Description

Generate output in the format used by the program PyRate

Usage

```
fossils.to.pyrate(fossils, python = TRUE, traits = NULL,
  cutoff = NULL, random = FALSE, min = NULL,
  exclude.extant.singletons = TRUE, file = "")
```

Arguments

fossils	Fossils object.
python	If TRUE the function outputs the data in the python format used by PyRate (default). If FALSE the function outputs a tab-delimited table used by tools associated with PyRate.
traits	Vector of trait values equal to the number of unique species in the fossils dataframe. The order should correspond to the order in which they appear in <code>unique(fossils\$sp)</code> .
cutoff	Exclude occurrences with age uncertainty greater than this value i.e. $h_{max} - h_{min} > cutoff$.
random	If TRUE use a random number from within the interval $U(h_{min}, h_{max})$ for specimen ages, otherwise use the midpoint of this interval (default). Applicable only when <code>python = TRUE</code> and for specimens with $h_{min} \neq h_{max}$.
min	Value used to represent the minimum possible interval age of extinct specimens with $h_{min} = 0$. By default <code>min = NULL</code> and the function will use the sampling times in the fossils dataframe.
exclude.extant.singletons	If TRUE exclude species that have extant samples only (default = TRUE).
file	Output file name.

Examples

```
set.seed(123)

# simulate tree
t = ape::rtree(6)

# assign a max age based on tree height
max.age = tree.max(t)

# define a set of non-uniform length intervals
times = c(0, sort(runif(3, min = 0, max = max.age)), max.age)
rates = c(1,2,3,4)

# simulate fossils reflect age uncertainty
f = sim.fossils.intervals(tree = t, interval.ages = times, rates = rates,
  use.exact.times = FALSE)

# simulate extant samples
rho = 1
f = sim.extant.samples(f, t, rho = 1)

plot(f, t)

# generate input files for pyrate
fossils.to.pyrate(f)
fossils.to.pyrate(f, python = FALSE)

# add trait values
traits = runif(length(unique(f$sp)))
fossils.to.pyrate(f, traits = traits)
```

FossilSim

FossilSim: A package for simulating and plotting fossil and taxonomy data

Description

This package provides functions for simulating both taxonomy and fossil data from an existing phylogeny.

Simulating taxonomy

Taxonomy can be simulated in FossilSim under a mixed model of speciation that can incorporate three modes of speciation – budding (or asymmetric), bifurcating (or symmetric) and anagenetic – in addition to cryptic speciation. A description of the resulting taxonomy objects and simulation functions can be found in the "Simulating taxonomy" vignette.

Simulating fossil data

Fossils can be simulated from a phylogeny or a taxonomy under a model of constant fossil recovery or time-dependent, environment-dependent or species-dependent fossil recovery. A description of the resulting fossil objects and simulation functions can be found in the "Simulating fossils" vignette.

Plotting functions

Both taxonomy and fossil objects are provided with custom plotting functions that highlight important features of the simulated objects along the original phylogeny. More details about these functions can be found in the vignettes or by calling `?plot.taxonomy` and `?plot.fossils`.

Compatibility with other packages

FossilSim is designed to use phylogenies in the ape format. It provides functions to convert to and from the `fossilRecordSimulation` format used by the package `paleotree` (see the vignette "Converting from and to paleotree format"), as well as functions to convert to the zero-edge format used by BEAST2 and RevBayes (see the vignette "Exporting sampled ancestor trees").

Examples

```
# simulate a tree using TreeSim conditioned on tip number
t = TreeSim::sim.bd.taxa(n = 10, numbsim = 1, lambda = 1, mu = 0.2)[[1]]

# simulate taxonomy under mixed speciation
s = sim.taxonomy(tree = t, beta = 0.5, lambda.a = 1, kappa = 0.1)
# plot the result
plot(s, tree = t, legend.position = "topleft")

# simulate fossils using the phylogeny and a constant fossil recovery rate
f = sim.fossils.poisson(rate = 3, tree = t)
# plot the result
plot(f, tree = t)

# simulate fossils using the taxonomy and a constant fossil recovery rate
f = sim.fossils.poisson(rate = 3, taxonomy = s)
# plot the result
plot(f, tree = t, taxonomy = s, show.taxonomy = TRUE)

# simulate fossils using time-dependent fossil recovery rates
f = sim.fossils.intervals(tree = t, rates = c(1, 0.1, 1, 0.1), max.age = tree.max(t), strata = 4)
# plot the result, with the time intervals
plot(f, tree = t, show.strata = TRUE, max.age = tree.max(t), strata = 4)
```

get.tip.descs	<i>Obtain the tips that define each node in a tree</i>
---------------	--

Description

Obtain the tips that define each node in a tree

Usage

```
get.tip.descs(tree)
```

Arguments

tree an object of class "Phylo".

Value

A list of vectors, with one entry for each node consisting of the tip labels that define that node.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
get.tip.descs(t)
```

paleotree.record.to.fossils

Transforms a fossilRecordSimulation object from package paleotree to a tree and taxonomy and fossils objects.

Description

The returned tree is in paleotree format, with zero-length edges leading to tips at bifurcation and anagenetic events. Fossils and taxonomy are only specified on non-zero-length edges. The label assigned to the parent of the origin or root will be zero.

Usage

```
paleotree.record.to.fossils(record, alphanumeric = TRUE)
```

Arguments

record fossilRecordSimulation object.

alphanumeric If TRUE function will return alphanumeric species labels (i.e. species labels contain the "t" prefix) (default). If FALSE function will return numeric only species labels.

Value

A list containing the converted tree, taxonomy and fossils

See Also

[taxonomy](#), [fossils](#), [fossils.to.paleotree.record](#)

Examples

```
if (requireNamespace("paleotree", quietly = TRUE)) {
# simulate record
record = paleotree::simFossilRecord(p=0.1, q=0.1,r=0.1, nruns=1, nTotalTaxa=c(30,40),
  nExtant=0, nSamp = c(5,25))

# transform format
l_tf = paleotree.record.to.fossils(record)
l_tf$tree
l_tf$taxonomy
l_tf$fossils
}
```

place.fossils

Place fossil samples from one tree in another tree, or find the ancestral node for each fossil sample in one tree.

Description

If "ext.tree" is not supplied, this function will find the direct ancestral node for each of the supplied fossil samples. If "ext.tree" is supplied, this function will find the direct ancestral node for each fossil in "ext.tree". This second behaviour is used for placing fossils simulated on a complete Birth-Death tree in the extant-only counterpart tree. This results in fossil samples being placed in the crown clades of the tree upon which they were simulated. When "ext.tree" is supplied, any fossil samples appearing before the MRCA of the crown group are discarded.

Usage

