Package 'SiPhyNetwork'

July 21, 2025

Title A Phylogenetic Simulator for Reticulate Evolution

Version 1.1.0

Description

A simulator for reticulate evolution under a birth-death-hybridization process. Here the birth-death process is extended to consider reticulate Evolution by allowing hybridization events to occur. The general purpose simulator allows the modeling of three different reticulate patterns: lineage generative hybridization, lineage neutral hybridization, and lineage degenerative hybridization. Users can also specify hybridization events to be dependent on a trait value or genetic distance. We also extend some phylogenetic tree utility and plotting functions for networks. We allow two different stopping conditions: simulated to a fixed time or number of taxa. When simulating to a fixed number of taxa, the user can simulate under the Generalized Sampling Approach that properly simulates phylogenies when assuming a uniform prior on the root age.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.3

Suggests testthat, knitr, rmarkdown, markdown

Imports stats, ape, Rcpp, rstackdeque, lifecycle

VignetteBuilder knitr

LinkingTo Rcpp,

NeedsCompilation yes

Author Joshua Justison [aut, cre, cph],
Claudia Solis-Lemus [aut, cph],
Tracy A. Heath [aut, cph],
Klaus Schliep [cph] (modified code from ape in 'write.net.R' and 'read.net.R'. Crediting original copyright holder),
Emmanuel Paradis [cph] (modified code from ape in 'write.net.R' and 'read.net.R'. Crediting original copyright holder),
Daniel Lawson [cph] (modified code from ape in 'write.net.R' and 'read.net.R'. Crediting original copyright holder)

Maintainer Joshua Justison <justisnojo@gmail.com>

Repository CRAN

Date/Publication 2023-04-14 20:10:02 UTC

Contents

`biconnectedComponents` *Biconnected Components*

Description

Find the biconnected components of a phylogeny

Usage

```
biconnectedComponents(edges, rt, nNode)
```

Arguments

`edges`	The edge matrix of a `phylo` object.
`rt`	The root node of the phylogeny
`nNode`	The number of nodes in the tree

Details

Find the biconnected components of a phylogeny

biconnected components

Value

A list with containing a vector of nodes for each biconnected component

---

`deleteTips` *Remove tips from a phylogenetic Network*

Description

This function removes certain tips from a phylogenetic network, returning the pruned network.

Usage

```
deleteTips(net, tips)
```

Arguments

`net`	An object of class 'evonet.'
`tips`	A numeric vector specifying the tip numbers to delete

Value

net The network tips removed.

Examples

```
set.seed(17) ##Set seed with smallest Quartran Prime
net<-sim.bdh.age(1,1,5,2,1,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),complete=FALSE)[[1]]
net<- deleteTips(net,c(1,6)) ##drop tips 1 and 6
```

getNetworkLevel *Get the level of a Network*

Description

This function gets the level of the network

Usage

```
getNetworkLevel(net)
```

Arguments

net	A phylogenetic network of class evonet.

Value

A numeric with the level of the network

Examples

```
net<- read.net(text= "((A:7,((B:2,C:2):3)#H1:2::0.6):3,(D:6,#H1:1::0.4):4);")
getNetworkLevel(net) ##returns 1
```

incompleteSampling *Sample Tips on a Phylogenetic Network*

Description

This function samples tips from a network. Only extant tips are downsampled from the network. Extinct tips, if they are present, will be unchanged.

Usage

```
incompleteSampling(net, rho, stochastic = FALSE)
```

Arguments

net	An object of class 'evonet.'
rho	The sampling probability.
stochastic	If stochastic=FALSE then for a network with n tips we sample n*rho tips. If stochastic=TRUE then each tip probability rho of being sampled.

Value

net A network with sampled tips

Examples

```
set.seed(23) ##set seed with the smallest Pillai prime
net<-sim.bdh.age(1,1,3,2,0.125,c(1/3,1/3,1/3),
hyb.inher.fxn = make.uniform.draw(),complete = FALSE)[[1]]
net<-incompleteSampling(net,0.5,stochastic=FALSE) ##randomly sample half of the extant taxa
```

`isFUstable` *Determine whether a phylogeny is FU-stable*

Description

This function assesses whether a network is FU-stable

Usage

```
isFUstable(net)
```

Arguments

`net`	A phylogenetic network of class `evonet`.

