Package 'BMhyb'

October 8, 2017

Version 1.5.2

Date 2017-10-8

Type Package

Title Hybrid Trait Evolution under Brownian Motion

Author Dwueng-Chwuan Jhwueng <djhwueng@umail.iu.edu> and Brian O'Meara <omeara.brian@gmail.com>

Maintainer Dwueng-Chwuan Jhwueng <djhwueng@umail.iu.edu>

Imports corpcor, numDeriv, geiger, ape, TreeSim, phytools, phylobase, mvtnorm, grDevices, graphics, methods, stats, lhs, viridis, Matrix, DEoptim

Suggests testthat

Description Analyzes the phenotypic evolution of species of hybrid origin on a phylogenetic network. This package can detect the hybrid vigor effect, a burst of variation at formation, and the relative portion of heritability from its parents. Parameters are estimated by maximum likelihood. Users need to enter a comparative data set, a phylogeny, and information on gene flow leading to hybrids. See Jhwueng and O'Meara (2017) <DOI:10.1101/023986>.

NeedsCompilation no

Repository CRAN

Date/Publication 2017-10-08 03:08:04 UTC

R topics documented:

`AdaptiveConfidenceIntervalSampling`
Confidence interval under adaptive cluster sampling technique.

Description

This function uses an adaptive cluster sampling technique to generate confidence interval for parameters of interest.

Usage

```
AdaptiveConfidenceIntervalSampling(par, fn, lower=-Inf, upper=Inf, desired.delta = 2,
n.points=5000, verbose=TRUE,  measurement.error=NULL, do.kappa.check=FALSE,
allow.restart=TRUE,  best.lnl = -Inf, likelihood.precision=0.001, restart.mode=FALSE, ...)
```

Arguments

par	parameter of interest for sampling in the model.
fn	the negative log-likelihood function.
lower	the lower bound for the values allowed for sampling.
upper	the upper bound for the values allowed for sampling.
desired.delta	a numeric value with a default value of 2 for the criteria that the desired log-likelihood shall no more than 2 unit away from the maximum.
n.points	number of required points for calculating likelihood value.
verbose	whether to print detailed information during the run.
measurement.error	single value or vector of fixed measurement errors.
do.kappa.check	if TRUE, check for matrix condition when calculating likelihood.
allow.restart	if TRUE, stop and go back to calling function if this finds a better value
best.lnl	the value of the best likelihood from the original run. Only relevant if allow.restart=TRUE.
likelihood.precision	a numerical value used for verifying the convergence of the estimation.
restart.mode	restart if better values found
...	further arguments passed(see details).

Details

This function starts with a set of parameter values, generates new points using the function GeneratedValues, and the likelihood value is calculated. Following an idea by Edwards (1992), a support region is based on points within a certain likelihood score of the best one (by default, 2 log likelihood units). The function tries to look over a region wide enough to encompass parameter values that are in this range. Note this is done for all traits at once. This generates wider (more conservative) confidence regions than looking at each parameter value separately. For example, if there were a ridge such that two values can covary without changing the likelihood, a univariate procedure that changes one but holds the others constant would give a narrow region, while our approach of trying many points would give a much wider region.

Value

a data frame where the first column contains the calculated likelihood values and the rest of columns are the grid points of the parameters generated under the adaptive sampling technique.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Edwards, A.W.F. 1993. Likelihood. Johns Hopkins University Press, Baltimore

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution in hybrid species on phylogenetic networks*. Submitted.

Seber G.A.F., Salehi M.M. 2013. Adaptive sampling designs: Inference for spatial and clustered population. Springer.

Examples

```
library(corpcor)
#assign the number of non hybrid taxa
ntax.nonhybrid<-8
#assign the number of hybrid
ntax.hybrid<-3
#simulate the network
network<-SimulateNetwork(ntax.nonhybrid=ntax.nonhybrid, ntax.hybrid=ntax.hybrid,
flow.proportion=0.5, origin.type='clade', birth = 1, death = 0.5, sample.f = 0.5,
tree.height = 3, allow.ghost=FALSE)

#generate the tip data
parameters <- c(0.05,1,1.7,0.005,0.5)
names(parameters) <- c("sigma.sq", "mu", "bt", "vh", "SE")
tips <- SimulateTipData(network$phy, network$flow, parameters)

#set free parameters
free.parameters<-rep(TRUE, 5)
names(free.parameters) <- c("sigma.sq", "mu", "bt", "vh", "SE")


#Simulate 100 samples
interval.results <- AdaptiveConfidenceIntervalSampling(parameters, fn=CalculateLikelihood,
lower=c(0, -Inf, 0, 0, 0)[which(free.parameters)], n.points=100,data=tips,
phy=network$phy, flow=network$flow, actual.params =
free.parameters[which(free.parameters)], allow.extrapolation=TRUE)

  #show the results
interval.results.in <- interval.results[which(interval.results[,1] -
min(interval.results[,1])<=2),]
interval.results.out <- interval.results[which(interval.results[,1] -
min(interval.results[,1])>2),]
interval.results.in
interval.results.out
```

`AdjustForDet` *Adjust the phylogeny.*

Description

Utility function to adjust the phylogeny to deal with numerical issues.

Usage

```
AdjustForDet(phy, max.attempts=100)
```

Arguments

phy	an object of class 'phylo'.
max.attempts	the maximum number of attempts to adjust the phylogeny. The default is 100.

Details

This function first calculates the determinant of the phylogeny using the function DetPass. When the determinant of the variance covariance matrix for the phylogeny is negative, this function will adjust the phylogeny by slightly lengthening the edge lengths of the tree. This procedure repeats until a well adjusted phylogeny is returned.

Value

an object of class 'phylo'.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution in hybrid species on phylogenetic networks*. Submitted.

Examples

```
library(ape)
#simulate a tree of 5 taxa
phy<-rtree(5)
#check and adjust the phylogeny
AdjustForDet(phy,max.attempts=100)
```

AICc *AICc*

Description

Calculates the second order Akaike's information criterion score for models of interest.

Usage

```
AICc(n, k, LogLik)
```

Arguments

n	number of taxa for the given phylogenetic tree. It represents the sample size(the number of species on the tip of phylogeny).
k	number of free parameters in the model.
LogLik	the minimum of the negative log-likelihood value obtained by optimizing the likelihood function.

Details

'AICc' is a function to compute the AICc values and is valid to select among different models. $AICc = 2 * n * k/(n - k - 1) - 2logL$ where L is the maximum likelihood for the model.

Value

The AICc values.

Author(s)

Dwueng-Chwuan Jhwueng

References

Burnham, K.P., and D.R. Anderson. 2004. *Model selection and inference: a practical information-theoretic approach*. Sec. Ed. Springer, New York.

Examples

```
  #assign the size
n<-5
#assign the number of parameter
k<-3
#assign the negative log likelihood value.
LogLik<- -2
#compute the AICc score
AICc(n,k,LogLik)
```

```
# result AICc value of 26.
```

AkaikeWeight *Akaike weight*

Description

Calculate Akaike weight for the models.

Usage

```
AkaikeWeight(Delta.AICc.Array)
```

Arguments

Delta.AICc.Array	Delta.AICc.Array is defined as the difference between the AICc value and the minimum AICc value among candidate models.

