Package 'OUwie'

July 2, 2026

Version 3.0.2

Date 2026-06-26

Title Analysis of Evolutionary Rates in an OU Framework

Depends R (>= 3.2.0), ape, corpcor, nloptr, geiger, RColorBrewer

Suggests testthat, knitr, rmarkdown

Imports igraph, numDeriv, phytools, paleotree, phangorn, stats, lhs, interp, grDevices, parallel, phylolm, GenSA, MASS, corHMM, data.table, expm, ggplot2, reshape2

Description Estimates rates for continuous character evolution under Brownian motion and Ornstein-Uhlenbeck based Hansen models that allow both the strength of the pull and stochastic motion to vary across selective regimes. Beaulieu et al. (2012).

URL https://github.com/thej022214/OUwie

License GPL (>= 2)

VignetteBuilder knitr

RoxygenNote 7.3.3

NeedsCompilation no

Author Jeremy M. Beaulieu [aut, cre], Brian O'Meara [aut]

Maintainer Jeremy M. Beaulieu <jmbeauli@uark.edu>

Repository CRAN

Date/Publication 2026-07-02 01:10:03 UTC

Contents

`check.identify` *A test of regime identifiability*

Description

Ho and Ane test for determining whether all regimes form connected subtrees, making both the ancestral state and the regime optima unidentifiable.

Usage

```
check.identify(phy, data, simmap.tree=FALSE, quiet=FALSE)
```

Arguments

`phy`	a phylogenetic tree, in `ape` "phylo" format and with internal nodes labeled denoting the ancestral selective regimes.
`data`	a data.frame containing species information (see Details).
`simmap.tree`	a logical indicating whether the input tree is in SIMMAP format. The default is `FALSE`.
`quiet`	a logical indicating whether messages should be written to the screen. The default is `FALSE`.

Value

This returns a vector with two elements, with the first being an indicator of identifiability (0=unidentifiable, 1=identifiable, and if `get.penalty=TRUE`, the second is the penalty used for the modified BIC.

Author(s)

Jeremy M. Beaulieu and Brian C. O'Meara

References

Ho, L.S.T., and C. Ane. 2014. Intrinsic inference difficulties for trait evolution with Ornstein-Uhlenbeck models. Methods in Ecology and Evolution, 5: 1133-1146.

Example *An example dataset*

Description

An example dataset containing a 64-tip birth-death tree with internal node labels denoting two selective regimes, and a trait file in the proper format: 1) Genus_species, 2) current selective regime, 3) continuous trait data.

Format

a tree of class "phylo" and a data frame with 3 columns and 64 rows

fix.kappa *Adjust tree for matrix condition*

Description

Iteratively deletes taxa with shortest tip length to try to get a variance covariance matrix with good matrix condition.

Usage

```
fix.kappa(phy, data, threshold = log(40))
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format and with internal nodes labeled denoting the ancestral selective regimes.
data	a data.frame containing species information (see Details).
threshold	log(condition), as measured by kappa(), which is too large

Details

Internally, OUwie uses an algorithm that can perform poorly when the variance covariance matrix is poorly conditioned (which can happen if two columns are very similar, as when the divergence depth of two species is very recent). This does not mean there is anything wrong with the biology, just that the numerical algorithms perform poorly in that case. If it's a model that can be fit in phylolm or geiger, those packages use a differnt algorithm that is more robust to this. What this function does is take your original tree and data and deletes taxa with the shortest branches, in order, to try to get a starting tree with generally good condition. Deleting data is always a sad thing, but this can result in a more accurate estimate of the likelihood and parameter values.

Value

This returns a list with two elements:

`$phy`	the phylogeny with taxa deleted.
`$data`	the data with taxa deleted.

Author(s)

Brian C. O'Meara

`OUwie` *Generalized Hansen models*

Description

Fits generalized Ornstein-Uhlenbeck-based Hansen models of continuous characters evolving under discrete selective regimes.

Usage

