Package 'corHMM'

October 30, 2015

Version 1.17

Date 2015-10-12

Title Analysis of Binary Character Evolution

Author Jeremy M. Beaulieu <jbeaulieu@nimbios.org>, Jeffrey C. Oliver <jeffreycoliver@gmail.com>, Brian O'Meara <bomeara@utk.edu>

Maintainer Jeremy Beaulieu <jbeaulieu@nimbios.org>

Depends ape, nloptr, GenSA

Imports expm, numDeriv, corpcor, phangorn, parallel

Description Fits a hidden rates model that allows different transition rate classes on different portions of a phylogeny by treating rate classes as hidden states in a Markov process and various other functions for evaluating models of binary character evolution.

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2015-10-30 01:08:56

R topics documented:

ancRECON *Ancestral state reconstruction*

Description

Infers ancestral states based on a set of model parameters

Usage

```
ancRECON(phy,data, p, method=c("joint", "marginal", "scaled"), hrm=FALSE,
rate.cat, ntraits=NULL, charnum=NULL, rate.mat=NULL,
model=c("ER", "SYM", "ARD"), root.p=NULL)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format.
data	a data matrix containing species information (see Details).
p	a vector of transition rates to be used to estimate ancestral states.
method	method used to calculate ancestral states at internal nodes (see Details).
hrm	a logical indicating whether the underlying model is the hidden rates model (HRM). The default is FALSE.
rate.cat	specifies the number of rate categories in the HRM.
ntraits	specifies the number of traits in the data file if the underlying model is not the HRM.
charnum	specifies the number of characters in the data file used in rayDISC.
rate.mat	a user-supplied rate matrix index of parameters to be optimized.
model	if the model is not HRM, specifies the underlying model.
root.p	a vector used to fix the probabilities at the root, but "yang" and "maddfitz" can also be supplied to use the method of Yang (2006) and FitzJohn et al (2009) respectively (see details).

Details

This is a stand alone function for computing the marginal, joint, or scaled likelihoods of internal nodes for a given set of transition rates. Like all other functions contained in corHMM, the tree does not have to be bifurcating in order for analyses to be carried out. IMPORTANT: If the corDISC, corHMM, and rayDISC functions are used they automatically provide a tree with the likeliest states as internal node labels. This function is intended for circumstances where the user would like to reconstruct states based on rates estimated elsewhere (e.g. BayesTraits, Mesquite, ape).

The algorithm based on Pupko et al. (2000, 2002) is used to calculate the joint estimates of ancestral states. The marginal method was originally implemented based on a description of an algorithm by Yang (2006). The basic idea is that the tree is rerooted on each internal node, with the marginal likelihood being the probabilities of observing the tips states given that the focal node is the root. However, this takes a ton of time as the number of nodes increase. But more importantly, this

does not work easily when the model contains asymmetric rates. Here, we use the same dynamic programming algorithm as Mesquite (Maddison and Maddison, 2011), which is time linear with the number of species and calculates the marginal probability at a node using an additional up and down pass of the tree. If `scaled`, the function uses the same algorithm from ace(). Note that the `scaled` method of ace() is simply the conditional likelihoods of observing everything at or above the focal node and these should NOT be used for ancestral state estimation.

The user can fix the root state probabilities by supplying a vector to `root.p`. For example, in the two trait case, if the hypothesis is that the root is 00, then the root vector would be `root.p=c(1,0,0,0)` for state combinations 00, 01, 10, and 11, respectively. If the user supplies the flag `root.p=`"yang", then the estimated transition rates are used to set the weights at the root (see pg. 124 Yang 2006), whereas specifying `root.p=`"maddfitz" employs the same procedure described by Maddison et al. (2007) and FitzJohn et al. (2009). Note that the default `root.p=NULL` assumes equal weighting among all possible states.

Value

For the `joint`, a vector of likeliest states at internal nodes and tips. For either `marginal` or `scaled`, a matrix of the probabilities of each state for each internal node are returned.

Author(s)

Jeremy M. Beaulieu and Jeffrey C. Oliver

References

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

Maddison, W.P. and D.R. Maddison. 2011. Mesquite: a modular system for evolutionary analysis. Version 2.75 http://mesquiteproject.org

Pupko, T., I. Pe'er, R. Shamir, and D. Graur. 2000. A fast algorithm for joint reconstruction of ancestral amino-acid sequences. Molecular Biology and Evolution 17:890-896.

Pupko, T., I. Pe'er, D. Graur, M. Hasegawa, and N Friedman N. 2002. A branch-and-bound algorithm for the inference of ancestral amino-acid sequences when the replacement rate varies among sites: application to the evolution of five gene families. Bioinformatics 18:1116-1123.

Yang, Z. 2006. Computational Molecular Evolution. London:Oxford.

Examples