```
place.fossils(tree, fossils, ext.tree)
```

Arguments

tree	an object of class "Phylo".
fossils	an object of class "fossils" that corresponds to fossil occurrences for the "tree" argument.
ext.tree	an object of class "Phylo" representing the extant counterpart to "tree", this can be obtained with <code>prune.fossil.tips(tree)</code> .

Value

a vector of node numbers corresponding to the direct ancestor of each fossil sample in "fossils".

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
f = sim.fossils.poisson(0.1, t, root.edge = FALSE)
place.fossils(t,f)
```

plot.fossils

Plot simulated fossils

Description

This function is adapted from the *ape* function `plot.phylo` used to plot phylogenetic trees. The function can be used to plot simulated fossils (`show.fossils = TRUE`), with or without the corresponding tree (`show.tree = TRUE`), stratigraphic intervals (`show.strata = TRUE`), stratigraphic ranges (`show.ranges = TRUE`) and sampling proxy data (`show.proxy = TRUE`). Interval ages can be specified as a vector (`interval.ages`) or a uniform set of interval ages can be specified using the number of intervals (`strata`) and maximum interval age (`max`), where interval length = $max.age/strata$. If no maximum age is specified, the function calculates a maximum interval age slightly older than the root edge (or root age if `root.edge = FALSE`), using the function `tree.max()`.

Usage

```
## S3 method for class 'fossils'
plot(x, tree, show.fossils = TRUE, show.tree = TRUE,
     show.ranges = FALSE, show.strata = FALSE, strata = 1,
     max.age = NULL, interval.ages = NULL, binned = FALSE,
     show.axis = TRUE, show.proxy = FALSE, proxy.data = NULL,
     show.preferred.environ = FALSE, preferred.environ = NULL,
     show.taxonomy = FALSE, taxonomy = NULL, root.edge = TRUE,
     hide.edge = FALSE, edge.width = 1, show.tip.label = FALSE,
     align.tip.label = FALSE, fossil.col = 1, range.col = rgb(0, 0, 1),
     extant.col = NULL, cex = 1.2, pch = 18, ...)
```

Arguments

<code>x</code>	Fossils object.
<code>tree</code>	Phylo object.
<code>show.fossils</code>	If TRUE plot fossils (default = TRUE).
<code>show.tree</code>	If TRUE plot the tree (default = TRUE).
<code>show.ranges</code>	If TRUE plot stratigraphic ranges (default = FALSE). If <code>show.taxonomy = FALSE</code> all occurrences along a single edge are grouped together (i.e. function assumes all speciation is symmetric).

<code>show.strata</code>	If TRUE plot strata (default = FALSE).
<code>strata</code>	Number of stratigraphic intervals (default = 1).
<code>max.age</code>	Maximum age of a set of equal length intervals. If no value is specified (max = NULL), the function uses a maximum age based on tree height.
<code>interval.ages</code>	Vector of stratigraphic interval ages, starting with the minimum age of the youngest interval and ending with the maximum age of the oldest interval.
<code>binned</code>	If TRUE fossils are plotted at the mid point of each interval.
<code>show.axis</code>	If TRUE plot x-axis (default = TRUE).
<code>show.proxy</code>	If TRUE add profile of sampling data to plot (e.g rates in time-dependent rates model) (default = FALSE).
<code>proxy.data</code>	Vector of sampling proxy data (default = NULL). Should be as long as the number of stratigraphic intervals.
<code>show.preferred.envirom</code>	If TRUE add species preferred environmental value (e.g. water depth) (default = FALSE). Only works if combined with <code>show.proxy = TRUE</code> .
<code>preferred.envirom</code>	Preferred environmental value (e.g. water depth). Currently only one value can be shown.
<code>show.taxonomy</code>	If TRUE highlight species taxonomy.
<code>taxonomy</code>	Taxonomy object.
<code>root.edge</code>	If TRUE include the root edge (default = TRUE).
<code>hide.edge</code>	If TRUE hide the root edge but still incorporate it into the automatic timescale (default = FALSE).
<code>edge.width</code>	A numeric vector giving the width of the branches of the plotted phylogeny. These are taken to be in the same order as the component edge of tree. If fewer widths are given than the number of edges, then the values are recycled.
<code>show.tip.label</code>	Whether to show the tip labels on the phylogeny (defaults to FALSE).
<code>align.tip.label</code>	A logical value or an integer. If TRUE, the tips are aligned and dotted lines are drawn between the tips of the tree and the labels. If an integer, the tips are aligned and this gives the type of the lines (following <code>lty</code>).
<code>fossil.col</code>	Colour of fossil occurrences.
<code>range.col</code>	Colour of stratigraphic ranges.
<code>extant.col</code>	Colour of extant samples. If <code>show.taxonomy = TRUE</code> <code>extant.col</code> will be ignored.
<code>cex</code>	Numeric value giving the factor used to scale the points representing the fossils when <code>show.fossils = TRUE</code> .
<code>pch</code>	Numeric value giving the symbol used for the points representing the fossils when <code>show.fossils = TRUE</code> .
<code>...</code>	Additional parameters to be passed to <code>plot.default</code> .

Examples

```

set.seed(123)

## simulate tree
t = TreeSim::sim.bd.taxa(8, 1, 1, 0.3)[[1]]

## simulate fossils under a Poisson sampling process
f = sim.fossils.poisson(rate = 3, tree = t)
plot(f, t)
# add a set of equal length strata
plot(f, t, show.strata = TRUE, strata = 4)
# show stratigraphic ranges
plot(f, t, show.strata = TRUE, strata = 4, show.ranges = TRUE)

## simulate fossils and highlight taxonomy
s = sim.taxonomy(t, 0.5, 1)
f = sim.fossils.poisson(rate = 3, taxonomy = s)
plot(f, t, taxonomy = s, show.taxonomy = TRUE, show.ranges = TRUE)

## simulate fossils under a non-uniform model of preservation
# assign a max interval based on tree height
max.age = tree.max(t)
times = c(0, 0.3, 1, max.age)
rates = c(4, 1, 0.1)
f = sim.fossils.intervals(t, interval.ages = times, rates = rates)
plot(f, t, show.strata = TRUE, interval.ages = times)
# add proxy data
plot(f, t, show.strata = TRUE, interval.ages = times, show.proxy = TRUE, proxy.data = rates)

```

plot.taxonomy

Plot simulated taxonomy

Description

This function is adapted from the *ape* function `plot.phylo` used to plot phylogenetic trees. The function can be used to plot simulated taxonomy along with the corresponding tree.

Usage

```

## S3 method for class 'taxonomy'
plot(x, tree, show.mode = TRUE, show.legend = TRUE,
     legend.position = "bottomleft", root.edge = TRUE,
     hide.edge = FALSE, edge.width = 1, show.tip.label = FALSE,
     align.tip.label = FALSE, cex = 1.2, ...)