```
OUwie(phy, data, model=c("BM1","BMS","OU1","OUM","OUMV","OUMA","OUMVA",
"TrendyM","TrendyMS"), simmap.tree=FALSE, root.age=NULL,scaleHeight=FALSE,
root.station=FALSE, get.root.theta=FALSE, shift.point=0.5, clade=NULL, tip.fog="none",
starting.vals=NULL, check.identify=TRUE, algorithm=c("invert", "three.point"),
diagn=FALSE, quiet=FALSE, warn=TRUE, lb = NULL, ub = NULL, opts = list(algorithm =
"NLOPT_LN_SBPLX", maxeval = "1000", ftol_rel = .Machine$double.eps^0.5), revert.old=FALSE)
```

Arguments

`phy`	a phylogenetic tree, in ape "phylo" format and with internal nodes labeled denoting the ancestral selective regimes.
`data`	a data.frame containing species information (see Details).
`model`	models to fit to comparative data (see Details).
`simmap.tree`	a logical indicating whether the input tree is in SIMMAP format. The default is `FALSE`.
`root.age`	indicates the age of the tree. This is to be used in cases where the "tips" are not contemporary, such as in cases for fossil trees. Default is `NULL` meaning latest tip is modern day.
`scaleHeight`	a logical indicating whether the total tree height should be scaled to 1 (see Details). The default is `FALSE`.
`root.station`	a logical indicating whether to assume a random starting point (TRUE) or a fixed starting point (FALSE) (see Details).
`get.root.theta`	a logical indicating whether the starting state, θ_0, should be estimated (see Details).

shift.point	the point along a branch where a regime change is assumed to have occurred (if SIMMAP=FALSE. The default is set to 0.5, or halfway along a branch.
clade	a list containing a pair of taxa whose MRCA is the clade of interest (see Details).
tip.fog	designates whether a fourth column in the data matrix contains within species variation for each species value ("known"). The tip fog is assumed to be the standard error of the species mean. The default is "none", but can be estimated by setting the option to "estimate".
starting.vals	a vector of initial values for the optimization search. For OU models, two must be supplied, with the first being the initial alpha value and the second being the initial sigma squared. For BM models, just a single value is needed.
check.identify	a logical indicating whether to check that the user-supplied regime paintings will produce identifiable theta estimates. The default is TRUE.
algorithm	designates whether the standard matrix inversion ('invert') or the faster 'three-point' algorithm of Ho and Ane (2013) should be used.
diagn	a logical indicating whether the full diagnostic analysis should be carried out. The default is FALSE.
quiet	a logical indicating whether progress should be written to the screen. The default is FALSE.
warn	a logical indicating whether a warning should be printed if the number of parameters exceeds ntips/10. The default is TRUE.
lb	if algorithm == "invert" a single value indicating the lower bound for the parameter values. if algorithm == "three.point", a vector of length 3 with position 1 as alpha lower bound, position 2 as sigma.sq's lower bound, position 3 as theta's lower bound. when set to NULL it will be a default value of 1e-9. Note that even if the model you're using doesn't include alpha (e.g. BM1), it must be included in this vector, but it will not be used to set any bounds.
ub	if algorithm == "invert" a single value indicating the upper bound for the parameter values. if algorithm == "three.point", a vector of length 3 with position 1 as alpha upper bound, position 2 as sigma.sq's upper bound, position 3 as theta's upper bound. when set to NULL it will be a default value of 100. Note that even if the model you're using doesn't include alpha (e.g. BM1), it must be included in this vector, but it will not be used to set any bounds.
opts	a list of options to pass to nloptr for the optimization: useful to adjust for faster, coarser searches
revert.old	a logical indicating whether or not to use an older implementation of the multiple alpha models. The default is "FALSE".

Details

This function fits various likelihood models for continuous characters evolving under discrete selective regimes. The function returns parameter estimates and their approximate standard errors. The R package nloptr provides a common interface to NLopt, an open-source library for nonlinear optimization. The likelihood function is maximized using the bounded subplex optimization routine (NLOPT_LN_SBPLX). As input all OUwie requires is a tree and a trait data.frame. The tree must be of class "phylo" and must contain the ancestral selective regimes as internal node labels. Internal

node labels can be applied manually or from some sort of ancestral state reconstruction procedure (BayesTraits, ape, diversitree, SIMMAP, etc.), which would then be brought into OUwie. This is essentially what is required by ouch and Brownie (though Brownie provides built-in ancestral state reconstruction capabilities). The trait data.frame must have column entries in the following order: [,1] species names, [,2] current selective regime, and [,3] the continuous trait of interest. Alternatively, if the user wants to incorporate tip.fog (`tip.fog="known"`), then a fourth column, [,4] must be included that provides the standard error estimates for each species mean. However, a global tip.fog for all taxa can be estimated from the data (`tip.fog="estimate"`). Also, a user can specify a particular clade as being in a different selective regime, by specifying a pair of species whose mrca is the root of the clade of interest [e.g., `clade=c("taxaA","taxaB")`]. OUwie will automatically assign internal node labels and update the data matrix according to this clade designation.

Possible models are as follows: single-rate Brownian motion (`model=BM1`), Brownian motion with different rate parameters for each state on a tree (`model=BMS`), Ornstein-Uhlenbeck model with a single optimum for all species (`model=OU1`), Ornstein-Uhlenbeck model with different state means and a single α and σ^2 acting all selective regimes (`model=OUM`), and new Ornstein-Uhlenbeck models that assume different state means as well as either multiple σ^2 (`model=OUMV`), multiple α (`model=OUMA`), or multiple α and σ^2 per selective regime (`model=OUMVA`).

Note (Sept. 4): From feedback from a user, we have concerns about the likelihood function for the OUMA and OUMVA models. As of Sept. 4, 2025, we recommend not using these models for now, but we expect the situation to be resolved soon (either with corrected likelihood functions or a definite proof that they are ok as is). That said, we leave these as options for reproducibility and debugging. If you do use them, be sure to note the package version and report this in your work.

By default, we drop the root optima and absorb the weight into whatever regime the root is in. In previous version we used to incorrectly refer to this as "stationarity". True stationarity assumes that the starting state comes from a distribution, and the covariance requires an additional variance term to account for the fact that, up until T=0, the lineage is assumed to have been evolving in the ancestral regime. We have added this in for the OU1 and OUM models only (root.station=TRUE).

Note, too, that when specifying the BMS model also be mindful of the root.station flag. When root.station=FALSE, the non-censored model of O'Meara et al. 2006 is invoked (i.e., a single regime at the root is estimated), and when root.station==TRUE the group mean model of Thomas et al. 2006 (i.e., the number of means equals the number of regimes). The latter case appears to be a strange special case of OU, in that it behaves similarly to the OUMV model, but without selection. I would say that this is more consistent with the censored test of O'Meara et al. (2006), as opposed to having any real connection to OU. In any case, more work is clearly needed to understand the behavior of the group means model, and therefore, I recommend setting root.station=FALSE in the BMS case.

The Hessian matrix is used as a means to estimate the approximate standard errors of the model parameters and to assess whether they are the maximum likelihood estimates. The variance-covariance matrix of the estimated values of α and σ^2 are computed as the inverse of the Hessian matrix and the standard errors are the square roots of the diagonals of this matrix. The Hessian is a matrix of second-order derivatives and is approximated in the R package `numDeriv`. So, if changes in the value of a parameter results in sharp changes in the slope around the maximum of the log-likelihood function, the second-order derivative will be large, the standard error will be small, and the parameter estimate is considered stable. On the other hand, if the second-order derivative is nearly zero, then the change in the slope around the maximum is also nearly zero, indicating that the parameter value can be moved in any direction without greatly affecting the log-likelihood. In such situations, the standard error of the parameter will be large.

For models that allow α and σ^2 to vary (i.e., OUMV, OUMA, and OUMVA), the complexity of the model can often times be greater than the information that is contained within the data. As a result one or many parameters are poorly estimated, which can cause the function to return a log-likelihood that is suboptimal. This has great potential for poor model choice and incorrect biological interpretations. An eigendecomposition of the Hessian can provide an indication of whether the search returned the maximum likelihood estimates. If all the eigenvalues of the Hessian are positive, then the Hessian is positive definite, and all parameter estimates are considered reliable. However, if there are both positive and negative eigenvalues, then the objective function is at a saddlepoint and one or several parameters cannot be estimated adequately. One solution is to just fit a simpler model. Another is to actually identify the offending parameters. This can be done through the examination of the eigenvectors. The row order corresponds to the entries in `index.matrix`, the columns correspond to the order of values in `eigval`, and the larger the value of the row entry the greater the association between the corresponding parameter and the eigenvalue. Thus, the largest values in the columns associated with negative eigenvalues are the parameters that are causing the objective function to be at a saddlepoint.

Value

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

`$loglik`	the maximum log-likelihood.
`$AIC`	Akaike information criterion.
`$AICc`	Akaike information criterion corrected for sample-size.
`$BIC`	Bayesian information criterion.
`$mBIC`	modified Bayesian information criterion of Ho and Ane (2014).
`$model`	The model being fit.
`$param.count`	The number of parameters counted in the model.
`$solution`	a matrix containing the maximum likelihood estimates of α and σ^2.
`$theta`	a matrix containing the maximum likelihood estimates of θ and its standard error.
`$solution.se`	a matrix containing the approximate standard errors of α and σ^2. The standard error is calculated as the diagonal of the inverse of the Hessian matrix.
`$tip.fog.est`	indicates value of the measurement error if it was estimated from the data.
`$tot.state`	A vector of names for the different regimes
`$index.mat`	The indices of the parameters being estimated are returned. The numbers correspond to the row in the `eigvect` and can useful for identifying the parameters that are causing the objective function to be at a saddlepoint (see Details).
`$simmap.tree`	A logical indicating whether the input phylogeny is a SIMMAP formatted tree.
`$root.age`	The user-supplied age at the root of the tree.
`$split.point`	The user-supplied point at which regime changes are assumed to have occurred.
`$opts`	Internal settings of the likelihood search.
`$data`	User-supplied dataset.
`$phy`	User-supplied tree.

`$root.station`	A logical indicating whether the starting state, θ_0, was estimated.
`$starting.vals`	A vector of user-supplied initial search parameters.
`$lb`	The lower bound set.
`$ub`	The upper bound set.
`$iterations`	Number of iterations of the likelihood search that were executed.
`$get.root.theta`	Indicates whether the root.theta was included in the model.
`$regime.weights`	A table containing parameter estimates and the weights for time spent in each regime for each tip.
`$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 (see Details).
`$eigvect`	The eigenvectors from the decomposition of the Hessian of the likelihood function is returned (see Details).
`$new.start`	The vector of values to use if you want to restart the run from this point (starting.vals for a new run).
`$algorithm`	The algorithm used to estimate parameters

Author(s)

Jeremy M. Beaulieu and Brian C. O'Meara

References

Beaulieu J.M., Jhwueng D.C., Boettiger C., and O'Meara B.C. 2012. Modeling stabilizing selection: Expanding the Ornstein-Uhlenbeck model of adaptive evolution. Evolution 66:2369-2383.

O'Meara B.C., Ane C., Sanderson P.C., Wainwright P.C. 2006. Testing for different rates of continuous trait evolution using likelihood. Evolution 60:922-933.

Butler M.A., King A.A. 2004. Phylogenetic comparative analysis: A modeling approach for adaptive evolution. American Naturalist 164:683-695.

Ho, L.S.T., and C. Ane. 2014. Intrinsic inference difficulties for trait evolution with Ornstein-Uhlenbeck models. Methods in Ecology and Evolution, 5: 1133-1146.

Thomas G.H., Freckleton R.P., and Szekely T. 2006. Comparative analysis of the influence of developmental mode on phenotypic diversification rates in shorebirds. Proceedings of the Royal Society, B. 273:1619-1624.

Examples