```
# Not run
## Load tree and trait
# data(primates)
## Obtain the marginal reconstruction for a set of parameters:
# param<-c(0.05,10,0.01,0.01,0.06,0,0.02,51.2)
# states<-ancRECON(primates$tree,primates$trait,p=param,method="marginal",
# hrm=FALSE,ntraits=2,model="ARD")
## Put likeliest states on the tree:
# pr<-apply(states$lik.anc.states,1,which.max)
# primates$tree$node.label <- pr
```

corDISC *Correlated evolution binary traits*

Description

Fits a model of correlated evolution between two or three binary traits

Usage

```
corDISC(phy,data, ntraits=2, rate.mat=NULL, model=c("ER","SYM","ARD"),
node.states=c("joint", "marginal", "scaled"), p=NULL, root.p=NULL, ip=NULL,
lb=0, ub=100, diagn=FALSE)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format.
data	a data matrix containing species information (see Details).
ntraits	specifies the number of traits to be included in the analysis.
rate.mat	a user-supplied rate matrix index of parameters to be optimized.
model	specifies the underlying model.
node.states	method used to calculate ancestral states at internal nodes (see Details).
p	a vector of transition rates. Allows the user to calculate the likelihood given a specified set of parameter values to specified as fixed and calculate the likelihood.
root.p	a vector used to fix the probabilities at the root, but "yang" and "maddfitz" can also be supplied to use the method of Yang (2006) and FitzJohn et al (2009) respectively (see details).
ip	initial values used for the likelihood search. Can be a single value or a vector of unique values for each parameter. The default is ip=1.
lb	lower bound for the likelihood search. The default is lb=0.
ub	upper bound for the likelihood search. The default is ub=100.
diagn	logical indicating whether diagnostic tests should be performed. The default is FALSE.

Details

The function takes a tree and a trait file and estimates transition rates and ancestral states for two or three binary characters (see Pagel 1994). Note, however, that rayDISC can be used to evaluate the same models as in corDISC, with the major difference being that, with rayDISC, the rate matrix would have to be manipulated using rate.mat.maker in order to remove parameters associated with dual transitions. With corDISC, the input phylogeny need not be bifurcating as the algorithm is implemented to handle multifucations. Polytomies are allowed by generalizing Felsenstein's (1981) pruning algorithm to be the product of the probability of observing the tip states of n descendant nodes, rather than two, as in the completely bifurcating case. For the trait file, the first column of

the trait file must contain the species labels to match to the tree, with the second column onwards corresponding to the binary traits of interest.

The user can fix the root state probabilities by supplying a vector to `root.p`. For example, in the two trait case, if the hypothesis is that the root is 00, then the root vector would be `root.p=c(1,0,0,0)` for state combinations 00, 01, 10, and 11, respectively. If the user supplies the flag `root.p=`"yang", then the estimated transition rates are used to set the weights at the root (see pg. 124 Yang 2006), whereas specifying `root.p=`"maddfitz" employs the same procedure described by Maddison et al. (2007) and FitzJohn et al. (2009). Note that the default `root.p=NULL` assumes equal weighting among all possible states.

We also note that scoring information that is missing for a species can be incorporated in the analysis by including an NA for that particular trait. corDISC will then set the trait vector so that the tip vector will reflect the probabilities that are compatible with our observations. For example, if the scoring for trait 1 is missing, but trait 2 is scored as 0, then the tip vector would be (1,0,1,0), for state combinations 00, 01, 10, and 11 respectively, given our observation that trait 2 is scored 0 (for a good discussion see Felsenstein 2004, pg. 255).

Value

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

`$loglik`	the maximum negative log-likelihood.
`$AIC`	Akaike information criterion.
`$AICc`	Akaike information criterion corrected for sample size.
`$ntraits`	The number of traits specified.
`$solution`	a matrix containing the maximum likelihood estimates of the transition rates.
`$solution.se`	a matrix containing the approximate standard errors of the transition rates. The standard error is calculated as the square root of the diagonal of the inverse of the Hessian matrix.
`$index.mat`	The indices of the parameters being estimated are returned. The numbers correspond to the row in the `eigvect` and can useful for identifying the parameters that are causing the objective function to be at a saddlepoint.
`$opts`	Internal settings of the likelihood search
`$data`	User-supplied dataset.
`$phy`	User-supplied tree.
`$states`	The likeliest states at each internal node.
`$tip.states`	NULL
`$iterations`	The number of iterations used by the optimization routine.
`$eigval`	The eigenvalues from the decomposition of the Hessian of the likelihood function. If any `eigval<0` then one or more parameters were not optimized during the likelihood search
`$eigvect`	The eigenvectors from the decomposition of the Hessian of the likelihood function is returned

Author(s)

Jeremy M. Beaulieu

References

Beaulieu J.M., and M.J. Donoghue 2013. Fruit evolution and diversification in campanulid angiosperms. Evolution, 67:3132-3144.