Value

A logical that is `TRUE` if the network is FU-stable

Examples

```
net<- read.net(text= "((A:7,((B:2,C:2):3)#H1:2::0.6):3,(D:6,#H1:1::0.4):4);")
isFUstable(net) ##returns TRUE
```

`isNormal` *#' Determine whether a phylogeny is Normal*

Description

This function assesses whether a network is Normal

Usage

```
isNormal(net)
```

Arguments

`net`	A phylogenetic network of class `evonet`.

Value

A logical that is TRUE if the network is Normal

Examples

```
net<- read.net(text= "((A:7,((B:2,C:2):3)#H1:2::0.6):3,(D:6,#H1:1::0.4):4);")
isNormal(net) ##returns TRUE
```

---

isTreeBased *Determine whether a network is tree-based*

Description

This function determines whether a network is tree-based

Usage

```
isTreeBased(net)
```

Arguments

net	A phylogenetic network of class evonet.

Details

A phylogenetic network is said to be tree-based if it can be constructed with a base tree that has additional linking arcs added. See jetten 2016 Corollary 2.11 for the algorithm used to determine whether the network is tree-based

Value

A logical that is TRUE if the network is tree-based

Examples

```
net<- read.net(text= "((A:7,((B:2,C:2):3)#H1:2::0.6):3,(D:6,#H1:1::0.4):4);")
isTreeBased(net) ##returns TRUE
```

isTreeChild *Determine whether a network is tree-child*

Description

This function determines whether a network is tree-child

Usage

```
isTreeChild(net)
```

Arguments

net	A phylogenetic network of class evonet.

Details

A phylogenetic network is said to be tree-child if all internal nodes have at least one tree-like or leaf node as children.

Value

A logical that is TRUE if the network is tree-child

Examples

```
net<- read.net(text= "((A:7,((B:2,C:2):3)#H1:2::0.6):3,(D:6,#H1:1::0.4):4);")
isTreeChild(net) ##returns TRUE
```

ltt.network *Lineages thru time on a network*

Description

This function Computes the number of lineages thru time on a network

Usage

```
ltt.network(phy, node_times = NULL)
```

Arguments

phy	An object of class 'evonet.'
node_times	A numeric vector specifying times of each node. If left NULL then the function will use the output from node.depth.edgelength(phy)

Value

A dataframe that consists of intervals. The first column denotes the start time of the interval while the second column denotes the end time. The third column depicts the number of lineages present in that interval. NOTE: due to computational precision, two nodes that appear to occur on the same time (as in the case of lineage neutral and generative hybridization) may be part of different intervals in the output data frame.

Examples

```
set.seed(17) ##smallest Quartan prime as seed
##Generate a tree with extinct leaves
net<-sim.bdh.age(1,1,5,2,1,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),complete=TRUE)[[1]]
ltt.network(net)
```

---

make.beta.draw *Function that makes a draw from a beta distribution*

Description

Create a function that makes draws from a beta distribution. Function calls `rbeta(1,alpha,beta)`

Usage

```
make.beta.draw(alpha, beta)
```

Arguments

`alpha`	The first shape parameter of the beta distribution
`beta`	The second shape parameter of the beta distribution

Value

A function that makes draws from a beta distribution.

Examples

```
set.seed(17)
inher_func<-make.beta.draw(10,10)
net<-sim.bdh.age(1,1,5,2,1,c(1/3,1/3,1/3),hyb.inher.fxn = inher_func,
complete=TRUE)[[1]]
```

make.categorical.draw *Function that makes a draw from a categorical distribution*

Description

Create a function that makes draws from a categorical distribution. Function calls `sample(x=inheritances,size=1,prob=w`

Usage

```
make.categorical.draw(inheritances, weights)
```

Arguments

`inheritances`	A vector of inheritance probabilities
`weights`	A vector of weights for each inheritance probability

Value

A function that makes draws from a categorical distribution.