```
data(tworegime)

#Plot the tree and the internal nodes to highlight the selective regimes:
select.reg<-character(length(tree$node.label))
select.reg[tree$node.label == 1] <- "black"
select.reg[tree$node.label == 2] <- "red"
plot(tree)
```

```
nodelabels(pch=21, bg=select.reg)



## Not run:
#To see the first 5 lines of the data matrix to see what how to
#structure the data:
trait[1:5,]

#Now fit an OU model that allows different sigma^2:
OUwie(tree,trait,model=c("OUMV"))

#Fit an OU model based on a clade of interest:
OUwie(tree,trait,model=c("OUMV"), clade=c("t50", "t64"), algorithm="three.point")

#For large trees, it may be useful to have ways to restart the search (due to
#finite time per run on a computing cluster, for example). You can do this
#by changing settings of OUwie runs. For example:

run1 <- OUwie(tree,trait,model=c("OUMV"), root.station=FALSE, algorithm="invert",
opts = list("algorithm"="NLOPT_LN_SBPLX", "maxeval"="500", "ftol_abs"=0.001))

save(run1, file="run1.rda")

#Then, later or in a different session:

load("run1.rda")

run2 <- OUwie(tree, trait, model=c("OUMV"), algorithm="three.point",
opts = list("algorithm"="NLOPT_LN_SBPLX", "maxeval"="500", "ftol_abs"=0.001),
starting.vals=run1$new.start)

#run2 will start off where run1 stopped.

