Package 'hisse'

April 5, 2017

Version 1.8.2

Date 2017-3-23

Title Hidden State Speciation and Extinction

Author Jeremy M. Beaulieu <jbeaulieu@nimbios.org>, Brian O'Meara <bomeara@utk.edu>

Maintainer Jeremy Beaulieu <jbeaulieu@nimbios.org>

Depends ape, deSolve, GenSA, subplex, nloptr

Suggests testthat, diversitree, knitr

Imports parallel, phytools, data.table, methods

Description Sets up and executes a HiSSE model (Hidden State Speciation and Extinction) on a phylogeny and character sets to test for hidden shifts in trait dependent rates of diversification.

License GPL (>= 2)

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-04-04 22:39:14 UTC

R topics documented:

BisseToHisse *BisseToHisse*

Description

A simple utility function for organizing parameters from diversitree to be in the right format for `marginRecon` function.

Usage

```
BisseToHisse(lambda, mu, q01, q10)
```

Arguments

`lambda`	a vector containing the parameter estimates for lambda0 and lambda1 from `diversitree`.
`mu`	a vector containing the parameter estimates for mu0 and mu1 from `diversitree`.
`q01`	the estimate of transition rate from state 0 to state 1 from `diversitree`.
`q10`	the estimate of transition rate from state 1 to state 0 from `diversitree`.

Value

A simple vector providing the proper format for BiSSE parameters and defaults in HiSSE for use in the `marginRecon` function.

Author(s)

Jeremy M. Beaulieu

Examples

```
#BiSSE ancestral reconstruction
#bisse.pars <- c(0.1, 0.2, 0.03, 0.03, 0.01, 0.01)
#hisse.par.vec <- BisseToHisse(lambda=c(0.1,0.2), mu=c(0.03,0.03), q01=0.01, q10=0.01)
```

hisse *Hidden State Speciation and Extinction*

Description

Sets up and executes a HiSSE model (Hidden State Speciation and Extinction) on a phylogeny and character distribution.