Felsenstein, J. 1981. A likelihood approach to character weighting and what it tells us about parsimony and compatibility. Biological Journal of the Linnean Society 16: 183-196.

Felsenstein J. 2004. Inferring phylogenies. Sunderland MA: Sinauer Associates.

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

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

Pagel, M. 1994. Detecting correlated evolution on phylogenies: a general method for the comparative analysis of discrete characters. Proceedings of the Royal Society, B. 255:37-45.

Examples

```
## Not run
## Load tree and data
# data(primates)

## Obtain the fit for two binary characters
# pp<-corDISC(primates$tree,primates$trait,ntraits=2,model="ARD",
# node.states="marginal", diagn=FALSE)
# pp

## State combination three is not an observed state, so for fun, let's remove
## these transitions:
# new.mat <- rate.mat.maker(hrm=FALSE, ntraits=2, model="ARD")
# new.mat <- rate.par.drop(new.mat, c(2,8,5,6))
# pp<-corDISC(primates$tree,primates$trait,ntraits=2,rate.mat=new.mat,model="ARD",
# node.states="marginal", diagn=FALSE)
# pp
```

---

corHMM *Hidden Rates Model*

Description

Estimates hidden rates underlying the evolution of a binary character

Usage

```
corHMM(phy, data, rate.cat, rate.mat=NULL, node.states=c("joint", "marginal","scaled"),
optim.method=c("subplex"), p=NULL, root.p=NULL, ip=NULL, nstarts=10, n.cores=NULL,
sann.its=5000, diagn=FALSE)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format.
data	a data matrix containing species information (see Details).
rate.cat	specifies the number of rate categories in the HRM.
rate.mat	a user-supplied rate matrix index of parameters to be optimized.
node.states	method used to calculate ancestral states at internal nodes (see Details).
optim.method	method used to perform optimization. The default is subplex, but a user can also specify twoStep is a two step process which begins with a stochastic simulated annealing search, followed by a subplex round to finish the search.
p	a vector of transition rates. Allows the user to calculate the likelihood given a specified set of parameter values to specified as fixed and calculate the likelihood.
root.p	a vector used to fix the probabilities at the root, but "yang" and "maddfitz" can also be supplied to use the method of Yang (2006) and FitzJohn et al (2009) respectively (see details).
ip	initial values used for the likelihood search. Can be a single value or a vector of unique values for each parameter. The default is ip=1.
nstarts	the number of random restarts to be performed. The default is nstarts=10.
n.cores	the number of processor cores to spread out the random restarts.
sann.its	a numeric indicating the number of times the simulated annealing algorithm should call the objective function.
diagn	logical indicating whether diagnostic tests should be performed. The default is FALSE.

Details

The function takes a tree and a trait file and estimates transition rates and ancestral states for a single binary character using the hidden rates model (HRM). The HRM is a generalization of the covarion model that allows different rate classes to be treated as "hidden" states in reconstructing ancestral character states. For example, for a model with two rate classes, slow (S) and fast (F), underlie each observed state of 0 and 1. Since we only observe states, we treat each observation as having a probability of 1 for being either in the F and S categories. In other words, a character state 0 at a tip is assumed to have a probability of 1 for being 0_S and 0_F. The likelihood function is then maximized using the bounded subplex optimization routine (optim.method=subplex) implemented in the R package nloptr, which provides a common interface to NLopt, an open-source library for nonlinear optimization. Users can also set optim.method=twoStep to specify a multi-step process that first involves a simulated annealing step followed by maximizing the likelihood using the subplex routine. In the former case, however, it is recommended that nstarts is set to a large value (e.g. 100) to ensure that the maximum likelihood solution is found. Users can set n.cores to parse the random restarts onto multiple processors.

The input phylogeny need not be bifurcating as the algorithm is implemented to handle multifucations. Polytomies are allowed by generalizing Felsenstein's (1981) pruning algorithm to be the product of the probability of observing the tip states of n descendant nodes, rather than two, as in the completely bifurcating case. The first column of the trait file must contain the species labels