## End(Not run)
```

OUwie.anc *Estimate ancestral states given a fitted OUwie model*

Description

Fits ancestral states (a joint estimate) given a fitted OUwie model. Currently, only works for trees with regimes painted on as node labels (rather than a simmap tree). The intended use case is just to visualize what the model is saying about evolution to help intuition (is the model something you can believe in?) rather than a firm estimate you should use to say, Yes, 56.4 MY, the ancestral body size was 17.34 mm. It could likely be anywhere from 1 to 100 mm with about equal chance. Please read details before using this function.

Usage

```
OUwie.anc(fitted.OUwie.object, opts = list("algorithm"="NLOPT_LN_BOBYQA",
"maxeval"="1000", "ftol_abs"=0.001), knowledge=FALSE, multiple_starts=1)
```

Arguments

`fitted.OUwie.object`	an object returned from the OUwie function
`opts`	a list of options to pass to nloptr for the ancestral state optimization
`knowledge`	a logical indicating whether or not you have read the documentation The default is FALSE.
`multiple_starts`	an integer indicating how many times to start the optimization. The default is 1.

Details

You probably DON'T want to use this function for anything more serious than poking around to make sure the data and model look right (oh, golly: my tips are all between 5 and 15 mm in size, and the ancestral states are 674 mm.). The request to implement ancestral state estimation resulted in many important comments from experts in the public R-SIG-PHYLO discussion forum. Here is a sampling; to see them all, go to R-SIG-PHYLO.

"So in short, yes, you can do it, with any number of methods. But why? If you can answer your biological question with methods that do not involve estimation of a parameter that is inherently fraught with error, it might be better to go another way. Bottom line - use caution and be thoughtful!" – Marguerite Butler

"I would add an extra caveat to Marguerite's excellent post: Most researchers work with extant taxa only, ignoring extinction. This causes a massive ascertainment bias, and the character states of the extinct taxa can often be very different to the ancestral state reconstructions, particularly if the evolutionary model is wrong. Eg. there has been an evolutionary trend for example. Ancestral state reconstructions based only on extant taxa should be treated as hypotheses to be tested with fossil data. I wouldn't rely on them for much more." – Simone Blomberg

"While I am at it, let me echo Simone and Marguerite's warnings. The predicted ancestral states will reflect the process you assumed to predict them. Hence, if you use them to make inferences about evolution, you will recover your own assumptions. I.e. if you predict from a model with no trend, you will find no trend, etc. Many comparative studies are flawed for this reason." – Thomas Hansen

"Let me add more warnings to Marguerite and Thomas's excellent responses. People may be tempted to infer ancestral states and then treat those inferences as data (and also to infer ancestral environments and then treat those inferences as data). In fact, I wonder whether that is not the main use people make of these inferences. But not only are those inferences very noisy, they are correlated with each other. So if you infer the ancestral state for the clade (Old World Monkeys, Apes) and also the ancestral state for the clade (New World Monkeys, (Old World Monkeys, Apes)) the two will typically not only be error-prone, but will also typically be subject to strongly correlated errors. Using them as data for further inferences is very dubious. It is better to figure out what your hypothesis is and then test it on the data from the tips of the tree, without the intermediate step of taking ancestral state inferences as observations. The popular science press in particular

demands a fly-on-the-wall account of what happened in evolution, and giving them the ancestral state inferences as if they were known precisely is a mistake." – Joe Felsenstein

"The minor twist I would throw in is that it's difficult to make universal generalizations about the quality of ancestral state estimation. If one is interested in the ancestral state value at node N, it might be reasonably estimated if it is nested high up within the phylogeny, if the rates of change aren't high, etc. And (local) trends etc might well be reliably inferred. We are pretty confident that the common ancestor of humans and chimps was larger than many deeper primate ancestors, for instance. If N is the root of your available phylogeny, however, you have to be much more cautious." – Nick Matzke

"I'll also add that I think there's a great deal to be skeptical of ancestral trait reconstruction even when large amounts of fossil data is available. You can try the exercise yourself: simulate pure BM on a non-ultrametric tree with lots of 'extinct' tips, and you'll still find pretty large confidence intervals on the estimates of the trait values. What does it mean to do ancestral trait reconstruction, if our calculations of uncertainty are that broad?" – Dave Bapst

These are some of the people who best know the power and limitations of the OU model in phylogenetics. Heed them!

To ensure that you've read this before use, please pass knowledge=TRUE as an argument to the function.

Value

`OUwie.anc` returns an object of class `OUwie.anc`. This is an OUwie object but with terminal species added at each node representing the ancestral states at each node (which are also included in the data object). There is also a `NodeRecon` element in the list that has the optimal ancestral states. There is not currently an estimate of uncertainty, but it is substantial.

Author(s)

Brian C. O'Meara

`OUwie.boot` *Parametric bootstrap function*

Description

A function that performs a parametric bootstrap for a set of user-specified model parameters

Usage

```
OUwie.boot(phy, data, model=c("BM1","BMS","OU1","OUM","OUMV","OUMA","OUMVA"),
 nboot=100, alpha, sigma.sq, theta, theta0, simmap.tree=FALSE, root.age=NULL,
 scaleHeight=FALSE, root.station=FALSE, get.root.theta=FALSE, shift.point=0.5,
 clade=NULL, tip.fog="none", algorithm=c("invert", "three.point"),
 diagn=FALSE, quiet=TRUE, warn=FALSE)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format and with internal nodes labeled denoting the ancestral selective regimes.
data	a data matrix containing species information.
model	models to fit to comparative data.
nboot	The number of bootstrap replicates.
alpha	a numeric vector giving the values of α for each selective regime.
sigma.sq	a numeric vector giving the values of σ^2 for each selective regime.
theta	a numeric vector giving the values of θ for each selective regime.
theta0	a numeric indicating the starting state, θ_0
simmap.tree	a logical indicating whether the input tree is in SIMMAP format. The default is FALSE.
root.age	indicates the age of the tree. This is to be used in cases where the "tips" are not contemporary, such as in cases for fossil trees. Default is NULL meaning latest tip is modern day.
get.root.theta	a logical indicating whether the starting state, θ_0, should be estimated (see Details).
scaleHeight	a logical indicating whether the total tree height should be scaled to 1 (see Details). The default is FALSE.
root.station	a logical indicating whether the starting state, θ_0, should be estimated (see Details).
shift.point	the point along a branch where a regime change is assumed to have occurred (if SIMMAP=FALSE. The default is set to 0.5, or halfway along a branch.
clade	a list containing a pair of taxa whose MRCA is the clade of interest.
tip.fog	designates whether a fourth column in the data matrix contains tip fog for each species value ("known"). The tip fog is assumed to be the standard error of the species mean. The default is "none".
algorithm	designates whether the standard matrix inversion ('invert') or the faster 'three-point' algorithm of Ho and Ane (2013) should be used.
diagn	a logical indicating whether the full diagnostic analysis should be carried out. The default is FALSE.
quiet	a logical indicating whether progress should be written to the screen. The default is TRUE.
warn	a logical indicating whether a warning should be printed if the number of parameters exceeds ntips/10. The default is FALSE.

