Package 'corHMM'

November 24, 2020

Version 2.5

Date 2020-11-24

Title Hidden Markov Models of Character Evolution

Maintainer Jeremy Beaulieu <jmbeauli@uark.edu>

Depends ape, nloptr, GenSA

Suggests testthat, knitr, rmarkdown

Imports expm, numDeriv, corpcor, MASS, nnet, phangorn, parallel, viridis, Rmpfr, igraph, phytools

Description Fits hidden Markov models of discrete character evolution which allow different transition rate classes on different portions of a phylogeny. Beaulieu et al (2013) <doi:10.1093/sysbio/syt034>.

License GPL (>= 2)

VignetteBuilder knitr

NeedsCompilation no

Author Jeremy Beaulieu [aut, cre],
Brian O'Meara [aut],
Jeffrey Oliver [aut],
James Boyko [aut]

Repository CRAN

Date/Publication 2020-11-24 15:50:06 UTC

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"),
rate.cat, ntraits=NULL, rate.mat=NULL,
model="ARD", root.p=NULL, get.likelihood=FALSE, get.tip.states = FALSE)
```

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. Can be one of: "joint", "marginal", or "scaled" (see Details).
rate.cat	specifies the number of rate categories in the HRM.
ntraits	currently, this is automaticall detected and can always be set to NULL.
rate.mat	a user-supplied rate matrix index of parameters to be optimized.
model	specifies the underlying model if a rate.mat is not provided ("ER", SYM", or "ARD").
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).
get.likelihood	a logical indicating whether to obtain the likelihood of the rates and states. The default is FALSE.
get.tip.states	a logical indicating whether just tip reconstructions should be output. The default is FALSE.

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, 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 analyzing a binary or multistate character, the order of root.p is the same order as the traits – e.g., for states 1, 2, 3, a `root.p=c(0,1,0)` would fix the root to be in state 2. 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.

Setting get.likelihood=TRUE will provide the user the joint likelihood of the rates and states.

Value

`$lik.tip.states`
: A matrix of the reconstructed tip values. If the number of rate.cats is greater than 2 then the probability that each observed state is in a particular hidden state is given.

`$lik.anc.states`
: For `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.

`info.anc.states`
: A vector containing the amount of information (in bits) that the tip states and model gives to each node.

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

```
data(primates)
phy <- multi2di(primates[[1]])
data <- primates[[2]]
MK_3state <- corHMM(phy = phy, data = data, rate.cat = 1)

# # one way to get the parameters from your corHMM object in the correct order
params <- sapply(na.omit(c(MK_3state$index.mat)),
function(x) na.omit(c(MK_3state$solution))[x])

# using custom params
states_1 <- ancRECON(phy = phy, data = MK_3state$data, p = params, method = "marginal",
rate.cat <- MK_3state$rate.cat, ntraits = NULL, rate.mat = MK_3state$index.mat,
root.p = MK_3state$root.p)
```

ConvertPhangornReconstructions

Convert phangorn reconstruction to a vector

Description

Converts a character reconstruction from phangorn into a vector of tip and node states. Nodes where there are equal weights among states, ties are broken at random.

Usage

```
ConvertPhangornReconstructions(x, site = 1, best = TRUE)
```

Arguments

x	The phyDat object that contains a character reconstruction from phangorn
site	The character number to convert into a vector
best	A logical indicating whether the state that maximizes some function (likelihood, parsimony, etc.) is to be returned.

Details

Creates a vector that contains the best tips and node state from a phangorn reconstruction.

---

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", "none"), lewis.asc.bias=FALSE, 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).
lewis.asc.bias	a logical indicating whether the ascertainment bias correction of Lewis et al. 2001 should be used. The default is FALSE.
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

__THIS FUNCTION IS NO LONGER NECESSARY AS IT IS NOW ENTIRELY SUBSUMED WITHIN__ `corHMM` (see _Generalized corHMM_ vignette). But we still provide it for those that are more comfortable using it than exploring the new `corHMM` function. As before, `corDISC` 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.
`$index.mat`	The indices of the parameters being estimated are returned. The numbers correspond to the row in the `eigvect` and can useful for identifying the parameters that are causing the objective function to be at a saddlepoint.
`$lewis.asc.bias`	The setting describing whether or not the Lewis ascertainment bias correction was used.

$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.

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

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