Examples

```
set.seed(17)
inher_func<-make.categorical.draw(inheritances=c(0.25,0.50,0.75),weights=c(0.25,0.5,0.25))
net<-sim.bdh.age(1,1,5,2,1,c(1/3,1/3,1/3),hyb.inher.fxn = inher_func,
complete=TRUE)[[1]]
```

make.exp.decay *Make an exponential decay function*

Description

Create an exponential decay function for genetic distance of two taxa and the probability of success of a hybridization event

Usage

```
make.exp.decay(t = 1, s = 1)
```

Arguments

`t`	A numeric representing how quickly the hybridization success decays. Samaller values denote a quicker decay
`s`	A numeric for the power that the genetic distance is raised.

Details

The function computes:

$$e^{-\frac{d^s}{t}}$$

where d is the genetic distance between taxa

Value

An exponential decay function

Examples

```
set.seed(17)
dist_func<- make.exp.decay(1,1)
net<-sim.bdh.age(1,1,5,2,2,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),
hyb.rate.fxn=dist_func,complete=TRUE)[[1]]
```

`make.linear.decay` *Make a linear decay function*

Description

Create a linear decay function for genetic distance of two taxa and the probability of success of a hybridization event

Usage

```
make.linear.decay(threshold)
```

Arguments

`threshold`	A numeric representing how quickly the hybridization success decays. Smaller values denote a quicker decay

Details

The function computes:

$$1 - \frac{d}{t}$$

where d is the genetic distance between taxa

Value

A linear decay function

Note

a distance d greater than t will return 0

Examples

```
set.seed(17)
dist_func<- make.linear.decay(0.5)
net<-sim.bdh.age(1,1,5,2,2,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),
hyb.rate.fxn=dist_func,complete=TRUE)[[1]]
```

`make.polynomial.decay` *Make a polynomial decay function*

Description

Create a polynomial decay function for genetic distance of two taxa and the probability of success of a hybridization event

Usage

```
make.polynomial.decay(threshold, degree = 1)
```

Arguments

`threshold`	A numeric denoting how quickly the polynomial function decays. Distances greater than the threshold will return a success probability of 0.
`degree`	The degree of the polynomial

Details

The function computes:

$$1 - \frac{d}{t}^{d} egree$$

Where d is the distance and t is the threshold

Value

An polynomial decay function

Examples

```
set.seed(17)
dist_func<- make.polynomial.decay(0.5,2)
net<-sim.bdh.age(1,1,5,2,2,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),
hyb.rate.fxn=dist_func,complete=TRUE)[[1]]
```

make.stepwise *Make a stepwise decay function*

Description

Create a stepwise decay function for genetic distance of two taxa and the probability of success of a hybridization event

Usage

```
make.stepwise(probs, distances)
```

Arguments

probs	Vector of dimension k, where k is the number of different probabilities of success. An individual time between (distances[i-1],distances[i]] has probability of success prob[i]
distances	Vector of k, containing the end of each interval where success probabilities shift. The first interval where success is prob[1] is [0,distances[1]]. For all i>1, the probability of success is prob[i] over the interval (distance[i-1],distances[i]].

Value

An stepwise decay function

Examples

```
set.seed(17)
dist_func<- make.stepwise(probs=c(1,0.5,0),distances=c(0.25,0.5,Inf))
net<-sim.bdh.age(1,1,5,2,2,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),
hyb.rate.fxn=dist_func,complete=TRUE)[[1]]
```

make.trait.model *Model for trait evolution across the phylogeny*

Description

Create a model that dictates how a discrete or continuous trait evolves and affects the diversification of the phylogeny. This function creates a list that dictates how the trait affects hybridizations, how the trait is changes over time, and how the trait is inherited across speciation and hybridization events.