Details

For n models of interest, the Akaikei weight for the i th model is defined as $w = exp(-0.5 * \Delta AICc_i)/sum_i dAICc_i$ where $\Delta AICc_i$ is the difference of the AICc value between the the ith model and the best model. The weights can be used in model averaging in advanced.

Value

Akaike weights.

Author(s)

Dwueng-Chwuan Jhwueng

References

Burham, K.P., Anderson, D.R. (2002) Model selection and multimodel inference: a practical information-theoretic approach. Second edition. Springer. New York.

Examples

```
#simulate 4 AICc values for 4 models.
AICc_Array<-rnorm(4, mean=10,sd=1)
#calculate the delta AICc
Delta.AICc.Array<-AICc_Array-min(AICc_Array)
#calculate the Akaike weight
AkaikeWeight(Delta.AICc.Array)
```

`AlterMatrixUsingDE` *Transforming current variance covariance matrix to a positive definite matrix*

Description

Use differntial evolution method to search for a positive definite matrix.

Usage

```
AlterMatrixUsingDE(V.modified)
```

Arguments

`V.modified`	a matrix

Details

The function is developed based on the theoretical method in Misha (2007), we apply it to search the positive definite matrix. It first starts search with the element in upper triangular matrix of the variance covariance matrix, then tries to search the positive definite matrix with all nonnegative elements.

Value

a postive definite matrix.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Mishra, S.K. 2007. *The nearest correlation matrix problem: Solution by differential evolution method of global optimization*. dx.doi.org/10.2139/ssrn.980403.

`AttachHybridsToDonor` *Attach hybrids to the donor species*

Description

This function attaches the hybrid to the donor species.

Usage

```
AttachHybridsToDonor(phy, flow, suffix="_DUPLICATE")
```

Arguments

`phy`	a phylogeny of phylo class.
`flow`	structure of gene flow.
`suffix`	a duplicated species that used for hybrid information in advanced.

Details

This function attaches hybrid to the donor and return the new phylogeny. It first identifies the flow clades using the function `LumpIntoClades`. Then for each clade containing the flow, it searches the donor taxa and then attaches the hybrid into the clade ancestor. For empirical analysis, users shall have the correct format for the flow structure to ensure the hybrid information.

Value

a merged tree that identifies the hybrid species from the original tree.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Study trait evolution on phylogenetic networks*. Submitted.

Examples

```
#set the number of non hybrid species
ntax.nonhybrid<-4
#set the number of hybrid species
ntax.hybrid<-1
#simulate a network
network<-SimulateNetwork(ntax.nonhybrid=ntax.nonhybrid, ntax.hybrid=ntax.hybrid,
flow.proportion=0.5, origin.type='clade', birth = 1, death = 0.5, sample.f = 0.5,
tree.height = 1, allow.ghost=FALSE)
#show the tree.
```

```
network$phy
#show the flow information
network$flow
  #Attach hybrid species to the donor and show the new tree.
  #The name of the hybrid is displayed with suffix  "_DUPLICATE"
AttachHybridsToDonor(network$phy, network$flow, suffix="_DUPLICATE")
```

`AttemptDeletionFix` *Delete taxa to improve matrix condition*

Description

This will delete a taxon or taxa not involved as a source of hybrid gene flow or recipient of gene flow to see if this fixes a network that is not positive definite. It's drastic action.

Usage

```
AttemptDeletionFix(phy, flow, params=c(1,0,0.1, 0, 0), m.vector = c(1,2))
```

Arguments

`phy`	an object of class 'phylo'.
`flow`	the flow matrix
`params`	parameters to use for creating the modified variance covariance matrix
`m.vector`	how many taxa to try deleting.

Details

m.vector=c(1,2) means it will try deleting a single taxon at random; if it works, stop there, if not, replace that one and delete another, until all single taxon deletions have been tried. Then it'll repeat for all possible pairs of taxa. You can add on to this (all groups of three taxa, etc.) though the space of things to try gets very large.

Value

an object of class 'phylo'.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution in hybrid species on phylogenetic networks*. Submitted.

Examples

```
library(ape)
#simulate a tree of 5 taxa
phy<-rtree(5)
#check and adjust the phylogeny
AdjustForDet(phy,max.attempts=100)
```

BMhyb *Comparative method for studying hybridization using Brownian motion for trait evolution*

Description

This function fits the Brownian motion model of continuous character to investigate hybrid species through the hybrid vigor β, and the variation at the burst of hybridization v_H. Measurement error SE is also considered as well as the parameters including the over all mean μ and the overall variance σ^2 in the typical Brownian motion model.

Usage

```
BMhyb(data, phy, flow, opt.method="Nelder-Mead", models=c(1,2,3,4), verbose=TRUE,
     get.se=TRUE, plot.se=TRUE, store.sims=FALSE, precision=2, auto.adjust=FALSE,
      likelihood.precision=0.001, allow.extrapolation=FALSE, n.points=5000,
            measurement.error = 0, do.kappa.check=FALSE, number.of.proportions = 101,
         number.of.proportions.adaptive = 101, allow.restart=TRUE,
         lower.bounds = c(0, -Inf, 0.000001, 0, 0),
        upper.bounds=c(10,Inf,100,100,100),
          badval.if.not.positive.definite=TRUE, attempt.deletion.fix=FALSE,
      starting.values=NULL, n.random.start.points=5000,
      do.Brissette.correction=FALSE,
      do.Higham.correction=TRUE, do.DE.correction=FALSE)
```

Arguments

data	continuous trait data containing species information in vector format.
phy	a tree in phylo class.
flow	a structure of gene flow.
opt.method	the method for used for optimization. The default is Nelder-Mead.
models	the model used for analysis (see details).
verbose	a TRUE/FALSE argument to start optimization.
get.se	a TRUE/FALSE argument estimation for doing simulation to estimate parameter uncertainty (see details).

`plot.se` a TRUE/FALSE argument for output the uncertainty plot for the model (see details).

`store.sims` a TRUE/FALSE argument to record the the parameter estimates and relevant values.

`precision` a numeric value to present the cutoff at which the user thinks the estimates become unreliable due to ill conditioned matrix.

`auto.adjust` a TRUE/FALSE argument to adjust the the phylogeny.

`likelihood.precision`
a numerical value used for verifying the convergence of the estimation.

`allow.extrapolation`
a TRUE/FALSE argument. If TRUE, and the VCV matrix was ill-conditioned, would use splines to estimate its likelihood.

`n.points` How many simulations to do to estimate confidence interval.

`measurement.error`
Estimate or fixed (see details).

`do.kappa.check` Check for VCV matrix condition, approximate if poor. Leave FALSE unless you're cleverer than us.

`number.of.proportions`
Control how many points to use for extrapolation for ill-conditioned matrices.

`number.of.proportions.adaptive`
Control how many points to use for extrapolation for ill-conditioned matrices when calculating confidence (you may want to do fewer to speed things up).

`allow.restart` if TRUE, stop and go back to calling function if this finds a better value.

`lower.bounds` minimum values of all possible parameters (non-free ones will be deleted).

`upper.bounds` maximum values of all possible parameters (non-free ones will be deleted).

`badval.if.not.positive.definite`
check to see if the network is numerically well-behaved).

`attempt.deletion.fix`
if the network is not well-behaved, try deleting taxa until it is.