to match to the tree, with the second corresponding to the binary trait of interest. Any variant of a model that assume either 1, 2, 3, 4, or 5 rate categories underlying the observed data can be evaluated. Note that for a given full model, the different rate classes are ordered from slowest (rate class R1) to fastest (rate class Rn) with respect to state 0.

The user can fix the root state probabilities by supplying a vector to `root.p`. For example, if the hypothesis is that the root is 0_S in a model with two hidden rates, then the root vector would be `root.p=c(1,0,0,0)` for state combinations 0_S, 1_S, 0_F, and 1_F, respectively. If the user supplies the flag `root.p=`"yang", then the estimated transition rates are used to set the weights at the root (see pg. 124 Yang 2006), whereas specifying `root.p=`"maddfitz" employs the same procedure described by Maddison et al. (2007) and FitzJohn et al. (2009). Note that the default `root.p=NULL` assumes equal weighting among all possible states.

Value

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

`$loglik`	the maximum negative log-likelihood.
`$AIC`	Akaike information criterion.
`$AICc`	Akaike information criterion corrected for sample size.
`$rate.cat`	The number of rate categories specified.
`$solution`	a matrix containing the maximum likelihood estimates of the transition rates. Note that the rate classes are ordered from slowest (R1) to fastest (Rn) with respect to state 0
`$solution.se`	a matrix containing the approximate standard errors of the transition rates. The standard error is calculated as the square root of the diagonal of the inverse of the Hessian matrix.
`$index.mat`	The indices of the parameters being estimated are returned. The numbers correspond to the row in the `eigvect` and can useful for identifying the parameters that are causing the objective function to be at a saddlepoint.
`$opts`	Internal settings of the likelihood search
`$data`	User-supplied dataset.
`$phy`	User-supplied tree.
`$states`	The likeliest states at each internal node. The state and rates reconstructed at internal nodes are in the order of the column headings of the rates matrix.
`$tip.states`	NULL
`$iterations`	The number of iterations used by the optimization routine.
`$eigval`	The eigenvalues from the decomposition of the Hessian of the likelihood function. If any `eigval<0` then one or more parameters were not optimized during the likelihood search
`$eigvect`	The eigenvectors from the decomposition of the Hessian of the likelihood function is returned

Author(s)

Jeremy M. Beaulieu

References

Beaulieu J.M., B.C. O'Meara, and M.J. Donoghue. 2013. Identifying hidden rate changes in the evolution of a binary morphological character: the evolution of plant habit in campanulid angiosperms. Systematic Biology 62:725-737.

Felsenstein, J. 1981. A likelihood approach to character weighting and what it tells us about parsimony and compatibility. Biological Journal of the Linnean Society 16: 183-196.

Felsenstein J. 2004. Inferring phylogenies. Sunderland MA: Sinauer Associates.

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

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

Yang, Z. 2006. Computational Molecular Evolution. Oxford Press:London.

Examples

```
## Not run
# data(primates)
## Obtain the fit of second rate class underlying a binary character:
# pp<-corHMM(primates$tree,primates$trait[,c(1,2)],rate.cat=2,node.states="marginal")
# pp
```

---

corPAINT *Binary character evolution with tree painting*

Description

Fits multiple rate models of correlated evolution between one, two, or three binary traits to paintings on branches

Usage

```
corPAINT(phy,data, ntraits=2, rate.mat=NULL, model=c("ER","SYM","ARD"),
node.states=c("joint", "marginal", "scaled"), p=NULL, root.p=NULL, ip=NULL,
lb=0, ub=100,diagn=FALSE)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format.
data	a data matrix containing species information (see Details).
ntraits	specifies the number of traits to included in the analysis.
rate.mat	a user-supplied rate matrix index of parameters to be optimized.
model	specifies the underlying model.
node.states	method used to calculate ancestral states at internal nodes (see Details).

p	a vector of transition rates. Allows the user to calculate the likelihood given a specified set of parameter values to specified as fixed and calculate the likelihood.
root.p	a vector used to fix the probabilities at the root, but "yang" and "maddfitz" can also be supplied to use the method of Yang (2006) and FitzJohn et al (2009) respectively (see details).
ip	initial values used for the likelihood search. Can be a single value or a vector of unique values for each parameter. The default is ip=1.
lb	lower bound for the likelihood search. The default is lb=0.
ub	upper bound for the likelihood search. The default is ub=100.
diagn	logical indicating whether diagnostic tests should be performed. The default is FALSE.