Details

A simple function for conducting a parametric bootstrap on parameters estimated in OUwie. As before, the input is a tree and a data file. The tree must be of class "phylo" and if simmap=FALSE must contain the ancestral selective regimes as internal node labels. The data file is a dataframe that must have column entries in the following order: [,1] species names and [,2] their current selective

regime. The user specifies the simulated parameter values (i.e. α, σ^2, θ_0, θ), which is assumed to be the maximum likelihood estimates obtained from an OUwie run.

Note that if `root.station` is `TRUE` (the default), θ_0 was dropped from the model. In this case, then, θ_0 should be set to the value of the selective regime mapped at the root (i.e., state 1 in the "tworegime" example dataset).

Value

`OUwie.boot` returns an object of class `OUwie.boot`. This is a matrix of column length equal to the number of parameters, and row length of the number of bootstrap replicates specified.

Author(s)

Jeremy M. Beaulieu

References

Beaulieu J.M., Jhwueng D.C., Boettiger C., and O'Meara B.C. 2012. Modeling stabilizing selection: Expanding the Ornstein-Uhlenbeck model of adaptive evolution. Evolution 66:2369-2383.

O'Meara B.C., Ane C., Sanderson P.C., Wainwright P.C. 2006. Testing for different rates of continuous trait evolution using likelihood. Evolution 60:922-933.

Butler M.A., King A.A. 2004. Phylogenetic comparative analysis: A modeling approach for adaptive evolution. American Naturalist 164:683-695.

Examples

```
data(tworegime)

##First step is estimate parameters under a particular model:
pp <- OUwie(tree,trait,model=c("OUMV"),root.station=FALSE, algorithm="three.point")

##Second step is to run bootstrap replicates:
boot.reps <- OUwie.boot(tree,trait,model="OUMV", nboot=10, alpha=pp$solution[1,],
sigma.sq=pp$solution[2,],theta=pp$theta[,1], theta0=pp$theta[1,1],
algorithm="three.point")

##Finally summarize to obtain the desired confidence -- here is the 95% CI:
apply(boot.reps, 2, quantile, probs=c(0.025,0.975))
```

OUwie.contour *Generates data for contour plot of likelihood surface*

Description

Generates the likelihood surface for pairs of free parameters for generalized Ornstein-Uhlenbeck-based Hansen models of continuous characters evolving under discrete selective regimes.

Usage

```
OUwie.contour(OUwie.obj, focal.params=c("alpha_1", "sigma.sq_1"),
focal.params.lower=c(0,0), focal.params.upper=c(5,5), nreps=1000, n.cores=NULL)
```

Arguments

OUwie.obj	an object of class "OUwie" that contains the focal parameters for conducting the likelihood surface search.
focal.params	a vector specifying the parameters that you would like the likelihood surface. The format is parameter_regime – that is, for theta in regime 1, the input would be "theta_1".
focal.params.lower	a vector specifying the lower bounds for the parameters. The values need to be in the order as the focal parameters.
focal.params.upper	a vector specifying the upper bounds for the parameters. The values need to be in the order as the focal parameters.
nreps	the number points to use to estimate the likelihood surface (see Details).
n.cores	specifies the number of independent processors to conduct the analysis. The default is NULL.

Details

This function samples a set of points to estimate the likelihood surface for any pair of parameters, letting the other parameters find their own optima. This process can be very slow, as it involves optimization nrep times (though with two fewer parameters than with the chosen model, as the focal parameter values are fixed). It uses a latin hypercube design to sample points across the user-defined range of the focal parameters.

The pair of parameters to examine is passed by focal.param. The parameters need to be one of three: theta, alpha, sigma.sq. For example, to do a plot of sigma.sq from the first regime and alpha from the second regime, one would pass `focal.param = c( "sigma.sq_1", "alpha_2")`. As another example, if the regimes are characters like, flower color, the focal parameter would be `focal.param = c( "sigma.sq_Red", "sigma.sq_Blue")`.

This returns a data.frame with the last two columns being the values of the points examined and the first column the loglik of those points. The first row contains the MLE. The data.frame can be incorporated into a plotting function to obtain a contour plot (see plot.OUwie.contour).

Value

surface.data	the parameter values and loglik
focal.params	the vector specifying the parameter pair for which likelihood surface is evaluated
focal.params.lower	the vector specifying the lower bounds for the parameter pair.
focal.params.upper	the vector specifying the upper bounds for the parameter pair.

Author(s)

Jeremy M. Beaulieu

References

Beaulieu J.M., Jhwueng D.C., Boettiger C., and O'Meara B.C. 2012. Modeling stabilizing selection: Expanding the Ornstein-Uhlenbeck model of adaptive evolution. Evolution 66:2369-2383.

OUwie.dredge *Generalized Detection of shifts in OU process*

Description

Allows the hypothesis free detection of shifts in the OU process. The number and location of shifts is estimated using a user-specified information criterion.