```
## 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, model = "ARD", node.states = "marginal",
fixed.nodes=FALSE, p=NULL, root.p="yang", ip=NULL, nstarts=0, n.cores=1,
get.tip.states = FALSE, lewis.asc.bias = FALSE, lower.bound = 1e-9, upper.bound = 100)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format.
data	a data.frame containing species information. The first column must be species names matching the phylogeny. Additional columns contain discrete character data.
rate.cat	specifies the number of rate categories (see Details).
rate.mat	a user-supplied index of parameters to be optimized.
model	One of "ARD", "SYM", or "ER". ARD: all rates differ. SYM: rates between any two states do not differ. ER: all rates are equal.
node.states	method used to calculate ancestral states at internal nodes (see Details).
fixed.nodes	specifies that states for nodes in the phylogeny are assumed fixed. These are supplied as node labels in the "phylo" object.
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=0.
n.cores	the number of processor cores to spread out the random restarts.
get.tip.states	a boolean indicating whether tip reconstructions should be output. The default is FALSE.

lewis.asc.bias	a boolean indicating whether to correct for observing a dataset that is not univariate. The default is FALSE.
lower.bound	lower bound for the likelihood search. The default is lower.bound=1e-9.
upper.bound	upper bound for the likelihood search. The default is upper.bound=100.

Details

This function takes a tree and a trait file and estimates transition rates and ancestral states for any number of discrete characters using a Markov model with or without "hidden" states. Users are advised to read the _Generalized corHMM_ vignette for details on how to make full use of corHMM's new functionality. In general, these models describe evolution as discrete transitions between observed states. If rate.class > 1, then the model is a hidden Markov model (HMM; also known as a hidden rates model (HRM)). The HRM is a generalization of the covarion model that allows different rate classes to be treated as "hidden" states. Essentially a hidden Markov model allows for multiple processes to describe the evolution of your observed character. This could be another (hidden) state or a large group of them. Regardless of the reason, an HMM is saying that not all observed characters are expected to act the same way.

The first column of the input data must be species names (as in the previous version), but there can be any number of data columns. If your dataset does have 2 or more columns of trait information, each column is taken to describe a separate character. The separation of character and state is an important one because corHMM will automatically remove dual transitions from your model. For example, say you had 3 characters each with 2 states (0 or 1), but only three of these combinations were ever observed 0_0_1, 0_1_0, or 1_0_0. With dual transitions disallowed, it is impossible to move between these combinations because it would mean simultaneously losing and gaining a state (0_0_1 -> 0_0_0 -> 0_1_0 in one step.) One way around this is to provide a custom rate matrix to corHMM where transitions are allowed between these states. However, this is also a case where it would seem appropriate to code the data as a single character with 3 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". corHMM 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].

The likelihood function is maximized using the bounded subplex optimization routine implemented in the R package nloptr, which provides a common interface to NLopt, an open-source library for nonlinear optimization. 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 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="NULL", then there is equal weighting among all possible states in the model. 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="yang".

Ancestral states can be estimated using marginal, joint, scaled, or none approaches. Marginal gives the likelihood of state at each node, integrating over the states at other nodes. Joint gives the optimal state at each node for the entire tree at once (it can only return the most likely state, i.e. it is not a probability like the marginal reconstruction). Scaled is included for compatibility with ape's ace() function. None suppresses calculation of ancestral states, which can dramatically speed up calculations if you're comparing models but make plotting difficult.