Usage

```
make.trait.model(
  initial_states,
  hyb.event.fxn,
  hyb.compatibility.fxn,
  time.fxn = NULL,
  spec.fxn = NULL
)
```

Arguments

initial_states	the initial state on the phylogeny. if simulating networks with twolineages=TRUE then a vector of length two will be required.
hyb.event.fxn	A function that describes how the trait changes after hybridization events. See Details for more information
hyb.compatibility.fxn	A function that describes whether hybridization events can occur between taxa based on their trait values. See Details for more information
time.fxn	A function that describes how trait values changes over time. See Details for more information
spec.fxn	A function that describes how trait values change at speciation events. See Details for more information

Details

hyb.event.fxn is a function that denotes the trait value of a hybrid child after a hybridization event. The function should have the argument parent_states, a vector with the trait states of the two parents to the hybrid child and inheritance. parent_states is vector with the states of the hybrid parents while inheritance is the inheritance probability of the first lineage denoted in parent_states. The function should return a single value for the trait state of the hybrid child.

hyb.compatibility.fxn describes when hybridization events can occur between two taxa based on their trait values. The function should have the arguments parent_states. The function should return TRUE for when a hybridization event is allowed to proceed and FALSE otherwise.

time.fxn is a function that describes how trait values change over time. The function should have the arguments trait_states and timestep in that order. trait_states is a vector containing the ploidy of all taxa while timestep is the amount of time given for ploidy evolution. The function should return a vector with the updated ploidy states of all taxa. The default value of NULL indicates that trait values will not evolve within a lineage over time. **NOTE:** Values can still change at speciation or hybridization events if allowed.

spec.fxn is a function that describes how trait values change at speciation events. The function should have the argument tip_state which has the state of the lineage just before speciation. The function should return a vector with two values, one denoting the trait of each of the two new species after the event. The default value of NULL causes the two children lineage to inherit the same trait value as the parental lineage

Value

A model for trait evolution to be used as the trait.model argument in a 'sim.bdh function"

Examples

```
initial_val<-2 ## The root starts off at 2N

###function for what happens at hybridization event
hyb_e_fxn <- function(parent_states,inheritance){
 ##For allopolyploidy we add the ploidy of both parents
 return(sum(parent_states))
}

##Function for determining whether hybridization occurs
hyb_c_fxn <-function(parent_states,hybrid_state){
 ##Hybridization occurs only when the ploidy is the same
 return(parent_states[1]==parent_states[2])
}


##Function for how the trait changes over time
t_fxn <- function(trait_states,timestep){
 ##We assume that autopolyploidy occur exponentially with rate lambda
 lambda<- 2 ##Rate of autopolyploidy

 ##The number of autopolyploidy events that occur on each lineage over the timestep
 nevents<-rpois(length(trait_states),timestep)

 ##each event doubles the ploidy
 new_states<- trait_states * (2^nevents)
 return(new_states)
}

##Function for how the trait changes at speciation events
s_fxn <-function(tip_state){
 ##Ploidy doesn't change at speciation events.
 ##Both daughter lineages have the same ploidy as the parent
 return(c(tip_state,tip_state))
}

trait_model<-make.trait.model(initial_states = initial_val,
                              hyb.event.fxn = hyb_e_fxn,
                              hyb.compatibility.fxn = hyb_c_fxn,
                              time.fxn = t_fxn,
                              spec.fxn = s_fxn)
```

`make.uniform.draw` *Function that makes a draw from a uniform distribution*

Description

Create a function that makes draws from a uniform distribution. Function calls `runif(1)`

Usage

```
make.uniform.draw()
```

Value

A function that makes draws from a uniform distribution

Examples

```
set.seed(17)
net<-sim.bdh.age(1,1,5,2,1,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),
complete=TRUE)[[1]]
```

---

network.gsa *Sample under the Generalized Sampling Approach*

Description

Takes a phylogeny and samples a period where n lineages exist. This method properly samples n taxa under the GSA

Usage

```
network.gsa(net, ntaxa, complete = TRUE, frac = 1, stochsampling = FALSE)
```

Arguments

net	A network to sample phylogenies.
ntaxa	The number of desired taxa.
complete	If complete = TRUE, the tree with the extinct lineages is returned. If complete = FALSE, the extinct lineages are suppressed.
frac	Sampling fraction: The proportion of extant tips included in the phylogeny (incomplete sampling).
stochsampling	When stochsampling=TRUE: Each extant tip is included into the final tree with probability frac.