Usage

```
hisse(phy, data, f=c(1,1), hidden.states=TRUE, turnover.anc=c(1,1,0,0),
eps.anc=c(1,1,0,0), trans.rate=NULL, turnover.beta=c(0,0,0,0),
eps.beta=c(0,0,0,0), timeslice=NULL, condition.on.survival=TRUE,
root.type="madfitz", root.p=NULL, output.type="turnover", sann=FALSE,
sann.its=10000, bounded.search=TRUE, max.tol=.Machine$double.eps^.25,
starting.vals=NULL, turnover.upper=50, eps.upper=50, trans.upper=100)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format and with internal nodes labeled denoting the ancestral selective regimes.
data	a data matrix containing species information (see Details).
f	vector of length 2 with the estimated proportion of extant species in state 0 and 1 that are included in the phylogeny. A value of c(0.25, 0.5) means that 25 percent of species in state 0 and 50 percent of species in state 1 are included in the phylogeny. By default all species are assumed to be sampled.
hidden.states	a logical indicating whether the model includes a hidden state. The default is FALSE.
turnover.anc	a vector of length 4, indicating the free parameters associated with the net turnover rates. Default settings is a BiSSE model with fixed turnover rates for both observed states (see Details).
eps.anc	a vector of length 4, indicating the free parameters associated with the extinction fractions. Default settings is a BiSSE model with fixed extinction fractions for both observed states (see Details).
trans.rate	provides the transition rate model.
turnover.beta	a vector of length 4, indicating the free parameters associated with time-varying net turnover rates (see Details).
eps.beta	a vector of length 4, indicating the free parameters associated with time-varying extinction fractions (see Details).
timeslice	a user-supplied time to split the tree.
condition.on.survival	a logical indicating whether the likelihood should be conditioned on the survival of two lineages and the speciation event subtending them (Nee et al. 1994). The default is TRUE.
root.type	indicates whether root prior assumption should based the procedure described by FitzJohn et al. 2009, "madfitz", assumed equal, "equal", or set to user, "user".
root.p	a vector indicating fixed root state probabilities. The default is NULL.
output.type	indicates whether the rates should be printed onscreen as the optimized variables, "turnover", transformed to reflect net diversification, "net.div", or transformed to reflect λ and μ, "raw".
sann	a logical indicating whether a two-step optimization procedure is to be used. The first includes a simulate annealing approach, with the second involving a refinement using subplex. The default is FALSE.

sann.its	a numeric indicating the number of times the simulated annealing algorithm should call the objective function.
bounded.search	a logical indicating whether or not bounds should be enforced during optimization. The default is is TRUE.
max.tol	supplies the relative optimization tolerance to subplex.
starting.vals	a vector of starting values to be used instead of the default settings. These are just three values given in the following order: turnover (1), extinction fraction (2), and a single transition rate (3)
turnover.upper	sets the upper bound for the turnover parameters.
eps.upper	sets the upper bound for the extinction fraction parameters.
trans.upper	sets the upper bound for the transition rate parameters.

Details

This function sets up and executes the HiSSE model. The model closely follows diversitree, although here we employ modified optimization procedures. For example, rather than optimizing birth and death separately, hisse optimizes orthogonal transformations of these variables: we let tau = birth+death define "net turnover", and we let eps = death/birth define the "extinction fraction". This reparameterization alleviates problems associated with overfitting when birth and death are highly correlated, but both matter in explaining the diversity pattern. As for data file format, hisse expects a two column matrix or data frame, with the first column containing the species names and the second containing the binary character information. Note that the order of the data file and the names in the "phylo" object need not be in the same order; hisse deals with this internally. Also, the character information MUST be binary (i.e., states are codes as 0 and 1), otherwise, the function will misbehave.

To setup a model, users input vectors containing values to indicate how many free parameters are to be estimated for each of the variables in the model. For example, the "turnover.anc" input vector is set by default as c(1,1,0,0). This means for state 0 and state 1, we are allowing one free parameter to define the net turnover rate (birth+death) in the model. This is essentially a BiSSE model with fixed turnover rates. Now, say we want to include separate turnover rates for both states we would simply input c(1,2,0,0). The last two entries, which in the preceding example are set to zero, correspond to the hidden states; the third entry corresponds to a hidden state associated with observed state 0, such that 0A (hidden state absent) is the first entry, and 0B (hidden state present) is the third entry. So, to set up a model with three turnover rates, where we include a free parameter for a hidden state associated with state 0 we input c(1,2,3,0). A full model would thus be c(1,2,3,4), which corresponds to four separate net turnover rates, for states 0A (first entry), 1A (second entry), 0B (third entry), and 1B (fourth entry). Extinction fraction, or "eps.anc", follows the same format, though including a zero for a state we want to include in the model corresponds to no extinction, which is the Yule equivalent. In general, we follow this format to make it easier to generate a large set of nested models. Once the model is specified, the parameters can be estimated using the subplex routine (default), or use a two-step process (i.e., sann=TRUE) that first employs a stochastic simulated annealing procedure, which is later refined using the subplex routine.

The "trans.rate" input is the transition model and has an entirely different setup than turnover and extinction rates. See TransMatMaker function for more details.

For user-specified "root.p", you should specify the probability for each state. If you are doing a hidden model, there will be four states: 0A, 1A, 0B, 1B. So if you wanted to say the root had to be state 0, you would specify "root.p = c(0.5, 0, 0.5, 0)".

Finally, the options ".beta" and "timeslice" are included, but neither have been tested – needless to say, use at your own risk (but really, though, you should probably forget that these options exist for the time being). The ".beta" provides a means for testing for time-varying rates, whereas "timeslice" splits the tree to allow the process to vary before and after some user defined time period. These options will be further developed in due course.

Value

`hisse` returns an object of class `hisse.fit`. This is a list with elements:

`$loglik`	the maximum negative log-likelihood.
`$AIC`	Akaike information criterion.
`$AICc`	Akaike information criterion corrected for sample-size.
`$solution`	a matrix containing the maximum likelihood estimates of the model parameters.
`$index.par`	an index matrix of the parameters being estimated.
`$f`	user-supplied sampling frequencies.
`$hidden.states`	a logical indicating whether a hidden state was included in the model.
`$condition.on.surivival`	a logical indicating whether the likelihood was conditioned on the survival of two lineages and the speciation event subtending them.
`$root.type`	indicates the user-specified root prior assumption.
`$root.p`	indicates whether the user-specified fixed root probabilities.
`$timeslice`	indicates whether the user-specified timeslice that split the tree.
`$phy`	user-supplied tree
`$data`	user-supplied dataset
`$iterations`	number of iterations of the likelihood search that were executed.
`$output.type`	the user-specified output.type to be printed on the screen.
`$max.tol`	relative optimization tolerance.
`$upper.bounds`	the vector of upper limits to the optimization search.
`$lower.bounds`	the vector of lower limits to the optimization search.

Author(s)

Jeremy M. Beaulieu

References

Beaulieu, J.M, and B.C. O'Meara. 2016. Detecting hidden diversification shifts in models of trait-dependent speciation and extinction. Syst. Biol. In press.

FitzJohn R.G., Maddison W.P., and Otto S.P. 2009. Estimating trait-dependent speciation and extinction rates from incompletely resolved phylogenies. Syst. Biol. 58:595-611.

Maddison W.P., Midford P.E., and Otto S.P. 2007. Estimating a binary characters effect on speciation and extinction. Syst. Biol. 56:701-710.

Nee S., May R.M., and Harvey P.H. 1994. The reconstructed evolutionary process. Philos. Trans. R. Soc. Lond. B Biol. Sci. 344:305-311.

Examples