`starting.values`
user given starting values (perhaps from BMhybGrid)

`n.random.start.points`
number of points to use for a starting grid search

`do.Brissette.correction`
If VCV is not positive definite, do Brissette et al. (2007) correction.

`do.Higham.correction`
If VCV is not positive definite, do Higham (2002).

`do.DE.correction`
If VCV is not positive definite, find a nearby matrix that is.

Details

Function BMhyb fits likelihood models for continuous characters. As an input, **BMhyb** requires a phylogenetic tree of the phylo class, a structure of gene flow and a comparative data. Currently the method is developed for univariate analysis where the comparative data includes a single trait for analyses. The full likelihood model includes several parameters: the ancestral state μ, the overall rate of evolution σ^2, the hybrid vigor β, the hybrid burst variation at formation v_H and the measurement error SE. The struture of gene flow is a five column table where the first and the second column contain the donor and recipient information. The third column is the information about the heritability factor $gamma$ which is a fraction of the recipient trait that comes from the source. For example, if one thought that 10 The fifth column, recipient time, records time from the root of the recipient that counting forward from the root when the gene flow happened from the donor. For more details, please see our paper. Note that gene flow out of and into a species occurs at a single time in this model, but it doesn't have to be at the same time for donor and recipient: for example, the genes could flow into an unsampled species, then later get passed to a recipient.

The function allows some fixed values of parameters and treats others as free parameters: model 1 fixes β at 1 but allows v_H to vary; model 2 allows β to vary but fixes v_H at 0; model 3 fixes β at 1 and v_H at 0; and model 4 allows both to vary. BMhyb fits the model through maximum likelihood. When setting $get.se$ to TRUE, the BMhyb will estimate parameter uncertainty through a sampling procedure (see help for $AdaptiveConfidenceIntervalSampling$. If $plot.se$ is TRUE, a PDF will be generated showing these regions. Model averaged parameter estimates are calculated by the Akaike weight.

$measurement.error$ tells the program how to deal with measurement error. If set to NULL, it estimates a fixed one shared by all species. If you enter a single value (which could be zero if you measure your species really, really well), it uses this as fixed measurement error for all species. You can also enter a vector of measurement errors to allow species to have different ones; order should match that of taxa in vcv.phylo(phy). By default, it assumes zero measurement error.

Parameter units matter; by convention, they are not reported in comparative methods, but that's unfortunate. If your analysis is of species body length, let's say you measure it in meters. You should probably log transform the measurements to fit the assumptions of Brownian motion (equal chance of a change by 1 whether the initial size is 10 m or 0.001 m; values can go arbitrarily bigger or smaller, rather than being bounded by zero).

$data$: you should give this to the function in log(meters). I.e., if you measure sharks of length 1 m, 1.2 m, and 3 m, you should pass in 0.00, 0.18, and 1.10 as the traits. $measurement.error$: if you're not having the program calculate it, this should also be in units of log(meters). If you can measure shark length +/- 0.01 m, then this would be -4.61.

The program returns variables that also have units. Assuming your tree has branch lengths in millions of years: $sigma - squared$, the rate of evolution, is in (log(meters))^2 / MY mu, the ancestral state, is in log(meters) $beta$, the scalar, is unitless. A value of 1.3 suggests that species formed from hybridization are 1.3 times larger than parent species. Note this is true for traits in log space (so in this case, for example, we'd expect a hybrid coming from species 2 m long to be 2.6 m long), but if you're using non-log transformed data the meaning of beta is different. vH is the variance that comes from a hybridization event. In this case, (log(meters))^2. SE is the inferred measurement error. It's in log(meters) units.

If you wanted to compare things on the same scale, take into account the units. For example, you could compare the amount of variance for a species generated by one hybridization event:

$sigma - squared$ * tree height = variance coming from Brownian motion on the tree v_H = variance coming from the hybridization event SE^2 = variance coming from measurement error.

Though the model seems straightforward (Brownian motion on a basic network) there are numerical issues that can lead to difficulty calculating the likelihood (look up ill-conditioned matrices if you're curious). We have spent a lot of development time trying to address this issue and have various approximations in place to deal with this, including scaling the tree and trying transformations of the tree and then interpolation to get an estimate of the likelihood. There is a tradeoff. A cleaner approach would be to just return NA if the likelihood could not be calculated with high precision, and you can set the program to do this with some of the settings here ($allow.extrapolation$=FALSE, $auto.adjust$=FALSE, decreasing $precision$ (a threshold for the condition number (kappa) of the matrix)). However, the real risk is that your hard-won dataset just generates an NA (or the equivalent, a very bad fixed negative log likelihood value, set to be something like 1e307 on many computers). Our defaults let you get an approximate answer; pay heed to the confidence intervals, as well. However, we recommend a "sniff test" for your results: the negative log likelihood values for your various models will vary, but should be something like 68.2 for one model, 74.3 for another, etc. If you get one model that is orders of magnitude different in log likelihood, something is likely deeply wrong.

Value

A summarized table including the type of model, the corresponding number of parameter, the parameter estimates, the likelihood values, the upper bound and lower bound of the parameters, and the Akaike weights for model averaging.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution in hybrid species on phylogenetic networks*. Submitted.

Burnham, K.P., and D.R. Anderson. 2004. *Model selection and inference: a practical information-theoretic approach*. Sec. Ed. Springer, New York.

Examples

```
#set up the number of non hybrid
ntax.nonhybrid<-2
#set up the number of non hybrid
ntax.hybrid<-1
#simulate a network
network<-SimulateNetwork(ntax.nonhybrid=ntax.nonhybrid, ntax.hybrid=ntax.hybrid,
flow.proportion=0.5, origin.type='clade', birth = 1, death = 0.5, sample.f = 0.5,
tree.height = 1, allow.ghost=FALSE)
    #simulate the tips data
tips<-rnorm(ntax.nonhybrid+ntax.hybrid)
names(tips)<-paste("t",(1:(ntax.nonhybrid+ntax.hybrid)),sep="")
#run the analysis uses model 4 under optimization method (more than 5 seconds)
```

```
BMhyb(tips,network$phy,network$flow, opt.method="Nelder-Mead", models=4, verbose=TRUE,
get.se=FALSE, plot.se=FALSE, store.sims=FALSE, precision=2, auto.adjust=FALSE,
likelihood.precision=0.001, allow.extrapolation=TRUE)

  #run the analysis uses model 4 under Grid method
  result<-BMhybGrid(tips,network$phy,network$flow, models=4, verbose=FALSE,
          get.se=FALSE, plot.se=FALSE, store.sims=FALSE, precision=2,
          auto.adjust=FALSE,likelihood.precision=0.001,
          allow.extrapolation=FALSE, n.points=10,
          measurement.error = 0, do.kappa.check=FALSE,
          number.of.proportions = 101, number.of.proportions.adaptive = 101,
          allow.restart=FALSE, lower.bounds = c(0, -Inf, 0.000001, 0, 0),
          upper.bounds=c(10,Inf,100,100,100),
          badval.if.not.positive.definite=TRUE,
          attempt.deletion.fix=FALSE, starting.values=NULL,
          do.Brissette.correction=FALSE, do.Higham.correction=TRUE,
          do.DE.correction=FALSE)
  #>result
  #  Model   sigma.sq         mu       bt       vh SE      AICc  NegLogL K
  #     4 0.02950803 0.02499673 8.064703 1.127043  0 -6.570164 2.714918 4
  #  sigma.sq.lower sigma.sq.upper mu.lower mu.upper bt.lower bt.upper vh.lower
  #              NA             NA       NA       NA       NA       NA       NA
  #  vh.upper SE.lower SE.upper penalty deltaAICc AkaikeWeight
  #       NA       NA       NA       0         0            1
```

BMhybGrid *Comparative method for studying hybridization using Brownian motion for trait evolution*

Description

This function fits the Brownian motion model of continuous character to investigate hybrid species through the hybrid vigor β, and the variation at the burst of hybridization v_H. Measurement error SE is also considered as well as the parameters including the over all mean μ and the overall variance σ^2 in the typical Brownian motion model.