Details

The function fits a model that applies different transition models between one, two or three binary characters based on the user-defined painting of branches on the tree of discrete "selective regimes". The trait file must be contructed in the following way: the first column of the trait file must contain the species labels to match to the tree, with the second, and so on, corresponding to the binary traits of interest. The last column in the trait file defines the current "selective regime" for each tip. The user can fix the root state probabilities by supplying a vector to the root.p, otherwise, the program assumes the marginal probability for the root. Also, like all other functions scoring information that is missing for a species can be incorporated in the analysis by including an NA for that particular trait. NOTE THAT ALTHOUGH THIS FUNCTION SHOULD WORK, IT IS CURRENTLY BEING DEVELOPED. SO USE AT YOUR OWN RISK.

Value

corPAINT returns an object of class corPAINT. This is a list with elements:

$loglik	the maximum negative log-likelihood.
$AIC	Akaike information criterion.
$AICc	Akaike information criterion corrected for sample size.
$ntraits	The number of traits specified.
$solution	a matrix containing the maximum likelihood estimates of the transition rates.
$solution.se	a matrix containing the approximate standard errors of the transition rates. The standard error is calculated as the square root of the diagonal of the inverse of the Hessian matrix.
$index.mat	The indices of the parameters being estimated are returned. The numbers correspond to the row in the eigvect and can useful for identifying the parameters that are causing the objective function to be at a saddlepoint.
$opts	Internal settings of the likelihood search
$data	User-supplied dataset.
$phy	User-supplied tree.
$states	The likeliest states at each internal node.

$tip.states	NULL
$iterations	The number of iterations used by the optimization routine.
$eigval	The eigenvalues from the decomposition of the Hessian of the likelihood function. If any eigval<0 then one or more parameters were not optimized during the likelihood search
$eigvect	The eigenvectors from the decomposition of the Hessian of the likelihood function is returned

Author(s)

Jeremy M. Beaulieu

Examples

```
# Not run
## Load tree and data
# data(primates.paint)
## Obtain the fit for two binary characters
# pp.null<-corDISC(primates.paint$tree,primates.paint$trait,ntraits=2,model="ER",
# node.states="marginal")
# pp.null
# pp.paint<-corPAINT(primates.paint$tree,primates.paint$trait,ntraits=2,model="ER",
# node.states="marginal")
# pp.paint
```

---

examples *Example datasets*

Description

Example files for running various functions in corHMM. The "primates" dataset comes from the example files provided by BayesTraits, though here we only include a single tree with branch lengths scaled to time. The "primates.paint" dataset is the same, but with the tree painted according to hypothetical regimes. Finally, the "rayDISC.example" dataset provides an example on how polymorphic data can be coded for rayDISC.

Format

a list object that contains a tree of class "phylo" and a dataframe that contains the trait data

References

Pagel, M., and A. Meade. 2006. Bayesian analysis of correlated evolution of discrete characters by reversible-jump Markov chain Monte Carlo. American Naturalist 167:808-825.

lewisMkv *Phylogenetic inference of morphology using mkv model*

Description

An implementation of Lewis' (2001) mkv model for inferring topology and branch lengths based on morphological characters.

Usage

```
lewisMkv(phy, data, include.gamma=FALSE, ngammacats=4, include.beta=FALSE,
exclude.sites=NULL, max.tol=.Machine$double.eps^0.25, ncores=NULL)
```

Arguments

phy	a fixed rooted or unrooted topology in ape "phylo" format.
data	a data frame containing character alignment (see Details).
include.gamma	a logical indicating whether a discrete gamma should be used to allow rates to vary across sites.
ngammacats	indicates how many discrete gamma categories to use.
include.beta	a logical indicating whether a beta distribution should be used to allow for unequal equilibrium frequencies among states. NOT YET IMPLEMENTED.
exclude.sites	a vector indicating which sites to exclude from the analysis.
max.tol	supplies the relative optimization tolerance to nlopt.
ncores	specifies the number of independent processors to conduct the analysis.

Details

This function implements a maximum likelihood version of the morphological model described by Lewis (2001), including the correction for only including variable characters. I've also included an option to allow for variation across sites based on a discrete gamma distribution. At the moment the function only estimates branch lengths (and the shape parameter if include.gamma==TRUE). In future versions I will add in a topology estimation algorithm. In the meantime, this function can be used to test different topological hypotheses. For example, in my case, I wrote this function as a means of testing the position of a fossil taxon.