```
## Not run
# library(diversitree)
# pars <- c(0.1, 0.2, 0.03, 0.03, 0.01, 0.01)
# set.seed(4)
# phy <- tree.bisse(pars, max.t=30, x0=0)
# sim.dat <- data.frame(names(phy$tip.state), phy$tip.state)
## Fit BiSSE equivalent:
# trans.rates.bisse <- TransMatMaker(hidden.states=FALSE)
# pp.bisse <- hisse(phy, sim.dat, hidden.states=FALSE, turnover.anc=c(1,2,0,0),
# eps.anc=c(1,2,0,0), trans.rate=trans.rates.bisse)

## Now fit HiSSE equivalent with a hidden state for state 1:
# trans.rates.hisse <- TransMatMaker(hidden.states=TRUE)
# trans.rates.hisse <- ParDrop(trans.rates.hisse, c(2,3,5,7,8,9,10,12))
# pp.hisse <- hisse(phy, sim.dat, hidden.states=TRUE, turnover.anc=c(1,2,0,3),
# eps.anc=c(1,2,0,3), trans.rate=trans.rates.hisse)
```

---

hisse.null4 *Four state trait-independent Hidden State Speciation and Extinction*

Description

Sets up and executes a four state trait-independent HiSSE model (Hidden State Speciation and Extinction) on a phylogeny and character set.

Usage

```
hisse.null4(phy, data, f=c(1,1), turnover.anc=rep(c(1,2,3,4),2),
eps.anc=rep(c(1,2,3,4),2), trans.type="equal", condition.on.survival=TRUE,
root.type="madfitz", root.p=NULL,  output.type="turnover", sann=FALSE,
sann.its=10000, bounded.search=TRUE, max.tol=.Machine$double.eps^.25,
starting.vals=NULL, turnover.upper=50, eps.upper=50, trans.upper=100)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format and with internal nodes labeled denoting the ancestral selective regimes.
data	a data matrix containing species information (see Details).
f	vector of length 2 with the estimated proportion of extant species in state 0 and 1 that are included in the phylogeny. A value of c(0.25, 0.5) means that 25 percent of species in state 0 and 50 percent of species in state 1 are included in the phylogeny. By default all species are assumed to be sampled.
turnover.anc	a vector of length 8, indicating the free parameters associated with the net turnover rates. Default setting assumes character independent diversification (see Details).

eps.anc	a vector of length 8, indicating the free parameters associated with the extinction fractions. Default setting assumes character independent diversification (see Details).
trans.type	provides the type of transition rate model. Currently this model allows two types: "equal", the default, which assumes all transitions are equal, and "three.rate", that assumes three rates (see Details).
condition.on.survival	a logical indicating whether the likelihood should be conditioned on the survival of two lineages and the speciation event subtending them (Nee et al. 1994). The default is TRUE.
root.type	indicates whether root prior assumption should based the procedure described by FitzJohn et al. 2009, "madfitz", assumed equal, "equal", or set to user, "user".
root.p	a vector indicating fixed root state probabilities. The default is NULL.
output.type	indicates whether the rates should be printed onscreen as the optimized variables, "turnover", transformed to reflect net diversification, "net.div", or transformed to reflect λ and μ, "raw".
sann	a logical indicating whether a two-step optimization procedure is to be used. The first includes a simulate annealing approach, with the second involving a refinement using subplex. The default is FALSE.
sann.its	a numeric indicating the number of times the simulated annealing algorithm should call the objective function.
bounded.search	a logical indicating whether or not bounds should be enforced during optimization. The default is is TRUE.
max.tol	supplies the relative optimization tolerance to subplex.
starting.vals	a vector of starting values to be used instead of the default settings. These are just three values given in the following order: turnover (1), extinction fraction (2), and a single transition rate (3)
turnover.upper	sets the upper bound for the turnover parameters.
eps.upper	sets the upper bound for the extinction fraction parameters.
trans.upper	sets the upper bound for the transition rate parameters.

Details

This function sets up and executes a four-state trait independent HiSSE model. The model closely follows hisse.