Usage

```
BMhybGrid(data, phy, flow, models=c(1,2,3,4), verbose=TRUE,
          get.se=TRUE, plot.se=TRUE, store.sims=TRUE, precision=2,
  auto.adjust=FALSE,likelihood.precision=0.001,
  allow.extrapolation=FALSE, n.points=5000,
  measurement.error = 0, do.kappa.check=FALSE,
  number.of.proportions = 101, number.of.proportions.adaptive = 101,
  allow.restart=TRUE, lower.bounds = c(0, -Inf, 0.000001, 0, 0),
  upper.bounds=c(10,Inf,100,100,100),
  badval.if.not.positive.definite=TRUE,
  attempt.deletion.fix=FALSE, starting.values=NULL,
```

```
  do.Brissette.correction=FALSE, do.Higham.correction=TRUE,
  do.DE.correction=FALSE)
```

Arguments

data	continuous trait data containing species information in vector format.
phy	a tree in phylo class.
flow	a structure of gene flow.
models	the model used for analysis (see details).
verbose	a TRUE/FALSE argument to start optimization.
get.se	a TRUE/FALSE argument estimation for doing simulation to estimate parameter uncertainty (see details).
plot.se	a TRUE/FALSE argument for output the uncertainty plot for the model (see details).
store.sims	a TRUE/FALSE argument to record the the parameter estimates and relevant values.
precision	a numeric value to present the cutoff at which the user thinks the estimates become unreliable due to ill conditioned matrix.
auto.adjust	a TRUE/FALSE argument to adjust the the phylogeny.
likelihood.precision	a numerical value used for verifying the convergence of the estimation.
allow.extrapolation	a TRUE/FALSE argument. If TRUE, and the VCV matrix was ill-conditioned, would use splines to estimate its likelihood.
n.points	How many simulations to do to estimate confidence interval.
measurement.error	Estimate or fixed (see details).
do.kappa.check	Check for VCV matrix condition, approximate if poor. Leave FALSE unless you're cleverer than us.
number.of.proportions	Control how many points to use for extrapolation for ill-conditioned matrices.
number.of.proportions.adaptive	Control how many points to use for extrapolation for ill-conditioned matrices when calculating confidence (you may want to do fewer to speed things up).
allow.restart	if TRUE, stop and go back to calling function if this finds a better value.
lower.bounds	minimum values of all possible parameters (non-free ones will be deleted).
upper.bounds	maximum values of all possible parameters (non-free ones will be deleted).
badval.if.not.positive.definite	check to see if the network is numerically well-behaved).
attempt.deletion.fix	if the network is not well-behaved, try deleting taxa until it is.

`starting.values`
: user given starting values (perhaps from BMhybGrid)

`do.Brissette.correction`
: If VCV is not positive definite, do Brissette et al. (2007) correction.

`do.Higham.correction`
: If VCV is not positive definite, do Higham (2002).

`do.DE.correction`
: If VCV is not positive definite, find a nearby matrix that is.

Details

Function BMhyb fits likelihood models for continuous characters. As an input, **BMhyb** requires a phylogenetic tree of the phylo class, a structure of gene flow and a comparative data. Currently the method is developed for univariate analysis where the comparative data includes a single trait for analyses. The full likelihood model includes several parameters: the ancestral state μ, the overall rate of evolution σ^2, the hybrid vigor β, the hybrid burst variation at formation v_H and the measurement error SE. The struture of gene flow is a five column table where the first and the second column contain the donor and recipient information. The third column is the information about the heritability factor $gamma$ which is a fraction of the recipient trait that comes from the source. For example, if one thought that 10 The fifth column, recipient time, records time from the root of the recipient that counting forward from the root when the gene flow happened from the donor. For more details, please see our paper. Note that gene flow out of and into a species occurs at a single time in this model, but it doesn't have to be at the same time for donor and recipient: for example, the genes could flow into an unsampled species, then later get passed to a recipient.

The function allows some fixed values of parameters and treats others as free parameters: model 1 fixes β at 1 but allows v_H to vary; model 2 allows β to vary but fixes v_H at 0; model 3 fixes β at 1 and v_H at 0; and model 4 allows both to vary. BMhyb fits the model through maximum likelihood. When setting $get.se$ to TRUE, the BMhyb will estimate parameter uncertainty through a sampling procedure (see help for $AdaptiveConfidenceIntervalSampling$. If $plot.se$ is TRUE, a PDF will be generated showing these regions. Model averaged parameter estimates are calculated by the Akaike weight.

$measurement.error$ tells the program how to deal with measurement error. If set to NULL, it estimates a fixed one shared by all species. If you enter a single value (which could be zero if you measure your species really, really well), it uses this as fixed measurement error for all species. You can also enter a vector of measurement errors to allow species to have different ones; order should match that of taxa in vcv.phylo(phy). By default, it assumes zero measurement error.

Parameter units matter; by convention, they are not reported in comparative methods, but that's unfortunate. If your analysis is of species body length, let's say you measure it in meters. You should probably log transform the measurements to fit the assumptions of Brownian motion (equal chance of a change by 1 whether the initial size is 10 m or 0.001 m; values can go arbitrarily bigger or smaller, rather than being bounded by zero).

$data$: you should give this to the function in log(meters). I.e., if you measure sharks of length 1 m, 1.2 m, and 3 m, you should pass in 0.00, 0.18, and 1.10 as the traits. $measurement.error$: if you're not having the program calculate it, this should also be in units of log(meters). If you can measure shark length +/- 0.01 m, then this would be -4.61.

The program returns variables that also have units. Assuming your tree has branch lengths in millions of years: $sigma - squared$, the rate of evolution, is in (log(meters))^2 / MY mu, the

ancestral state, is in log(meters) $beta$, the scalar, is unitless. A value of 1.3 suggests that species formed from hybridization are 1.3 times larger than parent species. Note this is true for traits in log space (so in this case, for example, we'd expect a hybrid coming from species 2 m long to be 2.6 m long), but if you're using non-log transformed data the meaning of beta is different. vH is the variance that comes from a hybridization event. In this case, (log(meters))^2. SE is the inferred measurement error. It's in log(meters) units.

If you wanted to compare things on the same scale, take into account the units. For example, you could compare the amount of variance for a species generated by one hybridization event: $sigma-squared$ * tree height = variance coming from Brownian motion on the tree v_H = variance coming from the hybridization event SE^2 = variance coming from measurement error.