A couple of things to note. First, the starting branch lengths are based on the Rogers-Swofford approach used by PAUP and described in Rogers and Swofford (1998). However, rather than using their maximum parsimony reconstruction approach, I rely on the acctran parsimony implementation in phangorn. Thus, I named the approach "Rogers-Swoffordish". Second, I implicitly assume that the input tree is unrooted, but this need not be the case. In the end, however, the outputted tree will be unrooted because phangorn automatically unroots a rooted tree when doing the acctran parsimony analysis. Finally, I've included a nexus reader that accompanies this function. Essentially, it reads in a nexus file of character alignment and organizes the data in such a way that it is readable by corHMM. For more information about this nexus reader see "readNexusMorph". But in general, the input format follows rayDISC, where ambiguous characters are given as "?", and partial ambiguous

characters are concatenated with "&" separating them. However, I assume that the taxon names are provided as row names, not as a separate column.

Value

`lewisMkv` returns an object of class `lewis.mkv`. This is a list with elements:

`$loglik`	The maximum negative log-likelihood.
`$AIC`	Akaike information criterion.
`$gamma.shape`	The MLE of the discrete gamma shape parameter.
`$phy`	The resulting phylogeny with inferred branch lengths.
`$data`	User-supplied character set.
`$opts`	Internal settings of the likelihood search

Author(s)

Jeremy M. Beaulieu

References

Felsenstein, J. 1992. Phylogenies from restriction sites: A maximum-likelihood approach. Evolution 46:159-173.

Lewis, P.O. 2001. A likelihood approach to estimating phylogeny from discrete morphological character data. Systematic Biology 50: 913-925.

Rogers, J.S., and D.L. Swofford. 1998. A fast method for approximating maximum likelihoods of phylogenetic trees from nucleotide sequences. Systematic Biology 47: 77-89.

Examples

```
## Not run
# morph.data <- readNexusMorph("AsteralesPollen_MPfixed.nex")
# phy <- read.nexus("Asterales.tre")
# tree.set <- corHMM:::AddTaxonEverywhere(phy, "Tubulifloridites_lilliei")
# pp <- lewisMkv(tree.set[[1]], morph.data, exclude.sites=c(17,18), ncores=4)
```

---

plotRECON *Plot ancestral state reconstructions*

Description

Plots maximum likelihood ancestral state estimates on tree

Usage

```
plotRECON(phy, likelihoods, piecolors=NULL, cex=0.5, pie.cex=0.25, file=NULL,
height=11, width=8.5, show.tip.label=TRUE, title=NULL, ...)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format.
likelihoods	likelihoods for ancestral states (see Details).
piecolors	a vector of colors for states.
cex	specifies the size of the font for labels (if used).
pie.cex	specifies the size of the symbols to plot on tree.
file	filename to which a pdf is saved.
height	height of plot.
width	width of plot.
show.tip.label	a logical indicating whether to draw tip labels to tree. The default is TRUE.
title	an optional title for the plot.
...	Additional arguments to be passed to the plot device

Details

Plots ancestral state estimates on provided tree. The likelihoods can be the states of an object of class rayDISC or class corDISC, or the lik.anc of an object of class ace (from the ape package).

Value

A plot indicating the maximum likelihood ancestral states at each internal node.

Author(s)

Jeffrey C. Oliver

See Also

corDISC, rayDISC

Examples

```
# Not run
## Load data
# data(rayDISC.example)
## Perform ancestral state estimation, using a single rate of evolution and marginal
## reconstruction of ancestral states
# recon <- rayDISC(rayDISC.example$tree,rayDISC.example$trait,model="ER",
# node.states="marginal")
## Plot reconstructions on tree
# plotRECON(rayDISC.example$tree,recon$states,title="rayDISC Example")
```

`rate.mat.maker` *Rate matrix maker*

Description

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

Usage

```
rate.mat.maker(rate.cat, hrm=TRUE, ntraits=NULL, nstates=NULL,
model=c("ER", "SYM", "ARD"))
rate.par.drop(rate.mat.index=NULL,drop.par=NULL)
rate.par.eq(rate.mat.index=NULL, eq.par=NULL)
```

Arguments

`rate.cat`	specifies the number of rate categories in the HRM.
`hrm`	a logical indicating whether the underlying model is the hidden rates model (HRM). The default is `FALSE`.
`ntraits`	specifies the number of traits in the data file if the underlying model is not the HRM.
`nstates`	specifies the number of characters in the data file used in rayDISC.
`model`	if the model is not HRM, specifies the underlying model.
`rate.mat.index`	A user-supplied rate matrix index to be manipulated.
`drop.par`	a vector of transitions to be dropped from the model. Use `rate.mat.index` to see what correspond to which transition.
`eq.par`	a vector of transitions pairs to be set equal. Use `rate.mat.index` to see what correspond to which transition.

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 drops a few parameters beforehand due to some hypothesis that he or she might have. The resulting matrix can then be plugged directly into corHMM, corDISC, or rayDISC.

Value

Returns a rate matrix index

Author(s)

Jeremy M. Beaulieu and Jeffrey C. Oliver

Examples