Like hisse, users input vectors containing values to indicate how many free parameters are to be estimated for each of the variables in the model. However, the null-four model assumes that "turnover.anc" and "eps.anc" are linked between the two observed states. Thus, users are unlikely to alter the inputs much, aside from perhaps fixing "turnover.anc" or "eps.anc" to be equal across the four hidden states, where the "turnover.anc" input vector is set as rep(c(1,1,1,1),2). For a Yule equivalent, the input vector for "eps.anc" would be rep(c(0,0,0,0),2). For how to setup a null-two model see the example code below.

For user-specified "root.p", you should specify the probability for each state. See help for "hisse" for more on other parameters for this function.

Value

`hisse` returns an object of class `hisse.fit`. This is a list with elements:

`$loglik`	the maximum negative log-likelihood.
`$AIC`	Akaike information criterion.
`$AICc`	Akaike information criterion corrected for sample-size.
`$solution`	a matrix containing the maximum likelihood estimates of the model parameters.
`$index.par`	an index matrix of the parameters being estimated.
`$f`	user-supplied sampling frequencies.
`$condition.on.surivival`	a logical indicating whether the likelihood was conditioned on the survival of two lineages and the speciation event subtending them.
`$root.type`	indicates the user-specified root prior assumption.
`$root.p`	indicates whether the user-specified fixed root probabilities.
`$phy`	user-supplied tree
`$data`	user-supplied dataset
`$output.type`	the user-specified output.type to be printed on the screen.
`$trans.type`	the user-specified transition model.
`$trans.mat`	the index matrix that specifies the free parameters in the transition model.
`$max.tol`	relative optimization tolerance.
`$upper.bounds`	the vector of upper limits to the optimization search.
`$lower.bounds`	the vector of lower limits to the optimization search.

Author(s)

Jeremy M. Beaulieu

References

Beaulieu, J.M, and B.C. O'Meara. 2016. Detecting hidden diversification shifts in models of trait-dependent speciation and extinction. Syst. Biol. In press.

Examples

```
## Not run
# library(diversitree)
# pars <- c(0.1, 0.2, 0.03, 0.03, 0.01, 0.01)
# set.seed(4)
# phy <- tree.bisse(pars, max.t=30, x0=0)
# sim.dat <- data.frame(names(phy$tip.state), phy$tip.state)

## Fit null-four HiSSE:
# pp.hisse.null <- hisse.null4(phy, sim.dat, turnover.anc=rep(c(1,2,3,4),2),
# eps.anc=rep(c(1,2,3,4),2), trans.type="equal")
```

```
## Fit null-two HiSSE model:
# trans.rates.hisse <- TransMatMaker(hidden.states=TRUE)
# trans.rates.hisse <- ParDrop(trans.rates.hisse, c(3,5,8,10))
# trans.rates.hisse[!is.na(trans.rates.hisse) & !trans.rates.hisse == 0] = 1
# pp.hisse <- hisse(phy, sim.dat, hidden.states=TRUE, turnover.anc=c(1,1,2,2),
# eps.anc=c(1,1,2,2), trans.rate=trans.rates.hisse)
```

`MarginRecon` *Ancestral State Estimation based on Marginal Reconstruction*

Description

Estimates the likeliest states for both internal nodes and tips of a phylogeny using the marginal reconstruction algorithm.

Usage

```
MarginRecon(phy, data, f, pars, hidden.states=TRUE, four.state.null=FALSE,
timeslice=NULL, condition.on.survival=TRUE, root.type="madfitz", root.p=NULL,
aic=NULL, verbose=TRUE, n.cores=NULL)
```

Arguments

`phy`	a phylogenetic tree, in ape "phylo" format and with internal nodes labeled denoting the ancestral selective regimes.
`data`	a data matrix containing species information (see Details).
`f`	vector of length 2 with the estimated proportion of extant species in state 0 and 1 that are included in the phylogeny. A value of c(0.25, 0.5) means that 25 percent of species in state 0 and 50 percent of species in state 1 are included in the phylogeny. By default all species are assumed to be known.
`pars`	vector containing the MLE of the parameters.
`hidden.states`	a logical indicating whether the model includes a hidden state. The default is `FALSE`.
`four.state.null`	a logical indicating whether the model is the null-four model. The default is `FALSE`.
`timeslice`	a user-supplied time to split the tree.
`condition.on.survival`	a logical indicating whether the likelihood should be conditioned on the survival of two lineages and the speciation event subtending them (Nee et al. 1994). The default is `TRUE`.
`root.type`	indicates whether root prior assumption should based the procedure described by FitzJohn et al. 2009, "madfitz", assumed equal, "equal", or set to user, "user".
`root.p`	a vector indicating fixed root state probabilities. The default is `NULL`.

aic	the AIC for the model being used for the reconstruction. This is used by the plotting function. The default is NULL.
verbose	a logical indicating whether progress should be printed to screen. The default is TRUE.
n.cores	specifies the number of independent processors to conduct the analysis.. The default is NULL.