Usage

```
OUwie.dredge(phy, data, criterion=c("AIC", "AICc", "BIC", "mBIC"), shift.max=3,
sigma.sq.max.k=3, alpha.max.k=3, root.age=NULL, scaleHeight=FALSE, root.station=FALSE,
shift.point=0.5, tip.fog="none", algorithm=c("invert", "three.point"), lb=NULL, ub=NULL,
opts = list("algorithm"="NLOPT_LN_SBPLX", "maxeval"="1000",
"ftol_rel"=.Machine$double.eps^0.5), verbose=FALSE)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format and with internal nodes labeled denoting the ancestral selective regimes
data	a dataframe containing two columns, taxon names in the first column, and species trait information in the second column.
criterion	information criterion to use for shift detection.
shift.max	maximum allowed number of shifts.
sigma.sq.max.k	maximum allowed number of sigma.sq parameters.
alpha.max.k	maximum allowed number of alpha parameters.

root.age	indicates the age of the tree. This is to be used in cases where the "tips" are not contemporary, such as in cases for fossil trees. Default is NULL meaning latest tip is modern day.
scaleHeight	a logical indicating whether the total tree height should be scaled to 1. The default is FALSE.
root.station	a logical indicating whether the starting state, θ_0, should be estimated.
shift.point	the point along a branch where a regime change is assumed to have occurred (if SIMMAP=FALSE. The default is set to 0.5, or halfway along a branch.
tip.fog	designates whether a fourth column in the data matrix contains tip fog for each species value ("known"). The tip fog is assumed to be the standard error of the species mean. The default is "none".
algorithm	designates whether the standard matrix inversion ('invert') or the faster 'three-point' algorithm of Ho and Ane (2013) should be used.
lb	if algorithm == "invert" a single value indicating the lower bound for the parameter values. if algorithm == "three.point", a vector of length 3 with position 1 as alpha lower bound, position 2 as sigma.sq's lower bound, position 3 as theta's lower bound. when set to NULL it will be a default value of 1e-9. Note that even if the model you're using doesn't include alpha (e.g. BM1), it must be included in this vector, but it will not be used to set any bounds.
ub	if algorithm == "invert" a single value indicating the upper bound for the parameter values. if algorithm == "three.point", a vector of length 3 with position 1 as alpha upper bound, position 2 as sigma.sq's upper bound, position 3 as theta's upper bound. when set to NULL it will be a default value of 100. Note that even if the model you're using doesn't include alpha (e.g. BM1), it must be included in this vector, but it will not be used to set any bounds.
opts	a list of options to pass to nloptr for the optimization: useful to adjust for faster, coarser searches
verbose	a logical indicating whether to print incremental steps

Details

This is an expanded version of the shift point model of Ho and Ane (2014). This is currently being tested, but as of now we strongly recommend using the mBIC criterion when searching for shifts.

Value

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

$loglik	the maximum log-likelihood.
$criterion	the information criterion to use for shift detection.
$criterion.score	the information criterion score used for shift detection.
$shift.model	The shift model estimated from the data.
$solution	a matrix containing the maximum likelihood estimates of α and σ^2.
$tip.fog.est	indicates value of the measurement error if it was estimated from the data.

`$theta`	a matrix containing the maximum likelihood estimates of θ.
`$tot.states`	A vector of names for the different regimes
`$index.mat`	The indices of the parameters being estimated are returned. The numbers correspond to the row in the `eigvect` and can useful for identifying the parameters that are causing the objective function to be at a saddlepoint (see Details)
`$simmap.tree`	A logical indicating whether the input phylogeny is a SIMMAP formatted tree.
`$root.age`	The user-supplied age at the root of the tree.
`$scaleHeight`	Indicates whether the tree was constrained to a total height of 1.
`$shift.point`	The user-specified portion of the branch where a regime shift occurs.
`$opts`	Settings used for optimization routine.
`$data`	The shift model dataset, which includes regime painting for each tip.
`$phy`	The shift model painted phylogeny.
`$root.station`	A logical indicating whether the starting state, θ_0, was estimated
`$starting.vals`	the starting values used for the parameter search.
`$regime.weights`	A table containing parameter estimates and the weights for time spent in each regime for each tip.

Author(s)

Jeremy M. Beaulieu

References

Ho, L.S.T., and C. Ane. 2014. Intrinsic inference difficulties for trait evolution with Ornstein-Uhlenbeck models. Methods in Ecology and Evolution, 5: 1133-1146.

`OUwie.fixed` *Generalized Hansen model likelihood calculator*

Description

Allows the user to calculate the likelihood given a specified set of parameter values

Usage

