Package 'cauphy'

May 8, 2026

Version 1.0.3

Title Trait Evolution on Phylogenies Using the Cauchy Process

Description The Cauchy Process can model pulsed continuous trait evolution on phylogenies. The likelihood is tractable, and is used for parameter inference and ancestral trait reconstruction.
See Bastide and Didier (2023) <doi:10.1093/sysbio/syad053>.

Depends R (>= 3.5), ape (>= 5.5)

Imports methods, robustbase, phylolm (>= 2.6.5), nloptr, pracma, foreach, doParallel, HDInterval

Suggests covr, knitr, rmarkdown, spelling, geiger, testthat (>= 3.0.0)

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Config/testthat/edition 3

LazyData true

VignetteBuilder knitr

URL https://gilles-didier.github.io/cauphy/

BugReports https://github.com/gilles-didier/cauphy/issues

Language en-US

NeedsCompilation yes

Author Gilles Didier [aut, cph],
Paul Bastide [aut, cre]

Maintainer Paul Bastide <paul.bastide@cnrs.fr>

Repository CRAN

Date/Publication 2024-10-01 13:30:05 UTC

Contents

`ancestral` *Posterior density of a node*

Description

Compute the posterior density of a node value under a fitted Cauchy process on a phylogenetic tree.

Usage

```
ancestral(x, ...)

## S3 method for class 'cauphylm'
ancestral(x, node, values, n_values = 100, n_cores = 1, ...)

## S3 method for class 'cauphyfit'
ancestral(x, node, values, n_values = 100, n_cores = 1, ...)
```

Arguments

`x`	an object of class `fitCauchy` or `cauphylm`.
`...`	other arguments to be passed to the method.
`node`	the vector of nodes for which to compute the posterior density. If not specified, the reconstruction is done on all the nodes.
`values`	the vector of values where the density should be computed. If not specified, the reconstruction is done for a grid of `n_values` values between `1.5 * min(x$y)` and `1.5 * max(x$y)`.

n_values	the number of point for the grid of values. Default to 100. Ignored if values is provided.
n_cores	number of cores for the parallelization. Default to 1.

Details

This function assumes a Cauchy Process on the tree with fitted parameters (see fitCauchy), and computes the posterior ancestral density of internal nodes, conditionally on the vector of tip values.

It computes the posterior density on all the points in values, that should be refined enough to get a good idea of the density curve.

Value

an object of S3 class ancestralCauchy, which is a matrix of posterior values, with nodes in rows and values in columns.

Methods (by class)

- ancestral(cauphylm): cauphylm object
- ancestral(cauphyfit): fitCauchy object

References

Bastide, P. and Didier, G. 2023. The Cauchy Process on Phylogenies: a Tractable Model for Pulsed Evolution. Systematic Biology. doi:10.1093/sysbio/syad053.

See Also

fitCauchy, cauphylm, plot.ancestralCauchy, plot_asr, increment, hdi.ancestralCauchy

Examples

```
set.seed(1289)
# Simulate tree and data
phy <- ape::rphylo(10, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                    parameters = list(root.value = 10, disp = 0.1))
# Fit the data
fit <- fitCauchy(phy, dat, model = "cauchy", method = "reml")
# Reconstruct the ancestral nodes
anc <- ancestral(fit)
plot_asr(fit, anc = anc, offset = 3)
plot(anc, type = "l", node = c(11, 17))
# Refine grid for node 12 and 17
anc2 <- ancestral(fit, node = c(12, 17), n_values = 1000)
plot(anc2, type = "l")
# Find HDI
library(HDInterval)
hdi_anc <- hdi(anc2)
hdi_anc
```

```
plot(anc2, interval = hdi_anc, type = "l")
```

---

cauphylm *Phylogenetic Regression using a Cauchy Process*

Description

Perform a phylogenetic regression using the Cauchy Process, by numerical optimization.

Usage

```
cauphylm(
  formula,
  data = list(),
  phy,
  model = c("cauchy", "lambda"),
  lower.bound = list(disp = 0, lambda = 0),
  upper.bound = list(disp = Inf, lambda = NULL),
  starting.value = list(disp = NULL, lambda = NULL),
  hessian = FALSE
)
```

Arguments

formula	a model formula.
data	a data frame containing variables in the model. If not found in data, the variables are taken from current environment.
phy	a phylogenetic tree of class phylo.
model	a model for the trait evolution. One of "cauchy" or "lambda" (see Details).
lower.bound	named list with lower bound values for the parameters. See Details for the default values.
upper.bound	named list with upper bound values for the parameters. See Details for the default values.
starting.value	named list initial values for the parameters. See Details for the default values.
hessian	if TRUE, then the numerical hessian is computed, for confidence interval computations. See compute_vcov.

