Package: pcmtools (via r-universe)

May 28, 2026

Title Tools for Phylogenetic Comparative Methods

Version 0.0.4.0

Description Various tools to help with performing phylogenetic comparative methods and curating/visualizing the results.

Depends R (>= 3.4)

License GPL (>= 2)

Encoding UTF-8

LazyData true

Imports zoo, phytools, ape

RoxygenNote 6.1.1

Config/pak/sysreqs libglpk-dev libxml2-dev

Repository https://willgearty.r-universe.dev

Date/Publication 2019-10-29 22:52:05 UTC

RemoteUrl https://github.com/willgearty/pcmtools

RemoteRef HEAD

RemoteSha 14fb49ce08d1c677971a756bb699554070482253

Contents

AICweights *Calculate AIC weights*

Description

This function takes a vector of AIC (Akaike Information Criterion) values and returns a vector of AIC weights using the formula from Burnham and Anderson (2002).

Usage

```
AICweights(AIC, na.rm = TRUE)
```

Arguments

AIC	A vector of values.
na.rm	Whether to remove NA values.

Details

If na.rm = FALSE and any values in AIC are NA, all returned values will be NA.

Value

A named vector of weights with names inherited from AIC.

Examples

```
AIC <- c(NA, 5, 10, 20, 25)
#ignore NAs
AICweights(AIC)

#should return all NAs
AICweights(AIC, na.rm = FALSE)
```

deltaAIC *Calculate deltaAIC*

Description

Calculate deltaAIC (Akaike Information Criterion), the absolute difference between the lowest AIC value and the other AIC values.

Usage

```
deltaAIC(AIC, na.rm = TRUE)
```

Arguments

AIC	A vector of values.
na.rm	Whether to remove NA values.

Details

If na.rm = FALSE and any values in AIC are NA, all returned values will be NA.

Value

A vector of weights.

Examples

```
AIC <- c(NA, 5, 10, 20, 25)
deltaAIC(AIC)

#should return all NAs
deltaAIC(AIC, na.rm = FALSE)
```

---

extractSimmapDensity *Extract posterior values from densityMap and/or contMap objects*

Description

This function takes a list of densityMap and/or contMap objects and extracts and aggregates the posterior values across them. The age and edge index of each posterior value are also included.

Usage

```
extractSimmapDensity(...)
```

Arguments

...	A vector of values.

Details

Assumes that the gradient resolution is identical for all objects (specified with res in densityMap and contMap). If the tree has a root.time element, this will be used as the max age, otherwise max(nodeHeights(tree)) will be used.

Value

A data.frame where rows represent the mapped traits through time and columns represent each of the supplied objects in .... The column names are set to the names of objects. Also includes a column for the edge index and a column for the age (in the units of the tree).

Examples

```
library(phytools)
# simulate tree
tree <- pbtree(n=70,scale=1)

# simulate discrete trait
Q <- matrix(c(-1,1,1,-1),2,2)
rownames(Q) <- colnames(Q)<-c(0,1)
x1 <- sim.history(tree,Q)$states

# generate stochastic maps and density map
mtrees <- make.simmap(tree,x1,nsim=100)
map1 <- densityMap(mtrees)

# simulate continuous trait
x2 <- fastBM(tree,sig2=0.1)

# generate cont map
map2 <- contMap(tree, x2)

# extract posterior densities
pp_data <- extractSimmapDensity(map1, map2)

# plot to see how traits have evolved with respect to one another
plot(pp_data$map1, pp_data$map2)
```

---

mergeMappedStates2 *Rename, merge, or split mapped states*

Description

This function renames, merges, or splits mapped states on a tree.

Usage

```
mergeMappedStates2(tree, old.states, new.state, node = NULL)
```

Arguments

tree	an object of class "simmap" or "multiSimmap" containing one or more phylogenetic trees with a mapped discrete character
old.states	state(s) to rename or merge
new.state	name for new state
node	a node index, used to specify a subclade of interest

Details

If node is specified, only mapped states of descendants edges of that node are modified. This function is based on the similarly named utility function by Liam Revell in phytools.