Though the model seems straightforward (Brownian motion on a basic network) there are numerical issues that can lead to difficulty calculating the likelihood (look up ill-conditioned matrices if you're curious). We have spent a lot of development time trying to address this issue and have various approximations in place to deal with this, including scaling the tree and trying transformations of the tree and then interpolation to get an estimate of the likelihood. There is a tradeoff. A cleaner approach would be to just return NA if the likelihood could not be calculated with high precision, and you can set the program to do this with some of the settings here ($allow.extrapolation$=FALSE, $auto.adjust$=FALSE, decreasing $precision$ (a threshold for the condition number (kappa) of the matrix)). However, the real risk is that your hard-won dataset just generates an NA (or the equivalent, a very bad fixed negative log likelihood value, set to be something like 1e307 on many computers). Our defaults let you get an approximate answer; pay heed to the confidence intervals, as well. However, we recommend a "sniff test" for your results: the negative log likelihood values for your various models will vary, but should be something like 68.2 for one model, 74.3 for another, etc. If you get one model that is orders of magnitude different in log likelihood, something is likely deeply wrong.

Value

A summarized table including the type of model, the corresponding number of parameter, the parameter estimates, the likelihood values, the upper bound and lower bound of the parameters, and the Akaike weights for model averaging.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution in hybrid species on phylogenetic networks*. Submitted.

Burnham, K.P., and D.R. Anderson. 2004. *Model selection and inference: a practical information-theoretic approach*. Sec. Ed. Springer, New York.

Examples

```
#set up the number of non hybrid
ntax.nonhybrid<-2
#set up the number of non hybrid
ntax.hybrid<-1
```

```
#simulate a network
network<-SimulateNetwork(ntax.nonhybrid=ntax.nonhybrid, ntax.hybrid=ntax.hybrid,
flow.proportion=0.5, origin.type='clade', birth = 1, death = 0.5, sample.f = 0.5,
tree.height = 1, allow.ghost=FALSE)
    #simulate the tips data
tips<-rnorm(ntax.nonhybrid+ntax.hybrid)
names(tips)<-paste("t",(1:(ntax.nonhybrid+ntax.hybrid)),sep="")
#run the analysis uses model 3

BMhyb(tips,network$phy,network$flow, opt.method="Nelder-Mead", models=3, verbose=TRUE,
get.se=FALSE, plot.se=FALSE, store.sims=FALSE, precision=2, auto.adjust=FALSE,
likelihood.precision=0.001, allow.extrapolation=TRUE)
```

`BrissetteEtAlCorrection` *Transforming current variance covariance matrix to a positive definite matrix*

Description

Use method to search for a positive definite matrix.

Usage

```
BrissetteEtAlCorrection(V.modified, min.eigenvalue=1e-6, max.attempts=10)
```

Arguments

`V.modified`	a matrix
`min.eigenvalue`	minimum of eigenvalues of V.modified
`max.attempts`	maimum number of attempt for transformation

Details

The function replaces negative eigenvalues of the variance covariance matrix to a small positive value, and then transform it back to the matrix.

Value

a postive definite matrix.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Brissette F., Khalili M. Leconte R. 2007. *Efficient stochastic generation of mult-site synthetic precipitation data*. Journal of Hydrology 345:121-133.

`CalculateLikelihood` *Calculate the likelihood value for the model*

Description

This function calculates the likelihood value for the model.

Usage

```
CalculateLikelihood(x, data, phy, flow, actual.params, precision=2,
proportion.mix.with.diag=0, allow.extrapolation=FALSE,
measurement.error = NULL, do.kappa.check = FALSE, number.of.proportions=101,
lower.b=c(0, -Inf, 0.000001, 0, 0), upper.b=c(10,Inf,100,100,100),
badval.if.not.positive.definite=TRUE, do.Brissette.correction=FALSE,
do.Higham.correction=TRUE, do.DE.correction=FALSE, return.penalty=FALSE, ...)
```

Arguments

`x`	parameter of interest.
`data`	the trait values.
`phy`	a tree of phylo class.
`flow`	strcuture of gene flow.
`actual.params`	the free parameters for hybridization.
`precision`	a number to verify the condition of the variance covariation for the network model.
`proportion.mix.with.diag`	the proportion value that applies to the diagonal of the vcv matrix.
`allow.extrapolation`	a TRUE/FALSE argument.
`measurement.error`	estimate or fixed (see details).
`do.kappa.check`	check for VCV matrix condition, approximate if poor. Leave TRUE unless you're cleverer than us.
`number.of.proportions`	control how many points to use for extrapolation for ill-conditioned matrices.
`lower.b`	minimum values of all possible parameters (non-free ones will be deleted).
`upper.b`	maximum values of all possible parameters (non-free ones will be deleted).
`badval.if.not.positive.definite`	If the VCV is not positive definite, return a large bad value for likelihood.

`do.Brissette.correction`
: If VCV is not positive definite, do Brissette et al. (2007) correction.

`do.Higham.correction`
: If VCV is not positive definite, do Higham (2002).

`do.DE.correction`
: If VCV is not positive definite, find a nearby matrix that is.

`return.penalty`
: If TRUE, return a penalty for approximating not the likelihood itself.

`...`
: place to absorb other arguments.

Details

This function calculates the likelihood value. As described in the argument, the x contains the parameter of interests including over all mean μ, the rate of evolution σ^2, the measurement error SE and the hybrid vigor β and the variation at the burst of hybridization v_H. Names of the taxa in data vector and the phylogeny must be match for further analysis, otherwise it would terminate immediately. The negative log likelihood function is calculated and a reasonable likelihood value is returned by checking the variance covariance matrix where a precision number is used for verifying the log conditioned number of the variance covariance of the network model. When the matrix is ill conditioned, we modestly adjust the matrix by the shrink the value of the off diagonal matrix using a proportion array. Then the likelihood is calculated using a spline approximation.

Value

negative log likelihood value.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution on phylogenetic networks*. Submitted.

Examples

```
#set the number of hybrid
ntax.nonhybrid<-5
#set the number of hybrid
ntax.hybrid<-1
#simulate a network
network<-SimulateNetwork(ntax.nonhybrid=ntax.nonhybrid, ntax.hybrid=ntax.hybrid,
flow.proportion=0.5, origin.type='clade', birth = 1, death = 0.5, sample.f = 0.5,
tree.height = 1, allow.ghost=FALSE)
sigma.sq <- 0.01;mu <- 1;SE <- 0
#simulatedata
data<-rnorm(ntax.nonhybrid+ntax.hybrid)
names(data)<-paste("t",(1:(ntax.nonhybrid+ntax.hybrid)),sep="")
#calculate the likelihood value
CalculateLikelihood(c(sigma.sq,mu,SE), data, network$phy, network$flow, "vh",
```

```
precision=2, proportion.mix.with.diag=0, allow.extrapolation=TRUE)
```

`cichlid` *Cichlid data from Kobmuller et al. 2007*

Description

This dataset contains the phylogeny, structure of gene flow and comparative data for extant species from Kobmuller et al. 2007.

Usage

```
cichlid
```

Format

a list with the phylogeny (phy), gene flow (flow), and comparative data (data) for cichlid species.

Source

Kobmuller et al. 2007

References

Kobmuller, S., N. Duftner, K. M. Sefc, M. Aibara,M. Stipacek, M. Blanc, B. Egger, and C. Sturmbauer. 2007.Reticulate phylogeny of gastropod-shell-breeding cichlids from Lake Tanganyika: the result of repeated introgressive hybridization. BMC Evolutionary Biology 7:7.