Details

In this implementation the marginal probability of state i for a focal node is simply the overall likelihood of the tree and data when the state of the focal node is fixed in state i. Note that the likeliest tip states can also be estimated: we observe state 1, but the underlying state could either be state 1A or 1B. Thus, for any given node or tip we traverse the entire tree as many times as there are states in the model. As the size of the tree grows, however, these repeated tree traversals can slow the calculation down considerably. For this reason, we allow the marginal calculation to be conducted in parallel across any number of independent computer processors.

For user-specified "root.p", you should specify the probability for each state. If you are doing a hidden model, there will be four states: 0A, 1A, 0B, 1B. So if you wanted to say the root had to be state 0, you would specify "root.p = c(0.5, 0, 0.5, 0)".

See help for "hisse" for more on other parameters for this function.

Value

MarginRecon returns an object of class hisse.states. This is a list with elements:

$node.mat	the marginal probabilities calculated for each node. They are ordered based on the elements in the edge matrix in dQuotephylo format.
$tip.mat	the marginal probabilities calculated for each tip. They are ordered based on the order of tip labels in the tree.
$rate.mat	a matrix that details the rates for each state combination. This is used by the plotting function.
$phy	a phylogenetic tree in the dQuotephylo format that contains the states with the highest marginal probability at each internal node.

Author(s)

Jeremy M. Beaulieu

References

Beaulieu, J.M, and B.C. O'Meara. 2016. Detecting hidden diversification shifts in models of trait-dependent speciation and extinction. Syst. Biol. In press.

FitzJohn R.G., Maddison W.P., and Otto S.P. 2009. Estimating trait-dependent speciation and extinction rates from incompletely resolved phylogenies. Syst. Biol. 58:595-611.

Maddison W.P., Midford P.E., and Otto S.P. 2007. Estimating a binary characters' effect on speciation and extinction. Syst. Biol. 56:701-710.

Nee S., May R.M., and Harvey P.H. 1994. The reconstructed evolutionary process. Philos. Trans. R. Soc. Lond. B Biol. Sci. 344:305-311.

Examples

```
## Not run
# library(diversitree)
# pars <- c(0.1, 0.2, 0.03, 0.03, 0.01, 0.01)
# set.seed(4)
# phy <- tree.bisse(pars, max.t=30, x0=0)
# sim.dat <- data.frame(names(phy$tip.state), phy$tip.state)
## Now fit HiSSE equivalent with a hidden state for state 1:
# trans.rates.hisse <- TransMatMaker(hidden.states=TRUE)
# trans.rates.hisse <- ParDrop(trans.rates.hisse, c(2,3,5,7,8,9,10,12))
# pp.hisse <- hisse(phy, sim.dat, hidden.states=TRUE, turnover.anc=c(1,2,0,3),
# eps.anc=c(1,2,0,3), trans.rate=trans.rates.hisse)
##Now reconstruct the likeliest states under this model:
# pp <- MarginRecon(phy, sim.dat, f=c(1,1), pars=pp.hisse$solution,
# hidden.states=TRUE)
```

`plot.hisse.states` *Plotting function hisse.states objects*

Description

A plotting function for visualizing changes in states and rates over a phylogeny

Usage

```
## S3 method for class 'hisse.states'
plot(x, rate.param, do.observed.only=TRUE, rate.colors=NULL,
state.colors=NULL, edge.width.rate=5, edge.width.state=2, type="fan",
rate.range=NULL, show.tip.label=TRUE, fsize=1.0, lims.percentage.correction=0.001,
legend="tips", legend.position=c(0, 0.2, 0, 0.2), legend.cex=0.4,
legend.kernel.rates="auto", legend.kernel.states="auto",
legend.bg="cornsilk3",...)
```

Arguments

`x`	a `hisse.states` object or a list of such objects.
`rate.param`	indicates the type of rates to plot. Options include: "turnover", "net.div", "speciation", "extinction", "extinction.fraction".
`do.observed.only`	a logical indicating whether just the states should be plotted; for now, only TRUE works.
`rate.colors`	user specified colors to be used for coloring rates.
`state.colors`	user specified colors to be used for coloring states.
`edge.width.rate`	the width of the rate lines.

`edge.width.state`
: the width of the state lines, which should be set so that it is narrower than the edge width for rates.