```

Arguments

x	Taxonomy object.
tree	Phylo object.
show.mode	Indicate speciation mode.
show.legend	Add a legend for the symbols indicating different speciation modes.
legend.position	Position of the legend. Options include "bottomleft" (default), "topleft", "bottomright", "topright" or a vector of length 2 with x, y coordinates.
root.edge	If TRUE include the root edge (default = TRUE).
hide.edge	If TRUE hide the root edge but still incorporate it into the automatic timescale (default = FALSE).
edge.width	A numeric vector giving the width of the branches of the plotted phylogeny. These are taken to be in the same order as the component edge of tree. If fewer widths are given than the number of edges, then the values are recycled.
show.tip.label	Whether to show the tip labels on the phylogeny (defaults to FALSE).
align.tip.label	A logical value or an integer. If TRUE, the tips are aligned and dotted lines are drawn between the tips of the tree and the labels. If an integer, the tips are aligned and this gives the type of the lines (lty).
cex	Numeric value giving the factor used to scale the points representing the fossils when show.fossils = TRUE.
...	Additional parameters to be passed to plot.default.

Examples

```
set.seed(123)

## simulate tree
t = TreeSim::sim.bd.taxa(8, 1, 1, 0.3)[[1]]

## simulate taxonomy
s = sim.taxonomy(t, 0.5, 1)

## plot the output
plot(s, t)
```

```
prune.fossil.tips      Remove fossil lineages from a tree
```

Description

Remove fossil lineages from a tree

Usage

```
prune.fossil.tips(tree)
```

Arguments

tree an object of class "Phylo".

Value

an object of class "Phylo". If fossil lineages were found in the tree these will be pruned, if not then the original tree is returned.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
prune.fossil.tips(t)
```

prune.fossils	<i>Removes all intermediate fossils from a combined tree and labels the first and last fossils of each lineage. Can be used with sampled or complete trees. If only one fossil is present for a particular species it is labelled as first.</i>
---------------	---

Description

Removes all intermediate fossils from a combined tree and labels the first and last fossils of each lineage. Can be used with sampled or complete trees. If only one fossil is present for a particular species it is labelled as first.

Usage

```
prune.fossils(tree)
```

Arguments

tree Combined tree with fossils.

Value

Tree with pruned fossils.

Examples

```
# simulate tree
t = ape::rtree(6)

# simulate fossils
f = sim.fossils.poisson(rate = 2, tree = t)

# transform format
t2 = SAtree.from.fossils(t,f)

# prune fossils
t4 = prune.fossils(t2)

# or transform to sampled tree first
t3 = sampled.tree.from.combined(t2)
t4 = prune.fossils(t3)
plot(t4)
```

rangeplot.asymmetric *Make an asymmetric stratigraphic range plot from a tree object of class phylo*

Description

Make an asymmetric stratigraphic range plot from a tree object of class phylo

Usage

```
rangeplot.asymmetric(x, complete = FALSE, ...)
```

Arguments

x	phylo object to plot.
complete	Plot unsampled species.
...	Additional parameters to be passed to plot.default.

Note

This function assumes all speciation events are asymmetric.

Examples

```
tree = sim.fbd.taxa(n = 10, numbsim = 1, lambda = 3, mu = 2, psi = 1, complete = TRUE)[[1]]
rangeplot.asymmetric(tree, complete=TRUE)
```

`reconcile.fossils.taxonomy`*Reconcile existing fossil and taxonomy objects*

Description

This function uses edge identifiers (`edge`) and fossil sampling times (`hmin`) to reassign fossil species identifiers (`sp`, `origin`) using an existing taxonomy object. It can only be used if exact fossil sampling times are known (i.e. `hmin = hmax`), otherwise edges containing multiple species may be indistinguishable.

Usage

```
reconcile.fossils.taxonomy(fossils, taxonomy)
```

Arguments

<code>fossils</code>	Fossils object.
<code>taxonomy</code>	Taxonomy object.

Value

An object of class `fossils`.

Examples

```
# simulate tree
t = ape::rtree(6)

# simulate fossils using the tree
rate = 2
f = sim.fossils.poisson(rate, tree = t)
plot(f, t)

# simulate fossils using taxonomy
s = sim.taxonomy(t, 0.5, 1, 0.5)
f = reconcile.fossils.taxonomy(f, s)
plot(f, t)
```

remove.stem.fossils *Remove fossil samples that occur in the stem*

Description

Remove fossil samples that occur in the stem

Usage

```
remove.stem.fossils(fossils, tree)
```

Arguments

fossils an object of class "fossils" that corresponds to fossil occurrences for "tree".
tree an object of class "Phylo".

Value

an object of class "fossils", containing only the fossil samples that occur in the crown.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]  
f = sim.fossils.poisson(0.1, t, root.edge = FALSE)  
remove.stem.fossils(f, t)
```

remove.stem.lineages *Remove stem lineages from a tree.*

Description

Remove stem lineages from a tree.

Usage

```
remove.stem.lineages(tree)
```

Arguments

tree an object of class "Phylo".

Value

an object of class "Phylo", if stem lineages were found in the tree these will be pruned; if not then the original tree is returned.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
remove.stem.lineages(t)
```

```
sampled.tree.from.combined
```

Removes all unsampled lineages from a combined tree. Extinct tips are only sampled if they are fossils. With default settings all extant tips are sampled.

Description

Removes all unsampled lineages from a combined tree. Extinct tips are only sampled if they are fossils. With default settings all extant tips are sampled.

Usage

```
sampled.tree.from.combined(tree, rho = 1, sampled_tips = NULL)
```

Arguments

tree	Combined tree with fossils.
rho	Sampling probability of extant tips. Default 1, will be disregarded if sampled_tips is not null.
sampled_tips	List of tip labels corresponding to sampled extant tips.

Value

Sampled tree with fossils.

Examples

```
# simulate tree
t = ape::rtree(6)

# simulate fossils
f = sim.fossils.poisson(rate = 2, tree = t)

# transform format
t2 = SATree.from.fossils(t,f)

# transform to sampled tree
t3 = sampled.tree.from.combined(t2)
plot(t3)
```

SAtree	<i>Tree with sampled ancestors represented as zero-length edges</i>
--------	---

Description

Converts a phylo object to SAtree, without modification of tip labels.

Usage

```
SAtree(tree, complete = TRUE)
```

Arguments

tree	Phylo object.
complete	Whether the tree is complete. Default TRUE. If the tree is not complete, then all fossil tips correspond to fossil samples, otherwise only sampled ancestors are considered samples.

SAtree.from.fossils	<i>Transforms a tree and fossils dataframe to a combined SA tree. Sampled ancestors are represented as tips on zero-length edges to maintain compatibility with the ape format. Tip labels are set to "species id"_"index", where the most recent tip of a given species receives index 1 and indices increase towards the past.</i>
---------------------	--

Description

Transforms a tree and fossils dataframe to a combined SA tree. Sampled ancestors are represented as tips on zero-length edges to maintain compatibility with the ape format. Tip labels are set to "species id"_"index", where the most recent tip of a given species receives index 1 and indices increase towards the past.

Usage