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.
$index.mat	The indices of the parameters being estimated are returned
$data	User-supplied dataset.
$data.legend	User-supplied dataset with an extra column of trait values corresponding to how corHMM calls the user data.
$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	The likeliest state at each tip. The state and rates reconstructed at the tips are in the order of the column headings of the rates matrix.
$states.info	a vector containing the amount of information (in bits) that the tip states and model gives to each node.
$iterations	The number of iterations used by the optimization routine.
$root.p	The root prior used in model estimation.

Author(s)

Jeremy M. Beaulieu and James D. Boyko

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.

Pagel, M. 1994. Detecting correlated evolution on phylogenies: a gneeral method for the comparative analysis of discrete characters. Proc. R. Soc. Lond. B 255:37-45.

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

Examples

```
data(primates)
phy <- multi2di(primates[[1]])
data <- primates[[2]]
MK_3state <- corHMM(phy = phy, data = data, rate.cat = 1)
MK_3state
```

`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.

getFullMat *Combines several rate class index matrices*

Description

Combines several index matrices which describe transitions between observed states into output a single index matrix for use in corHMM

Usage

```
getFullMat(StateMats, RateClassMat = NULL)
```

Arguments

StateMats	A list of index matrices describing transitions between observed states. Each unique number from 1 to n, will be independently estimated. Values of 0 are not estimated. Matrix entries of the same value are estimated to be the same rate.
RateClassMat	An optional index matrix which describes how StateMats are related to one another. This will be a matrix of size: length(StateMats) by length(StateMats). By default, all transitions between StateMats are allowed and independently estimated.

Details

This function is the final step in creating a custom hidden Markov model. It takes a list of index matrices (StateMats) which describe different ways that the observed states are related to one another and creates a single matrix to describe the model. The matrices are combined following Eq. 2 of Tarasov (2019). getFullMat is part of several functions which help the user efficiently create custom index matrices. Often, it will be more practical to begin constructing a custom model with `getRateMat4Dat`.

getStateMat will generate an index matrix of size n by n in which all transitions between the n states are allowed and independently estimated. That index matrix can then be manipulated by dropStateMatPars and equateStateMatPars. dropStateMatPars will drop specific rates from an index matrix. dropStateMatPars requires an index matrix and a vector of which rates should be dropped. equateStateMatPars will equates rates within an index matrix. equateStateMatPars requires an index matrix and a list of vectors each element of which should correspond to two or more rates to be equated.

Value

Returns an index matrix.

Author(s)

James D. Boyko

References

Tarasov, S. (2019). Integration of Anatomy Ontologies and Evo-Devo Using Structured Markov Models Suggests a New Framework for Modeling Discrete Phenotypic Traits. Systematic Biology, 68(5) 698-716. doi:10.1093/sysbio/syz005

See Also

`getRateMat4Dat`

Examples

```
data(primates)
phy <- primates[[1]]
phy <- multi2di(phy)
data <- primates[[2]]
# create a legend and rate mat from a multi-character dataset.
LegendAndRateMat <- getStateMat4Dat(data)
rate.mat <- LegendAndRateMat$rate.mat
legend <- LegendAndRateMat$legend

# To create a hidden markov model first define your rate classes (state-dependent processes)
# R1 will be a manually created SYM model
R1 <- equateStateMatPars(rate.mat, c(1:6))
# R2 will only allow transitions between 1 and 2
R2 <- dropStateMatPars(rate.mat, c(3,4))
# R1 and R2 will transtion at equal rates (i.e. the parameter process will be ER)
P <- getRateCatMat(2)
P <- equateStateMatPars(P, c(1,2))
# combine our state-dependnet processes and parameter process
HMM <- getFullMat(list(R1, R2), P)

# This can now be used in a corHMM model
CustomModel <- corHMM(phy = phy, data = data, rate.cat = 2, rate.mat = HMM, node.states = "none")
```

---

getStateMat4Dat *Produce an index matrix and legend from a dataset*

Description

Takes a dataset to produce an index matrix that corresponds to a single state-dependent process (i.e. a single rate category) and a legend which matches input data to the rows and columns of the index matrix and corHMM solution.