`type`
: a character string specifying the type of phylogeny to be drawn. See ape for options, default is a circle tree ("fan").

`rate.range`
: an optional two element vector. If present, specifies the range of rate values to use for plotting.

`show.tip.label`
: a logical indicating whether tip names should be included.

`fsize`
: sets the font size for the tip labels.

`lims.percentage.correction`
: deals with cases where the limits are slightly smaller than the values due to imprecision issues.

`legend`
: indicates the type of legend. Options include: "none", "traditional", "tips", "internal", "all".

`legend.position`
: the coordinates for placing the legend.

`legend.cex`
: the text size inside the legend.

`legend.kernel.rates`
: for legend=tips, internal, or all, lets you specify the way the density plot or histogram is made for rates. "auto" chooses what we think is the best option given your data, "hist" makes a histogram, "rectangular", "gaussian", and others make a density plot. See `?density` for all non-"hist" options.

`legend.kernel.states`
: as above, for states.

`legend.bg`
: sets the color for the legend background.

`...`
: further arguments to be passed to "plot" or "plot.hisse.states".

Details

Provides phylogeny that shows a heat map of the diversification rate parameter you specify (which could be turnover, net.div, speciation, extinction, or extinction.fraction). The discrete state reconstruction appears as lines on top of the heat map. If you give a single hisse.state object, it uses that; if you give it a list of them, it will model-average the results (it assumes the trees are the same). Colors can be specified by sending a vector of colors to rate.colors or state.colors (the defaults are red to blue for rate and white to black for state). You can specify two or more colors: c("red", "gray", "blue") for example. By default the visualization uses the minimum rate on the tree for the minimum color, and the maximum rate for the maximum color, but you may want to use the same color scale across models, even if some of them have a smaller range than others. To do this, pass a vector with the minimum and maximum rate across all models to the visualization for all models and they will use the same scale. There are many options for adding a legend. A traditional legend showing what values a color corresponds to is "traditional", like what plotSimmap will show in phytools. However, we can also use the legend to show the distribution of values, rather than just a key to color. "tips" shows a density plot of states or rates at tips, "internal" a distribution at internal nodes, and "all" at all nodes in the tree. For the density or histogram plots, you can let the package pick the best visualization or choose yourself whether to use a histogram or density plot, and if the latter, what kernel you want. The legend can be moved around the overall tree plot by

using "legend.position": this is a vector that specifies the "fig" argument to "par": c(x1, x2, y1, y2), where the values are the starting and ending positions as a fraction of the overall plot. By default, the legend starts at the lower left corner and continues up 20 the rest of the plot (c(0, 0.2, 0, 0.2)): by changing values, you can make the legend larger or smaller and change its position. The heatmap code is modified slightly from Liam Revell's phytools package.

Note that the examples trees generated in diversitree, which are detailed in other functions, will cause this function to throw an error. We are still working on why this occurs. But, these phy objects can be saved to another file, reuploaded, brought into MarginRecon, and then plotted with this function just fine.

Author(s)

Brian O'Meara

References

Beaulieu, J.M, and B.C. O'Meara. 2016. Detecting hidden diversification shifts in models of trait-dependent speciation and extinction. Syst. Biol. In press.

`SimToPhylo` *Convert simulated result to a tree*

Description

Converts the $results element of the return to a phylo object

Usage

```
SimToPhylo(results, include.extinct=FALSE, drop.stem=TRUE)
```

Arguments

`results`	dataframe of results from SimulateHisse.
`include.extinct`	include (TRUE) or delete (FALSE) extinct terminal taxa.
`drop.stem`	include the lineage leading to the first saved speciation even (FALSE) or cull it (TRUE).

Details

This takes the `$results` object from a `SimulateHisse` run and converts it to an ape `phylo` object. If there are no taxa on the final tree, it returns NA; if there is one taxon, it returns a one taxon tree. This is behavior different from diversitree's tree simulators, which returns NULL in both the zero and one taxon case. Extinct taxa can be pruned or not. For simulations starting with one taxon, and/or for simulations with some extinction, the final tree can have a time when there is a single lineage before it radiates into the crown group. This stem can be included or not. The tip states are stored in `$tip.state` in the returned tree.

Value

phylo	a phylo object in cladewise order.

Author(s)

Brian O'Meara

References

Beaulieu, J.M, and B.C. O'Meara. 2016. Detecting hidden diversification shifts in models of trait-dependent speciation and extinction. Syst. Biol. In press.