Value

An object of class "simmap" or "multiSimmap".

Examples

```
library(phytools)
# simulate a mapped tree
set.seed(4)
Q <- matrix(c(-2,1,1,1,-2,1,1,1,-2),3,3)
rownames(Q) <- colnames(Q) <- letters[1:3]
tree <- sim.history(pbtree(n=100,scale=1),Q)
cols <- setNames(c("blue","red","green","orange"),letters[1:4])

# plot the mapping
plot(tree, cols, ftype="i", fsize=0.7)

# split state c to state d within subclade
tree2 <- mergeMappedStates2(tree, "c", "d", 173)

# plot the new mapping
plot(tree2, cols, ftype="i", fsize=0.7)
```

OUwieAICSumm *Summarize the model fit across (many) OUwie results using AICc*

Description

Returns AICc weights and counts for OUwie results that have been processed using `OUwieParSumm`.

Usage

```
OUwieAICSumm(ou.parameters, na.rm = TRUE)
```

Arguments

`ou.parameters`	An object of class `ouwiepars` as output from `OUwieParSumm` or `CleanOUwieParameters`
`na.rm`	Whether to ignore model results with `NA` AICc values

Details

`na.rm = TRUE` will remove any models where `AICc == NA`, but will use other models for AIC weight calculation; `na.rm = FALSE` will remove any replicates (rows of the output) with any models where `AICc == NA`.

Value

A list with two elements:

Weights	A data.frame of the AICc weights for each model across the replicates
Counts	A named vector giving the number of replicates in which each model has the highest AICc weight

Examples

```
ou.parameters <- OUwieParSumm(ou.results, regime.mat)
OUwieAICSumm(ou.parameters)
```

OUwieCleanPar *Clean extracted parameters from OUwie results*

Description

Cleans the parameters that are extracted by OUwieParamSum based on user-specified upper and lower bounds.

Usage

```
OUwieCleanPar(ou.parameters, lower = list(), upper = list())
```

Arguments

ou.parameters	An object of class ouwiepars as output from OUwieParSumm
lower	A list of lower bounds for model parameters
upper	A list of upper bounds for model parameters

Details

Parameter estimates outside of these bounds will result in the AICc being changed to NA, which will affect downstream model averaging (see OUwieModelAvg).

Value

An ouwiepars object

Examples

```
ou.parameters <- OUwieParSumm(ou.results, regime.mat)

#Sets the AICc for the BM1 model to NA, so it wouldn't be included in downstream model averaging
OUwieCleanPar(ou.parameters, upper = list("AICc" = 45))
```

OUwieModelAvg *Model average the parameters across (many) OUwie results using AICc*

Description

Internally calculates AICc weights and uses them to model average the parameter values output from OUwieParSumm.

Usage

```
OUwieModelAvg(ou.parameters, OU.only = FALSE, na.rm = TRUE)
```

Arguments

ou.parameters	An object of class ouwiepars as output from OUwieParSumm or CleanOUwieParameters
OU.only	Whether any Brownian motion models should be dropped before model averaging
na.rm	Whether to ignore model results with NA AICc values

Details

na.rm = TRUE will remove any models where AICc == NA, but will use other models for model averaging; na.rm = FALSE will remove any replicates (rows of the output) with any models where AICc == NA.

Value

A list with two elements:

Weights	A data.frame of the AICc weights for each model across the replicates
Counts	A named vector giving the number of replicates in which each model has the highest AICc weight

Examples

```
ou.parameters <- OUwieParSumm(ou.results, regime.mat)
OUwieModelAvg(ou.parameters)
```

OUwieParSumm *Extract parameters from OUwie results*

Description

Extract various parameter values from the results of (many) OUwie analyses and maps the parameter values to different regimes based on the inputted regime map. Returns the parameters in a 4-D array, where the first dimension is the models, the second dimension is the parameters (including AICc), the third dimension is the regimes, and the fourth dimension is the replicates.