Usage

```
getStateMat4Dat(data, model = "ARD", dual = FALSE)
```

Arguments

data	A data matrix containing species information in the same format as the main corHMM function: column[,1] is species names, column[,2:n] are the discrete states.
model	One of "ARD", "SYM", or "ER". ARD: all rates differ. SYM: rates between any two states do not differ. ER: all rates are equal.
dual	A boolean indicating whether or not to include dual transitions.

Details

This function will generate an index matrix based on user provided data. It provides a useful starting point for further modifications using dropStateMatPars, equateStateMatPars, and getFullMat. If more than a single column of data is given double transitions between characters are disallowed. For example, if character 1 is the presence or absence of limbs, and character 2 is the presence or absence of fingers, then the transition from absence of limbs and fingers to presence of limbs and fingers is automatically disallowed. This is consistent with Pagel's (1994) model of correlated character evolution.

Value

$legend	A named vector. The elements of the vector are all the unique state combinations in the user data. The names of the vector are the state number assigned to each combination.
$rate.mat	A rate index matrix describing a single rate class.

Author(s)

James D. Boyko

References

Pagel, M. 1994. Detecting correlated evolution on phylogenies: a gneeral method for the comparative analysis of discrete characters. Proc. R. Soc. Lond. B 255:37-45.

See Also

getFullmat

Examples

```
data(primates)
phy <- primates[[1]]
phy <- multi2di(phy)
data <- primates[[2]]
# create a legend and rate mat from a multi-character dataset.
LegendAndRateMat <- getStateMat4Dat(data)
rate.mat <- LegendAndRateMat$rate.mat
legend <- LegendAndRateMat$legend
```

makeSimmap *Simulate a character history*

Description

Produces a character history given some of the outputs of a corHMM object.

Usage

```
makeSimmap(tree, data, model, rate.cat, root.p="yang", nSim=1, nCores=1)
```

Arguments

tree	A phylogeny of class phylo.
data	a data.frame containing species information. The first column must be species names matching the phylogeny. Additional columns contain discrete character data.
model	The transition rate matrix.
rate.cat	The number of rate categories.
root.p	The root prior to begin the sampling at the root. Currently only "yang" allowed.
nSim	The number of simmaps to be simulated.
nCores	The number of cores to be used.

Details

This function will generate a character history given a model and dataset. It has a similar structure to the simmap generated in phytools and follows the methods of Bollback (2006). If using hidden states, then it is necessary to reconstruct the tip probabilities as well as the node probabilities (i.e. get.tip.states must be TRUE when running corHMM). We chose not to implement any new plotting functions, instead makeSimmap produces a simmap object which is formatted so it can used with other R packages such as phytools (Revell, 2012). For additional capabilities, options, and biological examples we refer readers to the detailed _Generalized corHMM_ vignette.

Value

A list of simmaps.

Author(s)

James D. Boyko

References

Bollback, J. P. 2006. SIMMAP: stochastic character mapping of discrete traits on phylogenies. BMC Bioinformatics 7:88.

Revell, L. J. 2012. phytools: an R package for phylogenetic comparative biology (and other things). Methods in Ecology and Evolution, 3(2), 217-223.

Examples

```
data(primates)
phy <- primates[[1]]
phy <- multi2di(phy)
data <- primates[[2]]

##run corhmm
MK <- corHMM(phy, data, 1)

##get simmap from corhmm solution
model <- MK$solution
simmap <- makeSimmap(tree=phy, data=data, model=model, rate.cat=1, nSim=1, nCores=1)

## we import phytools plotSimmap for plotting
plotSimmap(simmap[[1]])
```

`plotMKmodel` *Plot a Markov model*

Description

Plots a diagram of a Markov model from the output of corHMM or a custom index matrix

Usage

```
plotMKmodel(corhmm.obj, rate.cat = NULL, display = "column", color = c("blue", "red"),
arrow.scale = 1, text.scale = 1, vertex.scale = 1)
```