```
SAtree.from.fossils(tree, fossils)
```

Arguments

tree	Phylo object.
fossils	Fossils object.

Value

A tree integrating the fossils.

Examples

```
# simulate tree
t = ape::rtree(6)

# simulate fossils
f = sim.fossils.poisson(rate = 2, tree = t)

# transform format
t2 = SATree.from.fossils(t,f)
plot(t2)
```

sim.anagenetic.species

Simulate anagenetic species on a taxonomy object

Description

Simulate anagenetic species on a taxonomy object

Usage

```
sim.anagenetic.species(tree, species, lambda.a)
```

Arguments

tree	Phylo object.
species	Taxonomy object.
lambda.a	Rate of anagenetic speciation. Default = 0.

Value

Object of class taxonomy.

See Also

[taxonomy](#)

Examples

```
t = ape::rtree(10)
sp = sim.taxonomy(t, 1)
sim.anagenetic.species(t, sp, 0.1)
```

sim.cryptic.species *Simulate cryptic species on a taxonomy object*

Description

Simulate cryptic species on a taxonomy object

Usage

```
sim.cryptic.species(species, kappa)
```

Arguments

species	Taxonomy object.
kappa	Probability that speciation event is cryptic.

Value

An object of class taxonomy. Note the origin or root can not be cryptic.

See Also

[taxonomy](#)

Examples

```
t = ape::rtree(10)
sp = sim.taxonomy(t, 1)
sim.cryptic.species(sp, 0.5)
```

sim.extant.samples *Include extant samples in the fossil object, with optional rho sampling.*

Description

Include extant samples in the fossil object, with optional rho sampling.

Usage

```
sim.extant.samples(fossils, tree = NULL, taxonomy = NULL, rho = 1,
  tol = NULL)
```


Arguments

fossils	Fossils object.
tree	Phylo object.
taxonomy	Taxonomy object.
rho	Extant species sampling probability.
tol	Rounding error tolerance for tip ages.

Value

An object of class fossils containing extant tip samples equal to the age of the tips (i.e. 0.0).

Examples

```
# simulate tree
lambda = 0.1
mu = 0.05
tips = 8
t = TreeSim::sim.bd.taxa(tips, 1, lambda, mu)[[1]]

# simulate fossils
f = sim.fossils.poisson(0.5, t)

# simulate extant samples
f = sim.extant.samples(f, t, rho = 0.5)
plot(f, t)
```

sim.fbd.age

sim.fbd.age: Simulating fossilized birth-death trees of a fixed age.

Description

sim.fbd.age: Simulating fossilized birth-death trees of a fixed age.

Usage

```
sim.fbd.age(age, numbsim, lambda, mu, psi, frac = 1, mrca = FALSE,
  complete = FALSE, K = 0)
```

Arguments

age	Time since origin / most recent common ancestor.
numbsim	Number of trees to simulate.
lambda	Speciation rate.
mu	Extinction rate.
psi	Fossil sampling rate.

frac	Sampling fraction: The actual number of tips is n/frac , but only n tips are included (incomplete sampling).
mrca	If mrca=FALSE: age is the time since origin. If mrca=TRUE: age is the time since most recent common ancestor of the extant tips.
complete	If complete = TRUE, the tree with the extinct and non-sampled lineages is returned. If complete = FALSE, the extinct and non-sampled lineages are suppressed.
K	If $K=0$, then lambda is constant. If $K>0$, density-dependent speciation is assumed, with speciation rate = $\lambda(1-m/K)$ when there are m extant species.

Value

Array of 'numbsim' SATrees with the time since origin / most recent common ancestor being 'age'. If tree goes extinct or no tips are sampled (only possible when mrca = FALSE), return value is '0'. If only one extant and no extinct tips are sampled, return value is '1'.

Examples

```
age = 1
lambda = 2.0
mu = 0.5
psi = 0.6
numbsim = 2
sim.fbd.age(age, numbsim, lambda, mu, psi)
```

sim.fbd.rateshift.taxa

sim.fbd.rateshift.taxa: Simulating fossilized birth death trees incorporating rate shifts.

Description

sim.fbd.rateshift.taxa: Simulating fossilized birth death trees incorporating rate shifts.

Usage

```
sim.fbd.rateshift.taxa(n, numbsim, lambda, mu, psi, times,
  complete = FALSE)
```

Arguments

n	Number of extant sampled tips.
numbsim	Number of trees to simulate.
lambda	Vector of speciation rates, the rate in entry i is the speciation rate prior (ancestral) to time times[i].

mu	Vector of extinction rates, the rate in entry i is the extinction rate prior (ancestral) to time times[i].
psi	Vector of fossil sampling rates, the rate in entry i is the fossil sampling rate prior (ancestral) to time times[i].
times	Vector of mass extinction and rate shift times. Time is 0 today and increasing going backwards in time. Specify the vector as times[i]<times[i+1]. times[1]=0 (today).
complete	If TRUE, the tree including the extinct lineages and non-sampled lineages is returned. If FALSE, the extinct lineages and non-sampled lineages are suppressed.

Value

List of numbsim simulated SAtrees with n extant sampled tips.

Examples

```
n = 10
numbsim = 1
sim.fbd.rateshift.taxa(n, numbsim, lambda = c(2,1), mu = c(0,0.3), psi = c(1,0.1), times = c(0,0.3))
```

sim.fbd.taxa	<i>sim.fbd.taxa: Simulating fossilized birth-death trees on a fixed number of extant taxa.</i>
--------------	--

Description

sim.fbd.taxa: Simulating fossilized birth-death trees on a fixed number of extant taxa.

Usage

```
sim.fbd.taxa(n, numbsim, lambda, mu, psi, frac = 1, complete = FALSE)
```

Arguments

n	Number of extant sampled tips.
numbsim	Number of trees to simulate.
lambda	Speciation rate.
mu	Extinction rate.
psi	Fossil sampling rate.
frac	When complete = FALSE: The actual number of extant tips is n/frac, but only n tips are included (incomplete sampling). When complete = TRUE: all extinct and non-sampled lineages are included, i.e. the tree has n/frac extant tips.
complete	If TRUE, the tree with the extinct and non-sampled lineages is returned. If FALSE, the extinct lineages are suppressed.

Value

List of numbsim simulated SATrees with n extant sampled tips.

Examples