Usage

```
OUwieParSumm(ou.results, regime.mat, params = c("Alpha", "Sigma.sq",
  "Theta", "Theta.se", "AICc"))
```

Arguments

ou.results	A list of lists (or just a list) of unmodified results from an OUwie analysis
regime.mat	A data frame mapping regimes to total regime options for each model (see details)
params	A vector specifying which parameter should be calculated/returned (see details)

Details

The regime.mat is the most important component, as it indicates which parameters should be mapped to which regime for each model. For example, in an OU1 model, the user would likely want the parameters mapped to all regimes, whereas in an OUM model, the user would likely want the parameters for each regime mapped exclusively to that regime. In more complex scenarios, the user may have multiple OUM models in which regimes are split or combined in different manners, such that the parameters for one regime in one OUM model may map to multiple regimes in the overall dataset. The rownames of this matrix should identify names for the regimes and the colnames should identify the models. It is assumed that the order of the models/colnames in regime.mat matches the order of the models in ou.results.

Valid options for params are "Alpha", "Sigma.sq", "Theta", "Theta.se", "Halflife" (phylogenetic half-life), "Stat.var" (stationary variance), "AIC", "AICc", and "BIC".

Value

An ouwiepars object. Basically a 4-D array that can be passed to other functions for further analysis/visualization.

Examples

```
## Not run:
library(OUwie)
data(tworegime)
ou.results <- list()
```

```
ou.results[[1]] <- OUwie(tree,trait,model=c("BM1"))
ou.results[[2]] <- OUwie(tree,trait,model=c("BMS"), root.station = FALSE)
ou.results[[3]] <- OUwie(tree,trait,model=c("OUM"))
ou.results[[4]] <- OUwie(tree,trait,model=c("OUMV"))

#Both regimes have same parameters for BM1 model. Both regimes have different parameters for other models.
regime.mat <- data.frame(BM1 = c(1, 1), BMS = c(1,2), OUM = c(1,2), OUMV = c(1,2), row.names = c(1,2))
## End(Not run)

OUwieParSumm(ou.results, regime.mat)
```

`traitStatsThroughTime` *Calculate phenotypic statistics through time*

Description

This function reconstructs a continuous trait through time and uses these reconstructions to estimate statistics through time.

Usage

```
traitStatsThroughTime(tree, x, funs = c("mean", "sd"), rnd = 0.1)
```

Arguments

`tree`	an object of class "phylo" or "simmap" containing a phylogenetic tree (optionally with a mapped discrete character)
`x`	a vector of tip values for species; names should correspond to the respective species names in `tree$tip.labels`
`rnd`	time bin boundary rounding factor
`fun`	a vector of function names as characters

Details

For each time bin, each of the specified functions in `fun` will be carried out across the branches that pass through that time bin. The boundaries of the time bins are defined by the node ages rounded by `rnd`. If the tree has a mapped discrete character, the functions will be carried out separately for each state of the character.

Value

A list with two elements:

`stats`	A data.frame containing the statistics through time
`xy`	A data.frame containing the edge coordinates for the tree

Examples

```
library(phytools)
# simulate tree
tree <- pbtree(n=70,scale=1)

# simulate discrete trait
Q <- matrix(c(-1,1,1,-1),2,2)
rownames(Q) <- colnames(Q)<-c(0,1)
tree <- sim.history(tree,Q)

# simulate continuous trait
x2 <- fastBM(tree,sig2=0.1)

# calculate stats through time
tstt <- traitStatsThroughTime(tree, x2)
coords <- tstt$xy
stats <- tstt$stats

# plot phenogram
ys <- c(coords$y1, coords$y2)
plot(NULL, xlim=c(0,1), ylim=c(min(ys), max(ys)), ylab="trait", xlab="time")
segments(coords$x1, coords$y1, coords$x2, coords$y2)

# plot stats
plot(stats$x, stats$mean)
```

Index