Examples

```
## Not run:
simulated.result <- SimulateHisse(c(.3, .1), c(.1, 0),
matrix(c(NA, 0.2, .3, NA), nrow=2), max.taxa=35, x0=1)
par(mfcol=c(1,2))
plot(SimToPhylo(simulated.result$results, include.extinct=TRUE))
plot(SimToPhylo(simulated.result$results, include.extinct=FALSE))

## End(Not run)
```

SimulateHisse *Simulate under a HiSSE model*

Description

Flexible simulation function allowing checkpointing

Usage

```
SimulateHisse(turnover.rates, eps.values, transition.rates, max.taxa=Inf, max.t=Inf,
max.wall.time=Inf, x0, nstart=1, checkpoint.file=NULL, checkpoint.frequency=100,
checkpoint.start.object=NULL, override.safeties=FALSE)
```

Arguments

turnover.rates	a vector of turnover rates.
eps.values	a vector of extinction fractions.
transition.rates	a matrix of transition rates.
max.taxa	have the simulation stop at max.taxa surviving taxa.
max.t	have the simulation stop at max.t height (including stem).
max.wall.time	have the simulation stop at this many seconds of running.
x0	the starting state.

`nstart` how many taxa you want to start from. Usual values are 1 (start with single lineage) or 2 (crown group).

`checkpoint.file` if you want to save progress (so you can restart from last saved point) include the name of a file.

`checkpoint.frequency` if there is a checkpoint file, frequency (in terms of number of steps: births, deaths, or transitions) at which this is saved.

`checkpoint.start.object` if you are starting from a checkpointed.file, the object loaded from that file. Typically called `checkpoint.result`.

`override.safeties` simulate even if there is no limit on number of taxa, depth of tree, or wall time.

Details

Note that currently, the simulator assumes `turnover.rates`, `eps.values`, and `transition rates` are in the same state order. Remember that `turnover.rates` are birth + death rates, and `eps.values` are death / birth rates.

We strongly advise putting some limits to stop the run. These can be any combination of `max.taxa`, `max.t`, and `max.wall.time`. For example, you could set up the run to stop when you hit 1000 taxa or 3600 seconds (one hour) of running on your computer, whichever comes first. If you want to run with no limits, you have to specify `override.safeties=TRUE`, and you should know what you're doing here (the runs might never finish). There is only one starting state (`x0`), which should be the index into the rate vectors, starting with 0: that is, if your states are 0A, 0B, 1A, and 1B, x0=0 would start in 0A, x0=3 would start in 1B. `nstart` lets you choose how many taxa to start with: it could be one lineage to start from a single taxon, in which case you'll have to wait some time for the first speciation event, or you could start with two, to start simulation with a clade of this size (though, if extinction is nonzero, you are not guaranteed to have the final crown group include both of these descendants). You can choose numbers higher than two, to start with, say, a 50-species polytomy. We're not sure why you would. You can add checkpointing: runs take a long time, and if it crashes, this will let you start from the same point in the last saved file (though with a different seed, unless you specify this yourself). If you want to save checkpoints as you go along, specify a file in `checkpoint.file`. Every `checkpoint.frequency` events, where an event is a speciation, extinction, or transition event, it will save current progress to a file. The smaller this value, the more frequently the simulation will be saved: fewer lost steps if you have to re-simulate, but slower during the run because it spends time writing to disk. If you want to start from a checkpointed analysis, load the file into R and then specify the object within R [not the file] containing the checkpointed results (by default, saved in an object called `checkpoint.result`) and the simulation will continue from that point.

Results are returned in a list, containing a dataframe with the outcome of the simulation (one row per edge in the tree, and containing information on the lengths, ancestor, tipward height from the original root, and state at the tipward end of the edge (you can get state at the rootward end by getting the tipward state of the ancestral edge). It may be of interest to know how many events of each type occurred, which is saved in `$birth.counts`, `$death.counts`, `$transition.counts`. The total number of surviving taxa is stored in `$n.surviving`.

The output can be returned as an ape `phylo` tree by passing the `$results` element of the final output to `SimToPhylo()`.

Value

This returns a list, with the following elements:

`$results`	a dataframe containing the simulation output: states at nodes and tips, ancestor-descendant pairs, branch lengths.
`$birth.counts`	a vector, in the same state order as turnover.rates et al., that includes the counts for the number of times species in each state speciated.
`$death.counts`	same as birth.counts, but counting extinction events in each state.
`$transition.counts`	matrix of counts of transitions from one state to another.
`$n.surviving`	the number of taxa alive at the end of the simulation.

Author(s)

Brian O'Meara

References

Beaulieu, J.M, and B.C. O'Meara. 2016. Detecting hidden diversification shifts in models of trait-dependent speciation and extinction. Syst. Biol. In press.