```
n = 10
lambda = 2.0
mu = 0.5
psi = 0.6
numbsim = 2
sim.fbd.taxa(n, numbsim, lambda, mu, psi)
```

sim.fossils.environment

Simulate fossils under an environment-dependent model of preservation (Holland, 1995)

Description

This function uses a three parameter Gaussian model to simulate non-uniform fossil recovery along a specified phylogeny. Preservation varies with respect to water depth, which is a useful proxy for changes in the depositional environment. The per interval probability of sampling is

$$P(\text{collection}) = PAe^{-(d - PD)^2 / 2 * DT^2}$$

where PA is species peak abundance, PD is preferred depth, DT is depth tolerance and d is current water depth. PD is the depth at which the species is most likely to be found and is equivalent to the mean of the distribution. PA is the probability of sampling an occurrence at this depth. DT is the potential of a species to be found at a range of depths and is equivalent to the standard deviation. Although here fossil recovery is described with respect to water depth, the model could be applied in the context of any environmental gradient.

The model uses a probability of collecting a fossil within a given interval, rather than a rate.

To simulate discrete fossil sampling events and times within each interval we need to convert the probability into a rate (use .rates = TRUE). This is done using the formula

$$\text{rate} = -\ln(1 - P(\text{collection})/t)$$

where t is the interval length. One caveat of this approach is that the model cannot use a probability of 1, as it would correspond to rate = infinity. In this instance we use an approximation for probabilities = 1 (e.g. pr.1.approx = 0.999).

Non-uniform interval ages can be specified as a vector (interval.ages) or a uniform set of interval ages can be specified using maximum interval age (max.age) and the number of intervals (strata), where interval length = max.age/strata.

A vector of values can be specified for the model parameters *PA*, *PD* and *DT* to allow for variation across lineages. If a vector is provided, each entry will apply to each unique species in the order in which they appear in the taxonomy object (if taxonomy is provided), or to each unique edge in the order in which they appear in the tree object. If the tree object has a root edge (`root.edge`), the first entry in the vector will apply to this edge.

Fossils can be simulated for a phylo (tree) or taxonomy (taxonomy) object. If both are specified, the function uses taxonomy. If no taxonomic information is provided, the function assumes all speciation is symmetric (i.e. bifurcating, $\beta = 1$).

Usage

```
sim.fossils.environment(tree = NULL, taxonomy = NULL,
  interval.ages = NULL, max.age = NULL, strata = NULL,
  proxy.data = NULL, PD = 0.5, DT = 0.5, PA = 0.5,
  root.edge = TRUE, use.rates = FALSE, pr.1.approx = 0.999)
```

Arguments

<code>tree</code>	Phylo object.
<code>taxonomy</code>	Taxonomy object.
<code>interval.ages</code>	Vector of stratigraphic interval ages, starting with the minimum age of the youngest interval and ending with the maximum age of the oldest interval.
<code>max.age</code>	Maximum age of the oldest stratigraphic interval or age at the base of the basin.
<code>strata</code>	Number of stratigraphic intervals.
<code>proxy.data</code>	Vector of relative water depth or other proxy data. The first number corresponds to the youngest interval. The length of the vector should be 1 less than the length of <code>interval.ages</code> .
<code>PD</code>	Preferred depth parameter value or a vector of values.
<code>DT</code>	Depth tolerance parameter value or a vector of values.
<code>PA</code>	Peak abundance parameter value or a vector of values.
<code>root.edge</code>	If TRUE include the root edge. Default = TRUE.
<code>use.rates</code>	If TRUE convert per interval sampling probability into a per interval Poisson rate. Default = FALSE.
<code>pr.1.approx</code>	Value used to approximate sampling probabilities = 1 when <code>use.rates = TRUE</code> .

Value

An object of class `fossils`, where `hmin` and `hmax` will equal the start and end times of the corresponding interval.

References

Holland, S.M. 1995. The stratigraphic distribution of fossils. *Paleobiology* 21: 92-109.

See Also

[sim.fossils.poisson](#), [sim.fossils.intervals](#), [sim.trait.values](#)

Examples

```
# simulate tree
t = ape::rtree(6)

# assign a max age based on tree height
max.age = tree.max(t)

# generate water depth profile
strata = 7
wd = sim.gradient(strata)

# simulate fossils using tree & max.age and strata
f = sim.fossils.environment(t, max.age = max.age, strata = strata,
  proxy.data = wd, PD = 0.5, DT = 1, PA = 1)
plot(f, t, show.proxy = TRUE, proxy.data = wd, strata = strata, show.strata = TRUE)

# simulate fossils using taxonomy & interval.ages
s = sim.taxonomy(t, 0.1, 0.1, 1)
times = seq(0, max.age, length.out = strata + 1)
f = sim.fossils.environment(taxonomy = s, interval.ages = times,
  proxy.data = wd, PD = 0.5, DT = 1, PA = 1)
plot(f, t, strata = strata, binned = TRUE)

# simulate fossils with variable preservation across lineages
dist = function() {runif(1)}
PD = sim.trait.values(1, taxonomy = s, model = "independent", dist = dist,
  change.pr = 0.1)
f = sim.fossils.environment(taxonomy = s, interval.ages = times,
  proxy.data = wd, PD = PD, DT = 1, PA = 1)
plot(f, t, strata = strata, binned = TRUE)
```

sim.fossils.intervals *Simulate fossils under a non-uniform model of preservation for a given set of consecutive time intervals*

Description

Intervals can be specified by specifying the interval boundaries using `interval.ages` or specifying both `max.age` and `strata`. In the second scenario all intervals will be of equal length. Preservation can be specified using rates, which represent the rates of a Poisson process in each interval, or probabilities, which represent the probabilities of sampling per interval. When using probabilities, at most one fossil per species will be sampled per interval.

Fossils can be simulated for a phylo (tree) or taxonomy (taxonomy) object. If both are specified, the function uses taxonomy. If no taxonomic information is provided, the function assumes all speciation is symmetric (i.e. bifurcating, $\beta = 1$).

Usage

```
sim.fossils.intervals(tree = NULL, taxonomy = NULL,  
  interval.ages = NULL, max.age = NULL, strata = NULL,  
  probabilities = NULL, rates = NULL, root.edge = TRUE,  
  use.exact.times = TRUE)
```

Arguments

tree	Phylo object.
taxonomy	Taxonomy object.
interval.ages	Vector of stratigraphic interval ages, starting with the minimum age of the youngest interval and ending with the maximum age of the oldest interval.
max.age	Maximum age of the oldest stratigraphic interval.
strata	Number of stratigraphic intervals.
probabilities	Probability of sampling/preservation in each interval. The number of probabilities should match the number of intervals and the first entry should correspond to youngest interval.
rates	Poisson sampling rate for each interval. The number of rates should match the number of intervals and the first entry should correspond to youngest interval.
root.edge	If TRUE include the root edge. Default = TRUE.
use.exact.times	If TRUE use exact sampling times. If FALSE hmin and hmax will equal the start and end times of the corresponding interval. Default = TRUE.

Value

An object of class fossils.

See Also

[sim.fossils.poisson](#), [sim.fossils.environment](#)

Examples