Value

A network with n extant taxa

Examples

```
set.seed(10)
ssa_net <-sim.bdh.taxa.ssa(n=20,numbsim=1,
lambda=1,mu=0.2,
nu=0.25, hybprops = c(1/3,1/3,1/3),
hyb.inher.fxn = make.beta.draw(1,1),
)[[1]]
gsa_net<-network.gsa(ssa_net,5)
```

`plottable.net` *Create a more Plotting-Friendly phylogenetic Network*

Description

This function creates a more plotting-friendly `evonet` object to be used with the `plot()` function

Usage

```
plottable.net(net, tol = 1e-08)
```

Arguments

`net`	An object of class `evonet`.
`tol`	a tolerance to account for floating point imprecision. any values smaller than `tol` are considered to be zero.

Value

a network to be used with the `plot()` function

Examples

```
#' set.seed(17) ##smallest Quartan prime as seed
##Generate a tree with extinct leaves
net<-sim.bdh.age(1,1,5,2,1,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),complete=TRUE)[[1]]
plot(plottable.net(net))
```

read.net *Read a Network from Parenthetic Format*

Description

This function reads a network from file using the Rich Newick format.

Usage

```
read.net(file = "", text = NULL, comment.char = "", ...)
```

Arguments

file	a file name specified by either a variable of mode character, or a double-quoted string; if file = "" (the default) then the tree is input on the keyboard, the entry being terminated with a blank line.
text	alternatively, the name of a variable of mode character which contains the tree(s) in parenthetic format. By default, this is ignored (set to NULL, meaning that the tree is read in a file); if text is not NULL, then the argument file is ignored.
comment.char	a single character, the remaining of the line after this character is ignored (this is passed directly to scan()).
...	further arguments to be passed to scan() and read.tree.

Details

If inheritance probabilities are included in the string, the returned evonet object will include an inheritance element. inheritance[i] corresponds to the inheritance probability of the hybrid edge denoted in reticulation[i,]

This function also accepts the optional arguments skip and tree.names. tree.names is used if there are several trees to be read and is a vector of mode character that gives names to the individual trees; if NULL (the default), the trees are named "tree1", "tree2", ... The optional argument skip denotes the number of lines of the input file to skip before beginning to read data (this is passed directly to scan()).

Value

A phylogenetic network of class evonet.

Examples

```
net<-read.net(text="((A:7,((B:2,C:2):3)#H1:2::0.6):3,(D:6,#H1:1::0.4):4);")
```

reconstructedNetwork *Remove Extinct Lineages from a Phylogenetic Network*

Description

This function removes all extinct tips from a phylogenetic network, returning the reconstructed network.

Usage

```
reconstructedNetwork(net)
```

Arguments

net	An object of class 'evonet.'

Value

net The reconstructed network with all extinct tips removed.

Examples

```
set.seed(17) ##smallest Quartan prime as seed
##Generate a tree with extinct leaves
net<-sim.bdh.age(1,1,5,2,1,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),complete=TRUE)[[1]]
recon_net<-reconstructedNetwork(net)
plot(net)
plot(recon_net)
```

sim.bdh.age *Simulate a Phylogenetic Network to a Specified Number of Taxa*

Description

Simulates a Phylogenetic Network under a birth-death-hybridization model. Simulates to a specified ages.

Usage

```
sim.bdh.age(
  age,
  numbsim,
  lambda,
  mu,
  nu,
  hybprops,
```

```
  hyb.inher.fxn,
  frac = 1,
  twolineages = FALSE,
  complete = TRUE,
  stochsampling = FALSE,
  hyb.rate.fxn = NULL,
  trait.model = NULL,
  mrca = deprecated()
)
```

Arguments

age	The time for each simulation.
numbsim	Number of networks to simulate.
lambda	Speciation rate.
mu	Extinction rate.
nu	Hybridization rate.
hybprops	Vector that represents the proportion of Hybridizations that are lineage generative, lineage degenerative, and lineage neutral respectively.
hyb.inher.fxn	A function for drawing the hybrid inheritance probabilities.
frac	Sampling fraction: The proportion of extant tips included in the phylogeny (incomplete sampling).
twolineages	If twolineages=TRUE: The process originates with two lineages that share a common ancestor. If twolineages=FALSE: The process originates with two lineages.
complete	If complete = TRUE, the tree with the extinct lineages is returned. If complete = FALSE, the extinct lineages are suppressed.
stochsampling	When stochsampling=TRUE: Each extant tip is included into the final tree with probability frac.
hyb.rate.fxn	The probability of a successful hybridization as a function of genetic distance between taxa. The default value of 'NULL" assumes that hybridization success is independent of genetic distance between taxa.
trait.model	A list that dictates how a trait affects the hybridization process. The default value of NULL doesn't take a trait into account for simulation. See Details for more information.
mrca	**[Deprecated]** Use the twolineages argument