Arguments

`corhmm.obj`	an object of class corHMM or matrix.
`rate.cat`	if using a custom matrix then the number of rate categories must be indicated.
`display`	the structure of the plot. one of "column", "square", or "row".
`color`	Either, 1. a vector of 2 colors to create a gradient from low transition rates (first element) to high transition rates (second element), or 2. "col.blind" which will use the color pallete "plasma" from viridis.
`arrow.scale`	determines the size of the arrows for the Markov diagram.
`text.scale`	determines the size of the text for the plotted matrix.
`vertex.scale`	determines the size of the text for the Markov diagram.

Details

Plots Markov models in a ball and stick type diagram next to its corresponding matrix. If plotting a hidden rates model it will produce a compound plot describing how the different rate classes are related to one another. If the input is a `corHMM` result then arrows are colored by relative rate. If the input is a custom matrix arrows are colored by the paramater index.

Value

Returns a ball and stick diagram of the input model.

Author(s)

James D. Boyko

Examples

```
data(primates)
phy <- primates[[1]]
phy <- multi2di(phy)
data <- primates[[2]]
# create a legend and rate mat from a multi-character dataset.
LegendAndRateMat <- getStateMat4Dat(data)
rate.mat <- LegendAndRateMat$rate.mat
legend <- LegendAndRateMat$legend

# To create a hidden markov model first define your rate classes (state-dependent processes)
# R1 will be a manually created SYM model
R1 <- equateStateMatPars(rate.mat, c(1:6))
# R2 will only allow transitions between 1 and 2
R2 <- dropStateMatPars(rate.mat, c(3,4))
# R1 and R2 will transtion at equal rates (i.e. the parameter process will be ER)
P <- getRateCatMat(2)
P <- equateStateMatPars(P, c(1,2))
# combine our state-dependnet processes and parameter process
HMM <- getFullMat(list(R1, R2), P)
# plot the input
plotMKmodel(HMM, rate.cat = 2)

# This can now be used in a corHMM model
CustomModel <- corHMM(phy = phy, data = data, rate.cat = 2, rate.mat = HMM, node.states = "none")
# plot the output
plotMKmodel(CustomModel)
```

---

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

```
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")
```

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", "none"), state.recon=c("subsequently"),
lewis.asc.bias=FALSE, p=NULL, root.p="yang", ip=NULL, lb=1e-9, ub=100, verbose=TRUE,
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.
state.recon	whether to reconstruct states jointly with the rates or subsequent to the rates being optimized.
lewis.asc.bias	a logical indicating whether the ascertainment bias correction of Lewis et al. 2001 should be used. The default is FALSE.
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=1e-9.
ub	upper bound for the likelihood search. The default is ub=100.
verbose	a logical indicating whether progress should be printed to the screen.
diagn	logical indicating whether diagnostic tests should be performed. The default is FALSE.

Details

__THIS FUNCTION IS NO LONGER NECESSARY AS IT IS NOW ENTIRELY SUBSUMED WITHIN__ `corHMM` (see _Generalized corHMM_ vignette). But we still provide it for those that are more comfortable using it than exploring the new `corHMM` function. As before, `rayDISC` 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].

In all ancestral state reconstruction implementations, the rates are first estimated, and subsequently, the MLE estimates of the rates are used to determine either the state probabilities (i.e., marginal or "scaled") or maximum likelihood states at nodes. This is the default – i.e., the state.recon="subsequently" argument. However, for this function only, we also allow for both rates and states to be estimated jointly. This can be done with state.recon="estimate". We also allow for a hypothesis about states at all or even some nodes to help fixed, with the rates (and in some cases some of the states) being estimated. This is state.recon="given". For more information please see Vignette "Getting Likelihoods From Reconstructions".

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.

`$lewis.asc.bias`
: The setting describing whether or not the Lewis ascertainment bias correction was used.

`$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.

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

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

```
### Example 1
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")
```

Index