```
# simulate tree  
t = ape::rtree(6)  
  
# assign a max age based on tree height  
max.age = tree.max(t)  
  
# simulate fossils using max.age and strata & probabilities  
strata = 4  
probability = rep(0.7, 4)
```

```
f = sim.fossils.intervals(t, max.age = max.age, strata = strata, probabilities = probability)
plot(f, t, strata = strata, show.strata = TRUE)

# simulate fossils using interval.ages & rates
times = c(0, sort(runif(3, min = 0, max = max.age)), max.age)
rates = c(5, 0, 5, 0)
f = sim.fossils.intervals(t, interval.ages = times, rates = rates)
plot(f, t, interval.ages = times, show.strata = TRUE)

# simulate fossils using taxonomy
s = sim.taxonomy(t, 0.1, 0.1, 1)
f = sim.fossils.intervals(taxonomy = s, interval.ages = times, rates = rates)
plot(f, t, interval.ages = times, show.strata = TRUE)
```

sim.fossils.poisson *Simulate fossils under a Poisson sampling model*

Description

Fossils can be simulated for a phylo (tree) or taxonomy (taxonomy) object. If both are specified, the function uses taxonomy. If no taxonomic information is provided, the function assumes all speciation is symmetric (i.e. bifurcating, $\beta = 1$). A vector of rates can be specified to allow for rate variation across lineages. If a vector is provided, each entry will apply to each unique species in the order in which they appear in the taxonomy object (if taxonomy is provided), or to each unique edge in the order in which they appear in the tree object. If the tree object has a root edge (root.edge), the first entry in the rates vector should correspond to this edge.

Usage

```
sim.fossils.poisson(rate, tree = NULL, taxonomy = NULL,
  root.edge = TRUE)
```

Arguments

rate	A single Poisson sampling rate or a vector of rates.
tree	Phylo object.
taxonomy	Taxonomy object.
root.edge	If TRUE include the root edge. Default = TRUE.

Value

An object of class fossils.

See Also

[sim.fossils.intervals](#), [sim.fossils.environment](#), [sim.trait.values](#)

Examples

```
# simulate tree
t = ape::rtree(6)

# simulate fossils using the tree
rate = 2
f = sim.fossils.poisson(rate, tree = t)
plot(f, t)

# simulate fossils using taxonomy
s = sim.taxonomy(t, 0.5, 1, 0.5)
f = sim.fossils.poisson(rate, taxonomy = s)
plot(f, t)

# simulate fossils with autocorrelated rate variation across lineages
rates = sim.trait.values(init = rate, taxonomy = s, v = 1)
f = sim.fossils.poisson(rates, taxonomy = s)
plot(f, t)
```

sim.gradient

Simulate an environmental gradient

Description

Function returns a vector using the sine wave function $y = depth * \sin(cycles * pi * (x - 1/4))$ for a given set of intervals. This vector can be used as a gradient to simulate fossils under an environment-dependent model of fossil recovery using the function `sim.fossils.environment`.

Usage

```
sim.gradient(strata, depth = 2, cycles = 2)
```

Arguments

strata	Number of stratigraphic intervals.
depth	Maximum water depth.
cycles	Number of cycles (e.g. transgressions and regressions).

Value

vector of sampled water depths.

See Also

[sim.fossils.environment](#)

Examples

```
strata = 100
wd = sim.gradient(strata)
plot(wd, type="l")
```

sim.interval.ages *Reassign fossil ages to user-specified stratigraphic intervals*

Description

Reassign exact fossil ages using the minimum and maximum ages of a set of stratigraphic intervals. If use.species.ages = TRUE the function will respect species durations and will not return minimum and maximum bounds that may be younger or older than the species durations. This requires supplying a phylo or taxonomy object.

Usage

```
sim.interval.ages(fossils, tree = NULL, taxonomy = NULL,
  interval.ages = NULL, max.age = NULL, strata = NULL,
  use.species.ages = FALSE, root.edge = TRUE, sim.extant = FALSE)
```

Arguments

fossils	Fossil object.
tree	Phylo object.
taxonomy	Taxonomy object.
interval.ages	Vector of stratigraphic interval ages, starting with the minimum age of the youngest interval and ending with the maximum age of the oldest interval.
max.age	Maximum age of the oldest stratigraphic interval.
strata	Number of stratigraphic intervals.
use.species.ages	If TRUE reassigned fossil ages will respect the speciation times. Default = FALSE.
root.edge	If TRUE include root edge.
sim.extant	If TRUE simulate age uncertainty for extant samples as well, default FALSE.

Value

An object of class fossils.

Examples

```
# simulate tree
t = ape::rtree(8)

# simulate fossils
rate = 2
f = sim.fossils.poisson(rate, t)
plot(f, t)

# assign a max age based on tree height
max.age = tree.max(t)

# define intervals
times = seq(0, max.age, length.out = 5)

# reassign ages
f = sim.interval.ages(f, t, interval.ages = times)

# plot output
plot(f, t, interval.ages = times)
```

sim.taxonomy

Simulate taxonomy

Description

Simulate a taxonomy object relating species identity to a phylo object under a mixed model of speciation. Anagenetic and cryptic species can also be added later using the `sim.anagenetic.species` and `sim.cryptic.species` functions.

Usage

```
sim.taxonomy(tree, beta = 0, lambda.a = 0, kappa = 0,
             root.edge = TRUE)
```

Arguments

tree	Phylo object.
beta	Probability of bifurcating speciation. Default = 0.
lambda.a	Rate of anagenetic speciation. Default = 0.
kappa	Probability that speciation event is cryptic. Default = 0.
root.edge	If TRUE include root edge. Default = TRUE.

Value

An object of class taxonomy.

See Also[taxonomy](#)**Examples**

```
t = ape::rtree(10)
sim.taxonomy(t, 0.5, 0.1, 0.5)
```

sim.tip.samples	<i>Include extant and extinct tip samples in the fossil object, with optional rho sampling.</i>
-----------------	---

Description

Include extant and extinct tip samples in the fossil object, with optional rho sampling.

Usage

```
sim.tip.samples(fossils, tree, taxonomy = NULL, rho = 1)
```

Arguments

fossils	Fossils object.
tree	Phylo object.
taxonomy	Taxonomy object.
rho	Tip sampling probability.

Value

An object of class fossils containing extant or extinct tip samples equal to the age of the tips.

Examples

```
# simulate tree
t = ape::rtree(6)