Details

hyb.inher.fxn should return values between 0 and 1 and shouldn't require any arguments. E.g. make.beta.draw and make.uniform.draw create functions that fit these specifications

hyb.rate.fxn should take one argument for the genetic distance. The function should be defined on the range $[0, Inf)$ and return values between $[0, 1]$

trait.model is a list with the following named elements:

- `initial` The initial trait state on the phylogeny
- `hyb.event.fxn` A function that denotes the trait of a hybrid child after a hybridization event. The function should have the arguments `parent_states` and `inheritance`. `parent_states` is vector with the ploidy states of the hybrid parents while `inheritance` is the inheritance probability of the first lineage denoted in `parent_states`.
- `hyb.compatibility.fxn` A function that describes when hybridization events can occur between two taxa based on their traits. The function should have the argument `parent_states`, a vector with the trait states of the two parents to the hybrid child. The function should return `TRUE` for when a hybridization event is allowed to proceed and `FALSE` otherwise.
- `time.fxn` A function that describes how traits change over time. The function should have the arguments `trait_states` and `timestep` in that order. `trait_states` is a vector containing the ploidy of all taxa while `timestep` is the amount of time given for trait evolution. The function should return a vector with the updated ploidy states of all taxa.
- `spec.fxn` A function that describes how the trait changes at speciation events.The function should have the argument `tip_state` which has the state of the lineage just before speciation. The function should return a vector with two values, one denoting the trait of each of the two new species after the event.

Value

out Returns a list of numbsim networks with the time since origin / most recent common ancestor being 'age.' If tree goes extinct or no tips are sampled, return value is '0'. If only one extant and no extinct tips are sampled, return value is '1'. Each network has an additional attribute "inheritance" that represents the inheritance probabilities on the edges in the "reticulation" attribute.

Examples

```
##smallest Quartan prime as seed for reproducibility
set.seed(17)
#Generate a tree with extinct leaves
net<-sim.bdh.age(1,1,5,2,1,c(1/3,1/3,1/3),hyb.inher.fxn = make.uniform.draw(),complete=TRUE)[[1]]
```

`sim.bdh.taxa.gsa` *Simulate a Phylogenetic Network to a Specified Number of Taxa*

Description

Simulates a Phylogenetic Network under a birth-death-hybridization model. Simulates to a specified number of extant tips under the General Sampling Approach.

Usage

```
sim.bdh.taxa.gsa(
  m,
  n,
  numbsim,
```

```
  lambda,
  mu,
  nu,
  hybprops,
  hyb.inher.fxn,
  frac = 1,
  twolineages = FALSE,
  complete = TRUE,
  stochsampling = FALSE,
  hyb.rate.fxn = NULL,
  trait.model = NULL,
  mrca = deprecated()
)
```

Arguments

m	The number of taxa to simulate under the SSA
n	The number of taxa.
numbsim	Number of networks to simulate.
lambda	Speciation rate.
mu	Extinction rate.
nu	Hybridization rate.
hybprops	Vector that represents the proportion of Hybridizations that are lineage generative, lineage degenerative, and lineage neutral respectively.
hyb.inher.fxn	A function for drawing the hybrid inheritance probabilities.
frac	Sampling fraction: The proportion of extant tips included in the phylogeny (incomplete sampling).
twolineages	If twolineages=TRUE: The process originates with two lineages that share a common ancestor. If twolineages=FALSE: The process originates with two lineages.
complete	If complete = TRUE, the tree with the extinct lineages is returned. If complete = FALSE, the extinct lineages are suppressed.
stochsampling	When stochsampling=TRUE: Each extant tip is included into the final tree with probability frac.
hyb.rate.fxn	The probability of a successful hybridization as a function of genetic distance between taxa. The default value of 'NULL" assumes that hybridization success is independent of genetic distance between taxa.
trait.model	A list that dictates how a trait affects the hybridization process. The default value of NULL doesn't take a trait into account for simulation. See Details for more information.
mrca	**[Deprecated]** Use the twolineages argument