```
#Generate a matrix for two binary traits:
rate.mat<-rate.mat.maker(hrm=FALSE,ntraits=2,model="ARD")
#Drop parameter 8 from the model
rate.mat<-rate.par.drop(rate.mat, drop.par=c(8))
#Set parameters 1 and 2 equal to one another:
rate.mat<-rate.par.eq(rate.mat, eq.par=c(1,2))

#Precursor model. There are many ways to do this, but here is one way
rate.mat<-rate.mat.maker(hrm=TRUE,rate.cat=2)
rate.mat<-rate.par.drop(rate.mat,c(1,3,4,6,7,8))
rate.mat<-rate.par.eq(rate.mat,c(1,2))
#Now add in a couple more connections:
rate.mat[3,2]<-1
rate.mat[2,3]<-1
#Now just use this matrix when using the corHMM function

#Here is a one way of doing a more complicated precursor:
rate.mat[3,2]<-2
rate.mat[1,3]<-3
rate.mat[2,3]<-4
#Again, just use this matrix when using the corHMM function

#Finally, here is an easier way of doing the precursor:
rate.mat<-rate.mat.maker(hrm=TRUE,rate.cat=2)
rate.mat<-rate.par.drop(rate.mat,c(1,3,4,7))
rate.mat[!is.na(rate.mat)]<-1

#Not run
# pp<-corHMM(primates$tree,primates$trait,rate.cat=2,rate.mat=rate.mat,
# node.states="marginal",diagn=FALSE)
```

---

rayDISC *Evolution of categorical traits*

Description

Fits a model of evolution for categorical traits, allowing for multi-state characters, polymorphisms, missing data, and incompletely resolved trees

Usage

```
rayDISC(phy,data, ntraits=1, charnum=1, rate.mat=NULL, model=c("ER","SYM","ARD"),
node.states=c("joint", "marginal", "scaled"), p=NULL, root.p=NULL, ip=NULL,
lb=0, ub=100, diagn=FALSE)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format.
data	a data matrix containing species information (see Details).
ntraits	specifies the number of traits to included in the analysis.
charnum	specified the character to analyze.
rate.mat	a user-supplied rate matrix index of parameters to be optimized.
model	specifies the underlying model.
node.states	method used to calculate ancestral states at internal nodes.
p	a vector of transition rates. Allows the user to calculate the likelihood given a specified set of parameter values to specified as fixed and calculate the likelihood.
root.p	a vector used to fix the probabilities at the root, but "yang" and "maddfitz" can also be supplied to use the method of Yang (2006) and FitzJohn et al (2009) respectively (see details).
ip	initial values used for the likelihood search. Can be a single value or a vector of unique values for each parameter. The default is ip=1.
lb	lower bound for the likelihood search. The default is lb=0.
ub	upper bound for the likelihood search. The default is ub=100.
diagn	logical indicating whether diagnostic tests should be performed. The default is FALSE.

Details

The function takes a tree and a trait file and estimates transition rates and ancestral states for binary or multistate characters. The first column of the trait file must contain the species labels to match to the tree, with the second, third, fourth, and so on, corresponding to the traits of interest. Use the charnum variable to select the trait for analysis. Also, the input phylogeny need not be bifurcating as the algorithm is implemented to handle multifucations. Polytomies are allowed by generalizing Felsenstein's (1981) pruning algorithm to be the product of the probability of observing the tip states of n descendant nodes, rather than two, as in the completely bifurcating case.

The user can fix the root state probabilities by supplying a vector to the root.p. If the user supplies the flag root.p="yang", then the estimated transition rates are used to set the weights at the root (see pg. 124 Yang 2006), whereas specifying root.p="maddfitz" employs the same procedure described by Maddison et al. (2007) and FitzJohn et al. (2009). Note that the default root.p=NULL assumes equal weighting among all possible states.

Ambiguities (polymorphic taxa or taxa missing data) are assigned likelihoods following Felsenstein (2004, p. 255). Polymorphic taxa are coded "&" with all states observed at a tip. For example, if a trait has four states and taxonA is observed to be in state 1 and 3, the character would be coded as "1&3". rayDISC then uses this information to assign a likelihood of 1.0 to both states. Missing data are treated as ambiguous for all states, thus all states for taxa missing data are assigned a likelihood of 1.0. For example, for a four-state character (i.e. DNA), a taxon missing data will have likelihoods of all four states equal to 1.0 [e.g. L(A)=1.0, L(C)=1.0, L(G)=1.0, L(T)=1.0].