# simulate fossils
f = sim.fossils.poisson(2, t)

# simulate tip samples
f = sim.tip.samples(f, t, rho = 0.5)
plot(f, t)
```

sim.trait.values *Simulate trait values with variation across lineages*

Description

Fossil recovery rates or other parameter values can be simulated for a phylo (tree) or taxonomy (taxonomy) object, under an "autocorrelated" or "independent" trait values model. Under the autocorrelated model, trait values evolve along lineages according to a Brownian motion process, where the strength of the relationship between ancestor and descendant values is determined by the parameter ν (v). If ν is small values will be more similar between ancestor and descendants, and if ν is zero all trait values will be equal. For a given species i with ancestor j , a new trait value κ_i is drawn from a lognormal distribution with

$$\kappa_i \sim LN(\ln([\kappa_j] - (\sigma^2/2)), \sigma)$$

where $\sigma = \nu * t_i$ and t_i is the lineage duration of the species. This fossil recovery model is described in Heath et al. (2014) and is equivalent to the autocorrelated relaxed clock model described in Kishino et al. (2001). Under the independent model a new trait value is drawn for each species from any valid user-specified distribution (`dist`). `change.pr` is the probability that a trait value will change at each speciation event. If `change.pr = 1` trait values will be updated at every speciation events.

Usage

```
sim.trait.values(init = 1, tree = NULL, taxonomy = NULL,
  root.edge = TRUE, model = "autocorrelated", v = 0.01,
  dist = function() { runif(1, 0, 2) }, change.pr = 1)
```

Arguments

<code>init</code>	Initial value at the origin or root of the phylo or taxonomy object. Default = 1.
<code>tree</code>	Phylo object.
<code>taxonomy</code>	Taxonomy object.
<code>root.edge</code>	If TRUE include the root edge. Default = TRUE.
<code>model</code>	Model used to simulate rate variation across lineages. Options include "autocorrelated" (default) or "independent".
<code>v</code>	Brownian motion parameter v used in the autocorrelated model. Default = 0.01.
<code>dist</code>	Distribution of trait values used to draw new values under the "independent" model. This parameter is ignored if <code>model = "autocorrelated"</code> . The default is a uniform distribution with $U(0, 2)$. The distribution function must return a single value.
<code>change.pr</code>	Probability that trait values change at speciation events. Default = 1.

Value

A vector of parameter values. Values are output for each species in the order in which they appear in the taxonomy object (if taxonomy was provided) or for each edge in the order in which they appear in the tree object. If the tree object has a root edge (`root.edge`), the first entry in the vector will correspond to this edge.

References

Heath et al. 2014. The fossilized birth-death process for coherent calibration of divergence-time estimates. PNAS 111:E2957-E2966.
 Kishino et al. 2001. Performance of a divergence time estimation method under a probabilistic model of rate evolution MBE 18:352-361.

Examples

```
# simulate tree
t = ape::rtree(6)

# simulate taxonomy
s = sim.taxonomy(t, 0.5, 1, 0.5)

# simulate rates under the autocorrelated trait values model
rate = 2
rates = sim.trait.values(rate, taxonomy = s, v = 1)
f = sim.fossils.poisson(rates, taxonomy = s)
plot(f, t)

# simulate rates under the independent trait values model
dist = function() { rlnorm(1, log(rate), 1) }
rates = sim.trait.values(rate, taxonomy = s, model = "independent", dist = dist)
f = sim.fossils.poisson(rates, taxonomy = s)
plot(f, t)

# simulate rates under the independent trait values model with infrequent changes
rates = sim.trait.values(rate, taxonomy = s, model = "independent",
  dist = dist, change.pr = 0.1)
f = sim.fossils.poisson(rates, taxonomy = s)
plot(f, t)
```

species.end

Find a species' end (i.e extinction) time from a taxonomy object

Description

Find a species' end (i.e extinction) time from a taxonomy object

Usage

```
species.end(species, taxonomy)
```

Arguments

species Species id (as written in taxonomy\$sp).
taxonomy Taxonomy object.

Value

End time.

species.start *Find a species' start (i.e speciation) time from a taxonomy object*

Description

Find a species' start (i.e speciation) time from a taxonomy object

Usage

species.start(species, taxonomy)

Arguments

species Species id (as written in taxonomy\$sp).
taxonomy Taxonomy object.

Value

Start time.

subsample.fossils.oldest
Obtain a subsample of fossil occurrences containing the oldest fossil sample in each node of the tree.

Description

Obtain a subsample of fossil occurrences containing the oldest fossil sample in each node of the tree.

Usage

subsample.fossils.oldest(fossils, tree, complete = TRUE)

Arguments

fossils	an object of class "fossils" that corresponds to fossil occurrences.
tree	an object of class "Phylo", representing the tree upon which the fossil occurrences were simulated.
complete	logical, if TRUE the oldest sample from each clade in the complete tree is returned, if FALSE the oldest sample from each clade in the extant only counterpart tree is returned.

Value

an object of class "fossils" containing the subsampled fossil occurrences.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
f = sim.fossils.poisson(0.1, t, root.edge = FALSE)
subsample.fossils.oldest(f, t, complete = FALSE)
```

subsample.fossils.oldest.and.youngest

Obtain a subsample of fossil occurrences containing the oldest and youngest fossil sample found at each node of the tree.

Description

Obtain a subsample of fossil occurrences containing the oldest and youngest fossil sample found at each node of the tree.

Usage

```
subsample.fossils.oldest.and.youngest(fossils, tree, complete = TRUE)
```

Arguments

fossils	an object of class "fossils" that corresponds to fossil occurrences.
tree	an object of class "Phylo", representing the tree upon which the fossil occurrences were simulated.
complete	logical, if TRUE the oldest and youngest sample from each clade in the complete tree is returned, if FALSE the oldest and youngest sample from each clade in the extant only counterpart tree is returned.

Value

an object of class "fossils" containing the subsampled fossil occurrences.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
f = sim.fossils.poisson(0.1, t, root.edge = FALSE)
subsample.fossils.oldest.and.youngest(f, t, complete = FALSE)
```

```
subsample.fossils.uniform
```

Obtain a uniform random sample of fossil occurrences.

Description

Obtain a uniform random sample of fossil occurrences.

Usage

```
subsample.fossils.uniform(fossils, proportion)
```

Arguments

`fossils` an object of class "fossils" that corresponds to fossil occurrences.
`proportion` the proportion of all fossil samples to return in the subsample.

Value

an object of class "fossils" containing the subsampled fossil occurrences.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
f = sim.fossils.poisson(0.1, t, root.edge = FALSE)
subsample.fossils.uniform(f, 0.5)
```

```
subsample.fossils.youngest
```

Obtain a subsample of fossil occurrences containing the youngest fossil sample in each node of the tree.