Details

`hyb.inher.fxn` should return values between 0 and 1 and shouldn't require any arguments. E.g. make.beta.draw and make.uniform.draw create functions that fit these specifications

`hyb.rate.fxn` should take one argument for the genetic distance. The function should be defined on the range $[0, Inf)$ and return values between $[0, 1]$

`trait.model` is a list with the following named elements:

- `initial` The initial trait state on the phylogeny
- `hyb.event.fxn` A function that denotes the trait of a hybrid child after a hybridization event. The function should have the arguments `parent_states` and `inheritance`. `parent_states` is vector with the ploidy states of the hybrid parents while `inheritance` is the inheritance probability of the first lineage denoted in `parent_states`.
- `hyb.compatibility.fxn` A function that describes when hybridization events can occur between two taxa based on their traits. The function should have the argument `parent_states`, a vector with the trait states of the two parents to the hybrid child. The function should return `TRUE` for when a hybridization event is allowed to proceed and `FALSE` otherwise.
- `time.fxn` A function that describes how traits change over time. The function should have the arguments `trait_states` and `timestep` in that order. `trait_states` is a vector containing the ploidy of all taxa while `timestep` is the amount of time given for trait evolution. The function should return a vector with the updated ploidy states of all taxa.
- `spec.fxn` A function that describes how the trait changes at speciation events.The function should have the argument `tip_state` which has the state of the lineage just before speciation. The function should return a vector with two values, one denoting the trait of each of the two new species after the event.

Value

out Returns a list of numbsim networks with the time since origin / most recent common ancestor being 'age.' If tree goes extinct or no tips are sampled, return value is '0'. If only one extant and no extinct tips are sampled, return value is '1'. Each network has an additional attribute "inheritance" that represents the inheritance probabilities on the edges in the "reticulation" attribute.

Examples

```
##smallest Quartan prime as seed for reproducibility
set.seed(17)
##Generate a tree with extinct leaves
net<-sim.bdh.taxa.gsa(m=21,n=5,1,3,2,0.5,c(1/3,1/3,1/3),
hyb.inher.fxn = make.uniform.draw(),complete=TRUE)[[1]]
```

`sim.bdh.taxa.ssa` *Simulate a Phylogenetic Network to a Specified Number of Taxa*

Description

Simulates a Phylogenetic Network under a birth-death-hybridization model. Simulates to a specified number of extant tips under the Simple Sampling Approach.

Usage

```
sim.bdh.taxa.ssa(
  n,
  numbsim,
  lambda,
  mu,
  nu,
  hybprops,
  hyb.inher.fxn,
  frac = 1,
  twolineages = FALSE,
  complete = TRUE,
  stochsampling = FALSE,
  hyb.rate.fxn = NULL,
  trait.model = NULL,
  mrca = deprecated()
)
```

Arguments

`n`	The number of taxa.
`numbsim`	Number of networks to simulate.
`lambda`	Speciation rate.
`mu`	Extinction rate.
`nu`	Hybridization rate.
`hybprops`	Vector that represents the proportion of Hybridizations that are lineage generative, lineage degenerative, and lineage neutral respectively.
`hyb.inher.fxn`	A function for drawing the hybrid inheritance probabilities.
`frac`	Sampling fraction: The proportion of extant tips included in the phylogeny (incomplete sampling).
`twolineages`	If `twolineages=TRUE`: The process originates with two lineages that share a common ancestor. If `twolineages=FALSE`: The process originates with two lineages.
`complete`	If complete = TRUE, the tree with the extinct lineages is returned. If complete = FALSE, the extinct lineages are suppressed.

stochsampling	When stochsampling=TRUE: Each extant tip is included into the final tree with probability frac.
hyb.rate.fxn	The probability of a successful hybridization as a function of genetic distance between taxa. The default value of 'NULL" assumes that hybridization success is independent of genetic distance between taxa.
trait.model	A list that dictates how a trait affects the hybridization process. The default value of NULL doesn't take a trait into account for simulation. See Details for more information.
mrca	**[Deprecated]** Use the twolineages argument