`ContourFromAdaptiveSampling`
Contour maps

Description

This function generates the contour maps for a pair of parameters of interest.

Usage

```
ContourFromAdaptiveSampling(sims, params.of.interest=NULL)
```

Arguments

`sims`	a data frame with the simulation results.
`params.of.interest`	name of parameter that is of interest to be plotted.

Details

This function generates the contour plots for the parameter of interest. It uses the simulated data from generating the confidence interval under adaptive sampling method where the likelihood value and parameters values are stored. Then it uses the function `PlotConvexHull` to generate the contour maps.

Value

It returns the plot of the contour map for a pair of parameters.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution on phylogenetic networks*. Submitted.

Examples

```
#simulate a data frame for two parameters
sims<-matrix(rnorm(300),ncol=3)
sims<-as.data.frame(sims)
colnames(sims)<-c("neglnL","param1","param2")
#plot the contour map for the parameters
ContourFromAdaptiveSampling(sims, params.of.interest=NULL)
```

ConvertExpm1 *Exponentials*

Description

ConvertExpm1 computes exp(x) - 1.

Usage

```
ConvertExpm1(x)
```

Arguments

x	x: a numeric or complex vector

Details

see function expm1 in for more detail descrition.

Value

A vector of the same length as 'x' containing the transformed value.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Becker, R.A., Chambers, J.M. and Wilks, A.R. 1988. *The New S Language*. Wadsworth and Brooks Cole.

`ConvertLog1P` *Logarithms*

Description

ConvertLog1P computes log(1+x) accurately also for |x| « 1.

Usage

```
ConvertLog1P(x)
```

Arguments

`x`	x: a numeric or complex vector

Details

see function log1p(x) in for more detail description.

Value

A vector of the same length as 'x' containing the transformed value.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Becker, R.A., Chambers, J.M. and Wilks, A.R. 1988. *The New S Language*. Wadsworth and Brooks Cole.

ConvertVectorToMatrix *Convert a vector into a matrix*

Description

The function outputs a matrix given a vector as its elements in the upper triangular part.

Usage

```
ConvertVectorToMatrix(x)
```

Arguments

x	x: a postive integer that indicates the number of elements in the upper triangular matrix.

Details

The length of vector equal the number of elements in a squared upper triangular matrix. It assigns value in the vector to the matrix accordingly. Eventually, the output of the matrix will be the matrix with lower triangular part equal to the upper triangular part.

Value

A square matrix

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

DetPass *determinant of the matrix*

Description

Calculate the determinant of the matrix and check whether it is positive.

Usage

```
DetPass(phy)
```

Arguments

phy	an object of class 'phylo'.

Details

This function first calculates variance covariance matrix C for the tree. It then calculates the determinant of the C, 0.0001*C, and 1000*C and verifies whether all the three determinants are positive and finite. If yes, a TRUE value is return which indicates that the C matrix is good for further use.

Value

a TRUE/FALSE value.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution on phylogenetic networks*. Submitted.

Examples

```
library(ape)
##simulate a tree of 5 taxa
phy<-rtree(5)
#check whether the determiant is positive
DetPass(phy)
```

GenerateRandomPositiveDefiniteMatrix

Generate positive definite matrix

Description

Generate postive definite matrix using QR decomposition

Usage

```
GenerateRandomPositiveDefiniteMatrix(n, ev = runif(n, 0, 10))
```

Arguments

n	size of matrix
ev	positive number

Details

Use QR decomposition to generate positive definite matrix. See reference.

Value

A positive definite matrix

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Ravi Varadhan, http://r.789695.n4.nabble.com/how-to-randomly-generate-a-n-by-n-positive-definite-matrix-in-R-td846858.html

---

`GenerateRandomValues` *Generate random values*

Description

Generate random value for parameters

Usage

```
GenerateRandomValues(data, free.parameters, lower, upper)
```

Arguments

`data`	numerical vector that contains trait data
`free.parameters`	TRUE/FALSE argument for free parameters
`lower`	lower bound for the new values
`upper`	upper bound for the new values

Details

The function generates random values for parameter estimate. The bound for each paramter is set up using exponential and uniform distribution accordingly.

Value

A vector contains the generate values for parameters.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

`GenerateValues` *Generate parameter values*

Description

This function generates new parameter values in a range.

Usage

```
GenerateValues(par, lower, upper, max.tries=100, expand.prob=0, examined.max,
examined.min)
```

Arguments

`par`	parameters of interest.
`lower`	the actual lower bound for the parameters.
`upper`	the actual upper bound for the parameters.
`max.tries`	maximum number of attempts to generate the parameter value. The default is set to 100.
`expand.prob`	a probability value to expand.
`examined.max`	the allowed maximum for the parameters values.
`examined.min`	the allowed minimum for the parameters values

Details

This function generates new parameter values using the uniform distribution with the interval(L, U) where for each parameter of interest, the lower bound of the interval is set to $L = \max(lower, 0.9 * examined.min)$ and the upper bound of the interval is set to $U = \min(upper, 1.1 * examined.max)$.

Value

The generated parameter values.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution in hybrid species on phylogenetic networks*. Submitted.

Seber G.A.F., Salehi M.M. 2013. Adaptive sampling designs: Inference for spatial and clustered population. Springer.

Examples

```
#parameter of interests
mu<-1;sigma.sq<-0.1;bt<-1;vH<-0;SE<-0
#parameters of interest
par<-c(mu,sigma.sq,bt,vH,SE)
#the lower bound
lower=c(-2,0,-10,0,0)
#the upper bound
upper<-c(2,1,10,100,100)
#the examined minimum value
examined.min<-c(-1.8,0.001,-9.8,0,0)
#the examined maximum value
examined.max<-c(1.90,1.1,10.2,100,100)
#simulate points
sim.points<-GenerateValues(par, lower, upper, max.tries=100, expand.prob=0,
examined.max, examined.min)
names(sim.points)<-c("mu","sigma.sq","bt","vH","SE")
#show result
sim.points
```

GetAncestor *Identify the ancestor*

Description

Given the descendant node for the tree, the function will return its ancestor node.

Usage

```
GetAncestor(phy, node)
```

Arguments

phy	an object of class 'phylo'.
node	descedant node.

Value

the ancestor node.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution on phylogenetic networks*. Submitted.

Examples

```
library(ape)
#simulate a tree of 3 taxa
  phy<-rtree(3)
  #plot the tree
  plot(phy)
  #descedant node
  node<-1
  #get the ancestor node, it will return 5.
  GetAncestor(phy,node)
```

`GetClade` *Get clade from the tree*

Description

Search and report the nodes of the tree that have the desired clade size.

Usage

```
GetClade(phy, clade.size)
```

Arguments

`phy`	an object of class 'phylo'.
`clade.size`	size of the clade.

Details

This function uses the 'phylo' class where the edges(ancestor-descendant relationship) of the tree are used to identify the interior node with desired number of descendent (clade.size) on the tips. It applies the function `findMRCA` in *phytools* to search the ancestor.

Value

the interior node that has descendants of size clade.size on the tip.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2014. *Studying trait evolution in hybrid species on phylogenetic networks*. Submitted.