Description

Obtain a subsample of fossil occurrences containing the youngest fossil sample in each node of the tree.

Usage

```
subsample.fossils.youngest(fossils, tree, complete = TRUE)
```

Arguments

fossils	an object of class "fossils" that corresponds to fossil occurrences.
tree	an object of class "Phylo", representing the tree upon which the fossil occurrences were simulated.
complete	logical, if TRUE the youngest sample from each clade in the complete tree is returned, if FALSE the youngest sample from each clade in the extant only counterpart tree is returned.

Value

an object of class "fossils" containing the subsampled fossil occurrences.

Examples

```
t = TreeSim::sim.bd.taxa(10, 1, 0.1, 0.05)[[1]]
f = sim.fossils.poisson(0.1, t, root.edge = FALSE)
subsample.fossils.youngest(f, t, complete = FALSE)
```

summary.taxonomy

Display taxonomy object

Description

Display taxonomy object

Usage

```
## S3 method for class 'taxonomy'
summary(object, max.length = 50, round.x = 12,
        details = TRUE, ...)
```

Arguments

object	Taxonomy object.
max.length	Max number of rows to print out.
round.x	Number of decimal places to be used for species and edge ages.
details	If TRUE include summary statistics.
...	Additional parameters to be passed to summary.default.

taxonomy	<i>Taxonomy object</i>
----------	------------------------

Description

Create a taxonomy object relating species identity to a phylo object.

Usage

taxonomy(data)

as.taxonomy(data)

is.taxonomy(data)

Arguments

data Dataframe of species taxonomy. See Details for the list of required fields.

Details

The taxonomy object includes the following 6 fields for each edge in the corresponding phylo object:

- sp true species identity label. If all species originated via budding or bifurcation this will always correspond to the terminal-most edge label (i.e. the youngest node) associated with each species. This is not the case if the data set also contains anagenetic species, when multiple species may be associated with a single edge
- edge edge label of the branch in the corresponding phylo object. Note some species may be associated with multiple edges
- parent = ancestor of species sp. Parent labels follow the same convention as species. The label assigned to the parent of the origin or root will be zero
- start = start time of the corresponding edge and/or origin time of the species. If the corresponding edge is also the oldest edge associated with the species this value will equal the species origination time. If speciation mode is asymmetric or symmetric the speciation time will match the start time of the corresponding edge. If speciation mode is anagenetic the speciation time will be younger than the start time of the corresponding edge
- end = end time of the corresponding edge and/or end time of the species. If the corresponding edge is also the youngest edge associated with the species this value will equal the species end time. Unless the species end time coincides with an anagenetic speciation event, the species end time will match the end time of the corresponding edge. If the species end time coincides with an anagenetic speciation event, the speciation time will be older than the end time of the corresponding edge
- mode = speciation mode. "o" = origin or "r" = root (the edge/species that began the process). "b" = asymmetric or budding speciation. "s" = symmetric or bifurcating speciation. "a" = anagenetic speciation

Optional fields:

- `cryptic` TRUE if the speciation event was cryptic. If missing the function assumes `cryptic = FALSE`
- `cryptic.id` = cryptic species identity. If `cryptic = TRUE` `cryptic.id` will differ from the true species identity `sp`

tree.max

Find the maximum age in a phylo object (root age or origin time)

Description

Function returns the the root age or the origin time (if `root.edge = TRUE`).

Usage

```
tree.max(tree, root.edge = TRUE)
```

Arguments

<code>tree</code>	Phylo object.
<code>root.edge</code>	If TRUE include the root edge (default = TRUE).

Value

max age

Examples

```
t = ape::rtree(6)
tree.max(t, root.edge = FALSE)
```

Index

- *Topic **Poisson**
 - sim.fossils.poisson, 32
- *Topic **birth**
 - sim.fbd.age, 25
 - sim.fbd.rateshift.taxa, 26
 - sim.fbd.taxa, 27
- *Topic **death**
 - sim.fbd.age, 25
 - sim.fbd.rateshift.taxa, 26
 - sim.fbd.taxa, 27
- *Topic **fossilized**
 - sim.fbd.age, 25
 - sim.fbd.rateshift.taxa, 26
 - sim.fbd.taxa, 27
- *Topic **fossil**
 - sim.fossils.environment, 28
 - sim.fossils.intervals, 30
 - sim.gradient, 33
- *Topic **non-uniform**
 - sim.fossils.environment, 28
 - sim.fossils.intervals, 30
 - sim.gradient, 33
- *Topic **preservation**
 - sim.fossils.environment, 28
- *Topic **preservation**
 - sim.fossils.intervals, 30
 - sim.gradient, 33
- *Topic **sampling**
 - sim.fossils.poisson, 32
- *Topic **uniform**
 - sim.fossils.intervals, 30
- as.fossils (fossils), 5
- as.taxonomy (taxonomy), 43
- beast.fbd.format, 3
- count.fossils, 4
- count.fossils.binned, 4
- fossils, 5, 7, 12
- fossils.to.BEAST.constraints, 5
- fossils.to.BEAST.start.tree, 6
- fossils.to.paleotree.record, 7, 12
- fossils.to.pyrate, 8
- FossilSim, 9
- FossilSim-package (FossilSim), 9
- get.tip.descs, 11
- is.fossils (fossils), 5
- is.taxonomy (taxonomy), 43
- paleotree.record.to.fossils, 7, 11
- place.fossils, 12
- plot.fossils, 13
- plot.taxonomy, 15
- prune.fossil.tips, 16
- prune.fossils, 17
- rangeplot.asymmetric, 18
- reconcile.fossils.taxonomy, 19
- remove.stem.fossils, 20
- remove.stem.lineages, 20
- sampled.tree.from.combined, 21
- SAtree, 22
- SAtree.from.fossils, 22
- sim.anagenetic.species, 23
- sim.cryptic.species, 24
- sim.extant.samples, 24
- sim.fbd.age, 25
- sim.fbd.rateshift.taxa, 26
- sim.fbd.taxa, 27
- sim.fossils.environment, 28, 31–33
- sim.fossils.intervals, 30, 30, 32
- sim.fossils.poisson, 30, 31, 32
- sim.gradient, 33
- sim.interval.ages, 34
- sim.taxonomy, 35
- sim.tip.samples, 36
- sim.trait.values, 30, 32, 37

species.end, [38](#)
species.start, [39](#)
subsample.fossils.oldest, [39](#)
subsample.fossils.oldest.and.youngest,
[40](#)
subsample.fossils.uniform, [41](#)
subsample.fossils.youngest, [41](#)
summary.taxonomy, [42](#)

taxonomy, [7](#), [12](#), [23](#), [24](#), [36](#), [43](#)
tree.max, [44](#)