```
OUwie.fixed(phy, data, model=c("BM1","BMS","OU1","OUM","OUMV","OUMA","OUMVA"),
simmap.tree=FALSE, root.age=NULL, scaleHeight=FALSE, root.station=FALSE,
get.root.theta=FALSE, shift.point=0.5, alpha=NULL, sigma.sq=NULL, theta=NULL,
clade=NULL, tip.fog="none", check.identify=TRUE, algorithm=c("invert", "three.point"),
tip.paths=NULL, quiet=FALSE, revert.old=FALSE)
```

Arguments

phy	a phylogenetic tree, in ape "phylo" format and with internal nodes labeled denoting the ancestral selective regimes
data	a dataframe containing species information (see Details)
model	models to fit to comparative data (see Details).
simmap.tree	a logical indicating whether the input tree is in SIMMAP format. The default is FALSE.
root.age	indicates the age of the tree. This is to be used in cases where the "tips" are not contemporary, such as in cases for fossil trees. Default is NULL meaning latest tip is modern day.
scaleHeight	a logical indicating whether the total tree height should be scaled to 1 (see Details). The default is FALSE.
root.station	a logical indicating whether the starting state, θ_0, should be estimated.
get.root.theta	a logical indicating whether the starting state, θ_0, should be estimated (see Details).
shift.point	the point along a branch where a regime change is assumed to have occurred (if SIMMAP=FALSE. The default is set to 0.5, or halfway along a branch.
alpha	a numeric vector giving the values of α for each selective regime
sigma.sq	a numeric vector giving the values of σ^2 for each selective regime
theta	a numeric vector giving the values of θ for each selective regime
clade	a list containing a pair of taxa whose MRCA is the clade of interest.
tip.fog	designates whether a fourth column in the data matrix contains tip fog for each species value ("known"). The tip fog is assumed to be the standard error of the species mean. If a single numeric value is supplied it will be applied to all species uniformly. The default is "none", meaning no tip fog is used.
check.identify	a logical indicating whether to check that the user-supplied regime paintings will produce identifiable theta estimates. The default is TRUE.
algorithm	designates whether the standard matrix inversion ('invert') or the faster 'three-point' algorithm of Ho and Ane (2013) should be used.
tip.paths	an optional list that can be provided by the user where each element from 1:nTips is the path from tip to root by labeled node number. The default is NULL and this option is used for internal speedups.
quiet	a logical indicating whether or not to print progress to the screen. The default is "FALSE".
revert.old	a logical indicating whether or not to use an older implementation of the multiple alpha models. The default is "FALSE".

Details

The input is a tree and a data file. The tree must be of class "phylo" and must contain the ancestral selective regimes as internal node labels. The data file is a data.frame that must have column entries in the following order: [,1] species names and [,2] their current selective regime. The user specifies the parameter values (i.e. α, σ^2, and θ).

Value

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

$loglik	the maximum log-likelihood.
$AIC	Akaike information criterion.
$AICc	Akaike information criterion corrected for sample-size.
$BIC	Schwartz information criterion.
$model	The model being fit
$param.count	The number of parameters counted in the model.
$solution	a matrix containing the maximum likelihood estimates of α and σ^2.
$theta	a matrix containing the maximum likelihood estimates of θ.
$tot.state	A vector of names for the different regimes
$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 (see Details)
$simmap.tree	A logical indicating whether the input phylogeny is a SIMMAP formatted tree.
$root.age	The user-supplied age at the root of the tree.
$shift.point	The user-specified portion of the branch where a regime shift occurs.
$data	User-supplied dataset
$phy	User-supplied tree
$root.station	A logical indicating whether the starting state, θ_0, was estimated
$scaleHeight	Indicates whether the tree was constrained to a total height of 1.
$get.root.theta	Indicates whether the root.theta was included in the model.
$regime.weights	A table containing parameter estimates and the weights for time spent in each regime for each tip.
$algorithm	The algorithm used to estimate parameters.

Author(s)

Jeremy M. Beaulieu

Examples

```
data(tworegime)

#Calculate the likelihood based on known values of
#alpha, sigma^2, and theta:
alpha=c(0.5632459,0.1726052)
sigma.sq=c(0.1064417,0.3461386)
theta=c(1.678196,0.4185894)

OUwie.fixed(tree,trait,model=c("OUMVA"), simmap.tree=FALSE, scaleHeight=FALSE,
clade=NULL, alpha=alpha,sigma.sq=sigma.sq,theta=theta, algorithm="three.point")
```

`OUwie.format` *Format data and tree for OUwie*

Description

Simplifies conversion of a tree and, optionally, data, for OUwie.

Usage

```
OUwie.format(phy, tip.regimes=NULL, tip.data=NULL, tip.fog=NULL,
tip.fog.percentage=NULL, verbose=TRUE)
```

Arguments

`phy`	a phylogenetic tree, in `ape` "phylo" format and optionally with internal nodes labeled denoting the ancestral selective regimes
`tip.regimes`	a named vector or one-column data.frame with rownames with the regimes for each tip
`tip.data`	a named vector or one-column data.frame with rownames with the trait values for each tip
`tip.fog`	a named vector or one-column data.frame with rownames with the tip fog for each tip
`tip.fog.percentage`	a value for the percentage of the trait value to use as the tip fog
`verbose`	a logical indicating whether or not to print messages

Details

OUwie expects a tree with regimes mapped on nodes or a simmap tree. It also wants a data.frame with a column for taxon names, one for regime, one (JUST one) for trait value, and optionally one for tip fog. This function gets all of those formatted.

If you pass in no data, just the tree, the function will return a tree with regime 1 appearing everywhere, as well as a trait data.frame with regime 1 and trait value NA everywhere. It also includes zero tip fog for all taxa. This is very unrealistic, of course, and creates a bias in results (any variance in tips must be caused by the variance in the evolutionary process if tip fog is assumed to be zero).

For 'tip.regimes', 'tip.data', and 'tip.fog' the user can pass in a vector with the values and the names corresponding to the names of the tips on the tree or a single column data.frame where the rownames match the taxon names on the tree. The names of the input data and the names on the tree do not need to be in the same order, but they do need to match: "Homo sapiens" and "Homo_sapiens" do not match, for example.

Another way of handling tip dfog is assuming it's a fixed percentage of the tip value: "My uncertainty in squid arm lengths is 67 percent". To do this, pass in the percentage: 'tip.fog.percentage=67'. Note that it is per_cent: that is, for 67 percent put in 67, not 0.67. It will only work if there is already a measurement for the trait.

Value

A list with the tree ('tree') and the data ('data')

Author(s)

Brian C. O'Meara

`OUwie.sim` *Generalized Hansen model simulator*

Description

Simulates the Ornstein-Uhlenbeck process of continuous characters evolving under discrete selective regimes.

Usage