Examples

```
library(ape)
library(phytools)
#simulate a tree
  phy<-rtree(3)
  #plot the tree
  plot(phy)
  #set up the clade size
  clade.size<-2
  #search the nodes that has desired clade size
  GetClade(phy,clade.size) #node 5 will be return
```

`GetMeansModified` *The species means for the network model*

Description

This function returns the species mean for the network model. For the non hybrid species, it returns as a parameter μ and for hybrid species it returns $\mu + \log \beta$.

Usage

```
GetMeansModified(x, phy, flow, actual.params)
```

Arguments

`x`	vector of the parameters: including the rate of evolution σ^2, overall mean μ, measurement error SE, hybrid vigor β, variation at the burst of hybridization v_H.
`phy`	a class of phylo tree.
`flow`	a flow struture of gene flow.
`actual.params`	The parameters that are related to the hybridization including the hybrid vigor β and variation at the burst of hybridization v_H.

Value

the species mean.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2015. *Study trait evolution on phylogenetic networks*. Submitted.

Examples

```
#number of nonhybrid species
ntax.nonhybrid<-5
#number of hybrid species
ntax.hybrid<-2
#simulate network
network<-SimulateNetwork(ntax.nonhybrid=ntax.nonhybrid, ntax.hybrid=ntax.hybrid,
flow.proportion=0.5, origin.type='clade', birth = 1, death = 0.5, sample.f = 0.5,
tree.height = 1, allow.ghost=FALSE)
    #set the parameter values
    sigma.sq <- 0.01;mu <- 1;SE <- 0;bt<-12;vh<-0
    par<-c(sigma.sq,mu,SE,bt,vh)
    names(par)<-c("sigma.sq","mu","SE","bt","vh")
    actual.params<-12
    names(actual.params)<-"bt"
#calculate the species means
GetMeansModified(par, network$phy, network$flow,actual.params)
```

GetVModified *Variance covariance matrix for the network model*

Description

A function returns the variance covariance function for the network model under the Brownian motion for trait evolution.

Usage

```
GetVModified(x, phy, flow, actual.params, measurement.error = NULL)
```

Arguments

x	vector of the parameters: including rate of evolution σ^2, overall mean μ, measurement error SE, hybrid vigor β, and variation at the burst of hybridization v_H.
phy	a class of phylo tree.
flow	a struture of gene flow.
actual.params	The parameters that are related to the hybridization including the hybrid vigor β and variation at the burst of hybridization v_H.
measurement.error	estimate or fixed (see details).

Details

This function calculates the variance covariance matrix under the network model. The function first read tree in *phylo* class and convert the variance covariance matrix use the Brownian motion model. It then uses the structure of the gene flow to identify the hybrid species, parents and the direction of the flow. The covariance of between the hybrid and non hybrid species is the weighted sum of the covariance from evolution along the tree plus evolution along the migration path. For more detail about the model description, please see Jhwueng and O'Meara 2016.

Value

the variance covariance matrix of size equal to the number of species on the tip of the phylogeny.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Study trait evolution on phylogenetic networks*. Submitted.

Examples

```
#set the number of non hybrid species
ntax.nonhybrid<-4
#set the number of hybrid species
ntax.hybrid<-1
#simulate a network
network<-SimulateNetwork(ntax.nonhybrid=ntax.nonhybrid, ntax.hybrid=ntax.hybrid,
flow.proportion=0.5, origin.type='clade', birth = 1, death = 0.5, sample.f = 0.5,
tree.height = 1, allow.ghost=FALSE)
#assign the parameter values
sigma.sq <- 0.01;mu <- 1;SE <- 0
#get the variance covariance matrix
GetVModified(c(sigma.sq,mu,SE), network$phy, network$flow, "bt")
```

IsPositiveDefinite *Check to make sure a variance covariance network (from a tree or network) is positive definite*

Description

Return TRUE if it is positive definite, FALSE otherwise

Usage

```
IsPositiveDefinite(V.modified)
```

Arguments

V.modified	a matrix

Details

Calculates the eigen values from a matrix, sees if the minimum is greater than zero

Value

a TRUE/FALSE value.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Studying trait evolution on phylogenetic networks*. Submitted.

LumpIntoClades *Lump into clades*

Description

This function Lump the hybrid species into a clades.

Usage

```
LumpIntoClades(phy,flow)
```

Arguments

phy	a phylogeny of phylo class.
flow	structure of gene flow.

Details

The function summarizes the flow structure into the clade where the donor clades contribute more than one recipient clades.

Value

a flow table that contains the donor clade, recipient clade, proportion of the gene flow,time from root donor, and time from root recipient.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2015. *Trait evolution on phylogenetic networks*. Submitted.

Examples

```
#set the number of non hybrid species
ntax.nonhybrid<-5
#set the number of hybrid species
ntax.hybrid<-2
#simulate a network
network<-SimulateNetwork(ntax.nonhybrid=ntax.nonhybrid, ntax.hybrid=ntax.hybrid,
flow.proportion=0.5, origin.type='clade', birth = 1, death = 0.5, sample.f = 0.5,
tree.height = 1, allow.ghost=FALSE)
#show the flow
network$flow
#show the new flow table
LumpIntoClades(network$phy, network$flow)
```

---

nicotiana *nicotiana data*

Description

The phylogeny, gene flow, and comparative data for nicotiana collected in literature.

Usage

```
nicotiana
```

Format

a list with the phylogeny (phy), gene flow (flow), and comparative data (data) for Nicotiana species.

Source

Chase et al. (2003), Clarkson et al. (2005), Komori et al. (2000)

References

Chase M.W., Knapp S., Cox A.V., Clarkson J.J., Butsko Y., Joseph J., Savolainen V., and Parokonny A.S. 2003. Molecular systematics, GISH and the origin of hybrid taxa in Nicotiana(Solanaceae). Annals of Botany 92: 107-127.

Clarkson J.J., Lim K.Y., Kovarik A., Chase M.W., Knapp S. and Leitch A.R. 2005. Long-term genome diploidization I allopolyploid Nicotiana section Repandae(Solanaceae). New Phytologist 168:241-252.

Komori T., Myers P.N., Yamada S., Kubo T., and Imaseki H. 2000. Comparative study of the Nicotiana species with respect to water deficit tolerance during early growth. Euphytica 116:121-130.

`PlotAICRegion` *Plot the region from which points were estimated*

Description

Plots all points evaluated.

Usage

```
PlotAICRegion(sims, show.finite.only=TRUE, true.params=NULL,  ...)
```

Arguments

`sims`	sims object from BMhybGrid.
`show.finite.only`	If true, only show where there were finite likelihoods; if false, show colors for these too
`true.params`	if known the true values used in a simulation
`...`	other arguments

Details

This plots the results of a grid search. By default, it shows the regions where there are finite values without distinguishing the best region (future work will add color and contour lines). A red diamond shows the best value found; a blue asterisk shows the true value (if known). Additional arguments to plot can be passed.

Value

plot of the space.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Trait evolution on phylogenetic networks*. Submitted.

`PlotConvexHull` *Convex hull*

Description

Plotting the convex hall for given 2D datasets.

Usage

```
PlotConvexHull(xcoord, ycoord, lcolor)
```

Arguments

`xcoord`	coordinate for the horizontal axis.
`ycoord`	coordinate for the vertical axis.
`lcolor`	color of the convect hull.