Details

hyb.inher.fxn should return values between 0 and 1 and shouldn't require any arguments. E.g. make.beta.draw and make.uniform.draw create functions that fit these specifications

hyb.rate.fxn should take one argument for the genetic distance. The function should be defined on the range $[0, Inf)$ and return values between $[0, 1]$

trait.model is a list with the following named elements:

- initial The initial trait state on the phylogeny
- hyb.event.fxn A function that denotes the trait of a hybrid child after a hybridization event. The function should have the arguments parent_states and inheritance. parent_states is vector with the ploidy states of the hybrid parents while inheritance is the inheritance probability of the first lineage denoted in parent_states.
- hyb.compatibility.fxn A function that describes when hybridization events can occur between two taxa based on their traits. The function should have the argument parent_states, a vector with the trait states of the two parents to the hybrid child. The function should return TRUE for when a hybridization event is allowed to proceed and FALSE otherwise.
- time.fxn A function that describes how traits change over time. The function should have the arguments trait_states and timestep in that order. trait_states is a vector containing the ploidy of all taxa while timestep is the amount of time given for trait evolution. The function should return a vector with the updated ploidy states of all taxa.
- spec.fxn A function that describes how the trait changes at speciation events.The function should have the argument tip_state which has the state of the lineage just before speciation. The function should return a vector with two values, one denoting the trait of each of the two new species after the event.

Value

out Returns a list of numbsim networks with the time since origin / most recent common ancestor being 'age.' If tree goes extinct or no tips are sampled, return value is '0'. If only one extant and no extinct tips are sampled, return value is '1'. Each network has an additional attribute "inheritance" that represents the inheritance probabilities on the edges in the "reticulation" attribute.

Examples

```
##smallest Quartan prime as seed for reproducibility
set.seed(17)
```

```
##Generate a tree with extinct leaves
net<-sim.bdh.taxa.ssa(5,1,5,2,1.5,c(1/3,1/3,1/3),
hyb.inher.fxn = make.uniform.draw(),complete=TRUE)[[1]]
```

`write.net` *Write a Network in Parenthetic Format*

Description

This function writes a network to file in the Extended Newick format.

Usage

```
write.net(
  net,
  file = "",
  append = FALSE,
  digits = 10,
  tree.names = FALSE,
  tol = 1e-08,
  swap.minor = TRUE
)
```

Arguments

`net`	A phylogenetic network of class `evonet`. The network may include an optional attribute 'inheritance" that represents the inheritance probabilities on the edges found in the 'reticulation' attribute
`file`	a file name specified by either a variable of mode character, or a double-quoted string; if 'file = """ (the default) then the tree is written on the standard output connection (i.e. the console).
`append`	a logical, if TRUE the tree is appended to the file without erasing the data possibly existing in the file, otherwise the file (if it exists) is overwritten ('FALSE" the default).
`digits`	a numeric giving the number of digits used for printing branch lengths.
`tree.names`	either a logical or a vector of mode character. If TRUE then any tree names will be written prior to the tree on each line. If character, specifies the name of "phylo" objects which can be written to the file.
`tol`	a numeric value giving the tolerance to consider a branch as length 0.
`swap.minor`	a logical, TRUE swaps hybrid edges around such that edges with inheritance <0.5 are always written as leaves

Details

The node labels and the root edge length, if available, are written in the file.

If inheritance probabilities are included in the network object as the 'inheritance' attribute, they are also written to file.

If tree.names == TRUE then a variant of the Newick format is written for which the name of a tree precedes the Newick format tree (parentheses are eventually deleted beforehand). The tree names are taken from the names attribute if present (they are ignored if tree.names is a character vector).

The tip labels (and the node labels if present) are checked before being printed: the leading and trailing spaces, and the leading left and trailing right parentheses are deleted; the other spaces are replaced by underscores; the commas, colons, semicolons, and the other parentheses are replaced with dashes

Value

a vector of mode character if file = "", none (invisible NULL) otherwise

Examples

```
net<-read.net(text="((A:7,((B:2,C:2):3)#H1:2::0.6):3,(D:6,#H1:1::0.4):4);")
write.net(net)
```

Index