```
OUwie.sim(phy=NULL, data=NULL, simmap.tree=FALSE, root.age=NULL, scaleHeight=FALSE,
alpha=NULL, sigma.sq=NULL, theta0=NULL, theta=NULL, tip.fog="none", shift.point=0.5,
fitted.object=NULL, get.all=FALSE)
```

Arguments

`phy`	a phylogenetic tree, in ape "phylo" format and with internal nodes labeled denoting the ancestral selective regimes
`data`	a dataframe containing species information (see Details). Not necessary to include if `simmap=TRUE`.
`simmap.tree`	a logical indicating whether the input tree is in SIMMAP format. The default is `FALSE`.
`root.age`	indicates the age of the tree. This is to be used in cases where the "tips" are not contemporary, such as in cases for fossil trees. Default is `NULL` meaning latest tip is modern day.
`scaleHeight`	a logical indicating whether the total tree height should be scaled to 1 (see Details). The default is `FALSE`.
`alpha`	a numeric vector giving the values of α for each selective regime (see Details)
`sigma.sq`	a numeric vector giving the values of σ^2 for each selective regime (see Details)
`theta0`	a numeric indicating the starting state, θ_0
`theta`	a numeric vector giving the values of θ for each selective regime (see Details)
`tip.fog`	designates whether a third column in the data matrix contains tip fog for each species value ("known"). The tip fog is assumed to be the standard error of the species mean. The default is "none".
`shift.point`	the point along a branch where a regime change is assumed to have occurred (if `SIMMAP=FALSE`. The default is set to 0.5, or halfway along a branch.
`fitted.object`	a model fit from OUwie to use for simulation.
`get.all`	a logical indicating whether or not the entire simulation history is to be returned. The default is `FALSE` meaning that only the tips are returned.

Details

The input is a tree and a data file OR a fitted OUwie object. The tree must be of class "phylo" and if `simmap=FALSE` must contain the ancestral selective regimes as internal node labels. The data file is a dataframe that must have column entries in the following order: [,1] species names and [,2] their current selective regime. If `tip.fog="known"` then a third column can be added which contains the tip fog for each species. Note that if `simmap=TRUE` no data file is needed. The user specifies the simulated parameter values (i.e. α, σ^2, θ_0, θ). Assuming two selective regimes, possible models can be specified as follows (Note that this assumes a stationary distribution at the root):

a. Single rate Brownian motion (`BM1`): alpha=c(1e-10,1e-10); sigma.sq=c(0.45,0.45); theta0=1.0; theta=c(0,0).

b. Brownian motion with different rate parameters for each state on a tree (`BMS`): alpha=c(1e-10,1e-10) sigma.sq=c(0.45,0.90); theta0=1.0; theta=c(0,0).

c. Ornstein Uhlenbeck with a single optimum for all species (`OU1`): alpha=c(0.1,0.1); sigma.sq=c(0.9,0.9); theta0=1; theta=c(1.0,1.0).

d. Ornstein Uhlenbeck model that assumes different state means and a single α and σ^2 (`OUM`): alpha=c(1.0,1.0); sigma.sq=c(0.45,0.45); theta0=1.0; theta=c(1.0,2.0).

e. Ornstein Uhlenbeck model that assumes different state means and multiple σ^2 (`OUMV`): alpha=c(1.0,1.0); sigma.sq=c(0.45,0.90); theta0=1.0; theta=c(1.0,2.0).

f. Ornstein Uhlenbeck model that assumes different state means and multiple α (`OUMA`): alpha=c(1.0,0.5); sigma.sq=c(0.45,0.45); theta0=1.0; theta=c(1.0,2.0).

g. Ornstein Uhlenbeck model that assumes different state means and multiple σ^2 and α (`OUMVA`): alpha=c(1.0,0.5); sigma.sq=c(0.45,0.9); theta0=1.0; theta=c(1.0,2.0).

With a fitted OUwie model, it just uses the parameters from that, ignoring any alpha, theta, etc. set in the function.

Value

A dataframe containing, as column entries, [,1] species names, [,2] current regime, [,3] simulated continuous trait, x.

Author(s)

Jeremy M. Beaulieu and Brian C. O'Meara

Examples

```
data(sim.ex)

#Simulate an Ornstein-Uhlenbeck model with different state means
#and a separate alpha and sigma^2 per selective regime
alpha=c(1.0,0.5)
sigma.sq=c(0.45,0.9)
theta0=1.0
theta=c(1.0,2.0)

sim.data<-OUwie.sim(tree,trait,simmap.tree=FALSE,scaleHeight=FALSE,
alpha=alpha,sigma.sq=sigma.sq,theta0=theta0,theta=theta)
```

```
#Now fit a model to this and simulate from the fitted results
result <- OUwie(tree, sim.data, model="OUMVA", simmap.tree=FALSE,scaleHeight=FALSE)
sim.data.2 <- OUwie.sim(fitted.object=result)
```

`plot.OUwie.contour` *Contour plot*

Description

A plotting function for visualizing likelihood surface for a pair of parameters using OUwie.contour data.

Usage

```
## S3 method for class 'OUwie.contour'
plot(x, mle.point=NULL, levels=c(0:20*0.1), xlab=NULL,
ylab=NULL, xlim=NULL, ylim=NULL, col=grey.colors(21, start=0, end=1), ...)
```

Arguments

`x`	a `OUwie.contour` object.
`mle.point`	specifies the color for the maximum likelihood set of parameters. By default the point is not plotted (i.e., set to `NULL`.
`levels`	the levels at which to draw contour lines, measured as lnL units away from the best values.
`xlab`	allows users to specify the x-axis label.
`ylab`	allows users to specify the y-axis label.
`xlim`	allows users to specify the lower and upper limits of the x-axis.
`ylim`	allows users to specify the lower and upper limits of the y-axis.
`col`	indicates the color gradient.
`...`	additional parameters to control the plot.

Author(s)

Jeremy M. Beaulieu

References

Beaulieu J.M., Jhwueng D.C., Boettiger C., and O'Meara B.C. 2012. Modeling stabilizing selection: Expanding the Ornstein-Uhlenbeck model of adaptive evolution. Evolution 66:2369-2383.

Index