Value

rayDISC returns an object of class rayDISC. This is a list with elements:

$loglik	the maximum negative log-likelihood.
$AIC	Akaike information criterion.
$AICc	Akaike information criterion corrected for sample size.
$ntraits	The number of traits specified.
$solution	a matrix containing the maximum likelihood estimates of the transition rates.
$solution.se	a matrix containing the approximate standard errors of the transition rates. The standard error is calculated as the square root of the diagonal of the inverse of the Hessian matrix.
$index.mat	The indices of the parameters being estimated are returned. The numbers correspond to the row in the eigvect and can useful for identifying the parameters that are causing the objective function to be at a saddlepoint.
$opts	Internal settings of the likelihood search.
$data	User-supplied dataset.
$phy	User-supplied tree.
$states	The likeliest states at each internal node.
$tip.states	NULL
$iterations	The number of iterations used by the optimization routine.
$eigval	The eigenvalues from the decomposition of the Hessian of the likelihood function. If any eigval<0 then one or more parameters were not optimized during the likelihood search.
$eigvect	The eigenvectors from the decomposition of the Hessian of the likelihood function is returned.
$bound.hit	A logical for diagnosing if rate parameters were constrained by lb or ub values during optimization.
$message.tree	A list of taxa which were listed in the data matrix, but were not present in the passed phylo object. These taxa will be excluded from the analysis. message.tree is null if all taxa in data are included in tree.
$message.data	A list of taxa which were present in the passed phylo object, but lacked data in the passed data matrix. These taxa will be coded as missing data (all states equally likely). message.data is null if all taxa in tree have entries in data matrix.

Author(s)

Jeffrey C. Oliver and Jeremy M. Beaulieu

References

Felsenstein, J. 1981. A likelihood approach to character weighting and what it tells us about parsimony and compatibility. Biological Journal of the Linnean Society 16: 183-196.

Felsenstein J. 2004. Inferring phylogenies. Sunderland MA: Sinauer Associates.

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

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

See Also

plotRECON

Examples

```
# Not run
### Example 1
# Load data
# data(rayDISC.example)

## Perform ancestral state estimation, using an asymmetric model of evolution and marginal
## reconstruction of ancestral states
# recon <- rayDISC(rayDISC.example$tree,rayDISC.example$trait,model="ARD",
# node.states="marginal")

## Plot reconstructions on tree
# plotRECON(rayDISC.example$tree,recon$states)

### Example 2
## Perform ancestral state estimation on second character, using a single-rate model of
## evolution, marginal reconstruction of ancestral states, and setting the lower bound for
##parameter estimates to 0.01
# recon <- rayDISC(rayDISC.example$tree,rayDISC.example$trait,charnum=2,model="ER",
# node.states="marginal",lb=0.01)

### Example 3
## Perform ancestral state estimation on third character, using a single-rate model of
## evolution and joint reconstruction of ancestral states
# recon <- rayDISC(rayDISC.example$tree,rayDISC.example$trait,charnum=3,
# model="ER",node.states="joint")
```

---

readNexusMorph *Reads morphological character data In NEXUS Format*

Description

This function reads a file with character scoring in the NEXUS format and provides a properly formatted data set to input into the lewisMkv function.

Usage

```
readNexusMorph(file)
```

Arguments

file	a file name specified by either a variable of mode character, or a double-quoted string

Details

This parser is really just a hacked version of "read.nexus.data" that is provided by ape. I rewrote parts to deal with completely and incompletely ambiguous characters, gaps, and so that the format of the data fits perfectly into lewisMkv. As with "read.nexus.data" this parser reads data from a file written in a *restricted* NEXUS format. I've posted on my website (http://www.jeremybeaulieu.org/r.html) an example NEXUS file properly formatted that will work with this function. But there is one major change worth mentioning here:

- Multistate characters ARE allowed. That is, NEXUS allows you to specify multiple character states at a character position either as an uncertainty or as an actual appearance of multiple states, {XY}. Examples:
  taxon 0011?110 **— OK!**
  taxon 0011-110 **— OK!**
  taxon 0011{01}110 **— OK!**
  taxon 00?1{01}1-0 **— OK!**
  taxon 0011(01)110 **— STILL NOT OK!**

Value

A data.frame of character scorings across sites for a set of species. Gaps and completely missing characters are coded as "?", and "&" separate scorings of partially ambiguous characters. Will work as a direct input for "lewisMkv".

Author(s)

Jeremy M. Beaulieu

Index