Examples

```
## Not run:
simulated.result <- SimulateHisse(c(.3, .1), c(.1, 0),
matrix(c(NA, 0.2, .3, NA), nrow=2), max.taxa=35, x0=1)
par(mfcol=c(1,2))
plot(SimToPhylo(simulated.result$results, include.extinct=TRUE))
plot(SimToPhylo(simulated.result$results, include.extinct=FALSE))

## End(Not run)
```

SupportRegion *Adaptive Sampling of the Likelihood Surface*

Description

Adaptively samples points for each parameter to obtain an estimate of the confidence intervals.

Usage

```
SupportRegion(hisse.obj, n.points=1000, scale.int=0.1, desired.delta=2,
output.type="turnover", hidden.states=TRUE, condition.on.survival=TRUE,
root.type="madfitz", root.p=NULL, verbose=TRUE)
```

Arguments

hisse.obj	an object of class hisse.fit that contains the MLE from a model run.
n.points	indicates the number of points to sample.
scale.int	the scaling multiplier that defines interval to randomly sample. By default the value is set to 0.1, meaning that values are drawn at random along an interval that encompasses 10 percent above and below the MLE.
desired.delta	defines the number lnL units away from the MLE to include. By default the value is set to 2.
output.type	indicates whether the rates should be printed onscreen as the optimized variables, "turnover", transformed to reflect net diversification, "net.div", or transformed to reflect λ and μ, "raw".
hidden.states	a logical indicating whether the model includes a hidden state. The default is TRUE.
condition.on.survival	a logical indicating whether the likelihood was conditioned on the survival of two lineages and the speciation event subtending them (Nee et al. 1994). The default is TRUE.
root.type	indicates whether root prior assumption should based the procedure described by FitzJohn et al. 2009, "madfitz", assumed equal, "equal", or set to user, "user".
root.p	a vector indicating fixed root state probabilities. The default is NULL.
verbose	a logical indicating whether progress should be printed to the screen. The default is TRUE.

Details

This function provides a means for sampling the likelihood surface quickly to estimate confidence intervals that reflect the uncertainty in the MLE. The function starts with the MLE from the hisse run. It then uses a scaling multiplier to generate an interval by which to randomly alter each parameter. However, the algorithm was designed to "feel" the boundaries of the random search. In other words, when the algorithm begins to sample the hinterlands of the surface, it will know to restrict the boundary to allow sampling of more reasonable values based on the currently sampled set. The goal of this sampling process is to find points within some desired distance from the MLE; by default we assume this distance is 2 lnLik. The confidence interval can be estimated directly from these points. The full set of points tried are also provided and can be used to generate contour plots (though, it is not entirely straightforward to do so – but certainly doable).

Value

SupportRegion returns an object of class hisse.support. This is a list with elements:

$ci	the sampled confidence interval.
$points.within.region	the sampled points that within 2lnL units from the MLE.
$all.points	all points sampled by the adaptive sampler.

Author(s)

Jeremy M. Beaulieu

References

Beaulieu, J.M, and B.C. O'Meara. 2016. Detecting hidden diversification shifts in models of trait-dependent speciation and extinction. Syst. Biol. In press.

FitzJohn R.G., Maddison W.P., and Otto S.P. 2009. Estimating trait-dependent speciation and extinction rates from incompletely resolved phylogenies. Syst. Biol. 58:595-611.

Maddison W.P., Midford P.E., and Otto S.P. 2007. Estimating a binary characters effect on speciation and extinction. Syst. Biol. 56:701-710.

Nee S., May R.M., and Harvey P.H. 1994. The reconstructed evolutionary process. Philos. Trans. R. Soc. Lond. B Biol. Sci. 344:305-311.

TransMatMaker *Transition Rate matrix generator*

Description

Generates and manipulates the index of the rate parameters to be optimized

Usage

```
TransMatMaker(hidden.states=FALSE)
ParDrop(rate.mat.index=NULL, drop.par=NULL)
ParEqual(rate.mat.index=NULL, eq.par=NULL)
```

Arguments

`hidden.states`	a logical indicating whether the underlying model includes hidden states. The default is `FALSE`.
`rate.mat.index`	A user-supplied rate matrix index to be manipulated.
`drop.par`	a vector of transitions to be dropped from the model.
`eq.par`	a vector of transitions pairs to be set equal.

Details

Outputs the full index of the rate parameters that are to be optimized. The intention is that a user might want to see how the matrix is designed prior to an analysis and perhaps drop a few parameters beforehand due to some hypothesis that he or she might have. The resulting matrix is to be plugged directly into `hisse`.

Value

Returns a rate matrix index

Index