Details

This function uses the base function *chull()*, which takes x,y coordinates of data and outputs a vector(hpts variable) of integers that indicate the points in dataset that comprise the convex hull around data. It then uses the base function lines to draw the convex hull using the x,y coordinates stored in hpts.

Value

The convex hull plot for a set of 2D points.

Author(s)

Ken Takagi

References

https://chitchatr.wordpress.com/2011/12/30/convex-hull-around-scatter-plot-in-r/

Examples

```
# Create a set of random data to plot convex hull around
x<-rnorm(100,0.8,0.3)
y<- rnorm(100,0.8,0.3)
#get max and min of all x and y data for nice plotting
xrange<-range(x)
yrange<-range(y)
#plot it up!
plot(x,y,type="p",pch=1,col='black',xlim=c(xrange),ylim=c(yrange))
PlotConvexHull(xcoord=x,ycoord=y,lcolor='red')
```

PlotNetwork *Phylogenetic Network Plot*

Description

Plots the phylogenetic network: including the phylogenetic tree and arrows that indicate the direction of gene flows.

Usage

```
PlotNetwork(phy, flow, col.non="black", col.hybrid="red", col.donor="blue",
name.padding=1.5, cex=1, xlab="", bty="n", head.length=0.2, edge.width=2,
col.tree="darkgray", col.arrow="red", arrow.width=1, try.rotations=FALSE, ...)
```

Arguments

phy	a class phylo tree object.
flow	a flow structure of gene flow.
col.non	color of non hybrid taxa.
col.hybrid	color of hybrid taxa.
col.donor	color of the donor.
name.padding	the size that allowed for the plot in the x axis.
cex	number indicating the amount by which plotting text and symbols should be scaled relative to the default. 1=default, 1.5 is 50 percent larger, 0.5 is 50 percent smaller, etc.
xlab	X axis label using font and character expansion par("font.lab") and color par("col.lab").
bty	the type of box to be drawn around the legend. The allowed values are "o" (the default) and "n".
head.length	length of the head.
edge.width	width of the edges.
col.tree	color of the tree.

col.arrow	color of gene flow arrow.
arrow.width	width of the arrow.
try.rotations	Each time try randomly rotating nodes of the tree.
...	other arguments

Details

This function generates plots for network models. It first draws the phylogenetic tree for the given species. Then uses the gene flow data to draw the arrow from the donor species to recipient species. The hybrid taxa are shown in $col.hybrid$ with the gene flow direction from the donor species (colored in $col.donor$).

Value

plot of the network that involves the phylogeny and the gene flow direction.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Trait evolution on phylogenetic networks*. Submitted.

Examples

```
#set the number of non hybrid species
ntax.nonhybrid<-10
#set the number of hybrid species
ntax.hybrid<-3
#simulate the network with desired species
network<-SimulateNetwork(ntax.nonhybrid=ntax.nonhybrid, ntax.hybrid=ntax.hybrid,
flow.proportion=0.5, origin.type='clade', birth = 1, death = 0.5, sample.f = 0.5,
tree.height = 1, allow.ghost=FALSE)
#plot the network
PlotNetwork(network$phy,network$flow)
```

PositiveDefiniteOptimizationFn

Distance function for optimization

Description

Distance Function between two matrices for optimization searching of positive definite matrix.

Usage

```
PositiveDefiniteOptimizationFn(x, original)
```

Arguments

x	x: a vector that contains elements in the upper triangular part of the adjusted variance covariance matrix.
original	original: a vector that contains elements in the upper triangular part of the original variance covariance matrix.

Details

It returns a distance between the new matrix and the original matrix. If the new matrix contains negative element or is not positive definite, then it will return more penalty for the distance.

Value

a numerica value of distance

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

SimulateNetwork *Simulate phylogenetic network*

Description

Simulate network with desired number of taxa and number of hybrids.

Usage

```
SimulateNetwork(ntax.nonhybrid=100, ntax.hybrid=10, flow.proportion=0.5,
origin.type=c("clade", "individual"), birth = 1, death = 1, sample.f = 0.5,
tree.height = 1, allow.ghost=FALSE)
```

Arguments

ntax.nonhybrid	number of non hybrid taxa.
ntax.hybrid	number of hybrid taxa.
flow.proportion	the gene flow proportion from the parent.
origin.type	the type where the hybrids were formed. clade: hybrid formed and then speciate; individual: hybrid species formed and evolved without further speciation.

birth	birth rate for the tree speciation.
death	death rate (extinction) of the tree.
sample.f	sampling frequency.
tree.height	the height of the tree.
allow.ghost	allows ghost lineage that persists for a while, donates genes to a hybrid, and eventually goes extinct (or at least unsampled). Otherwise, hybridization event will be between coeval species with extant descendants.

Details

This function generates a tree with relevant hybridization information (the structure of gene flow information). Gene flow cannot go back in time. In general, gene flow can go forward in time via ghost lineages. If no ghost lineages are allowed, then there must be temporal overlap between the donor and recipient lineages.

Value

a list that contains

phy	a birth death tree with number of taxa of ntax.nonhybrid + ntax.hybrid
flow	The flow structure of the gene flow

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2016. *Trait evolution on phylogenetic networks*. Submitted.

Examples

```
library(TreeSim)
#set up the number for non hybrid species
non.hybrid <- 3
#set up the number for hybrid species
hybrid <- 1
#set up the gene flow proportion
flow<- 0.5
#set up the hybridization type to original
origins<- "individual"
#start to simulate the network
network<-SimulateNetwork(ntax.nonhybrid=non.hybrid, ntax.hybrid=hybrid,
flow.proportion=flow, origin.type="individual", birth = 1, death = 0.5,
sample.f = 0.5, tree.height = 1, allow.ghost=FALSE)
#print out the result
(network)
```

`SimulateTipData` *Simulate data on a network*

Description

Simulate comparative data under the given a network and relevant parameters.

Usage

```
SimulateTipData(phy, flow, params, suffix="_DUPLICATE")
```

Arguments

`phy`	a phylo object.
`flow`	data.frame of the flow parameters.
`params`	named vector of BMhyb parameter values.
`suffix`	taxa are duplicated on the network; this suffix distinguishes them.

Details

This function simulates tip data on a network, under a BMhyb model.

Value

a named vector of tip values.

Author(s)

Brian O'Meara, Dwueng-Chwuan Jhwueng.

References

Jhwueng D.C. and O'Meara B.C. 2015. *Trait evolution on phylogenetic networks*. Submitted.

Examples

```
library(TreeSim)
#set up the number for non hybrid species
non.hybrid <- 3
#set up the number for hybrid species
hybrid <- 1
#set up the gene flow proportion
flow<- 0.5
#set up the hybridization type to original
origins<- "individual"
#start to simulate the network
network<-SimulateNetwork(ntax.nonhybrid=non.hybrid, ntax.hybrid=hybrid,
```

```
flow.proportion=flow, origin.type="individual", birth = 1, death = 0.5,
sample.f = 0.5, tree.height = 1, allow.ghost=FALSE)
parameters <- c(0.01,1,1,0,0)
names(parameters) <- c("sigma.sq", "mu", "bt", "vh", "SE")
tip.data <- SimulateTipData(network$phy, network$flow, parameters)
print(tip.data)
```

Index