Details

This function fits a Cauchy Process on the phylogeny, using maximum likelihood and the "fixed.root" method (see fitCauchy). It further assumes that the root value x0 is a linear combination of the covariables in formula. The corresponding regression model is:

$$Y = X\beta + E,$$

with:

Y the vector of traits at the tips of the tree;

X the regression matrix of covariables in `formula`;

β the vector of coefficients;

E a centered error vector that is Cauchy distributed, and can be seen as the result of a Cauchy process starting at 0 at the root, and with a dispersion `disp` (see `fitCauchy`).

Unless specified by the user, the initial values for the parameters are taken according to the following heuristics:

`coefficients`: β are obtained from a robust regression using `lmrob.S`;

`disp`: is initialized from the trait centered and normalized by tip heights, with one of the following statistics, taken from Rousseeuw & Croux 1993:

`IQR`: half of the inter-quartile range (see `IQR`);

`MAD`: median absolute deviation with constant equal to 1 (see `mad`);

`Sn`: Sn statistics with constant 0.7071 (see `Sn`);

`Qn`: Qn statistics with constant 1.2071 (see `Qn`).

Unless specified by the user, `disp` is taken positive unbounded.

The function uses `nloptr` for the numerical optimization of the (restricted) likelihood, computed with function `logDensityTipsCauchy`. It uses algorithms `BOBYQA` and `MLSL_LDS` for local and global optimization.

If `model="lambda"`, the CP is fit on a tree with branch lengths re-scaled using the Pagel's lambda transform (see `transf.branch.lengths`), and the `lambda` value is estimated using numerical optimization. The default initial value for the `lambda` parameter is computed using adequate robust moments. The default maximum value is computed using `phytools:::maxLambda`, and is the ratio between the maximum height of a tip node over the maximum height of an internal node. This can be larger than 1. The default minimum value is 0.

Value

`coefficients`	the named vector of estimated coefficients.
`disp`	the maximum likelihood estimate of the dispersion parameter.
`logLik`	the maximum of the log likelihood.
`p`	the number of all parameters of the model.
`aic`	AIC value of the model.
`fitted.values`	fitted values
`residuals`	raw residuals
`y`	response
`X`	design matrix
`n`	number of observations (tips in the tree)
`d`	number of dependent variables
`formula`	the model formula
`call`	the original call to the function
`model`	the phylogenetic model for the covariance
`phy`	the phylogenetic tree
`lambda`	the ml estimate of the lambda parameter (for `model="lambda"`)

References

Bastide, P. and Didier, G. 2023. The Cauchy Process on Phylogenies: a Tractable Model for Pulsed Evolution. Systematic Biology. doi:10.1093/sysbio/syad053.

Rothenberg T. J., Fisher F. M., Tilanus C. B. 1964. A Note on Estimation from a Cauchy Sample. Journal of the American Statistical Association. 59:460–463.

Rousseeuw P.J., Croux C. 1993. Alternatives to the Median Absolute Deviation. Journal of the American Statistical Association. 88:1273–1283.

See Also

fitCauchy, confint.cauphylm, ancestral, increment, logDensityTipsCauchy, phylolm

Examples

```
# Simulate tree and data
set.seed(1289)
phy <- ape::rphylo(20, 0.1, 0)
error <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                      parameters = list(root.value = 0, disp = 0.1))
x1 <- ape::rTraitCont(phy, model = "BM", sigma = 0.1, root.value = 0)
trait <- 3 + 2*x1 + error
# Fit the data
fit <- cauphylm(trait ~ x1, phy = phy)
fit
# Approximate confidence intervals
confint(fit)
```

---

compute_vcov *Compute Approximated Variance Covariance Matrix*

Description

Find the approximated variance covariance matrix of the parameters.

Usage

```
compute_vcov(obj)
```

Arguments

obj	a fitted object, either with fitCauchy or cauphylm.

Details

This function computes the numerical Hessian of the likelihood at the optimal value using function hessian, and then uses its inverse to approximate the variance covariance matrix. It can be used to compute confidence intervals with functions confint.cauphylm or confint.cauphyfit.

confint.cauphylm and confint.cauphyfit internally call compute_vcov, but do not save the result. This function can be used to save the vcov matrix.

Value

The same object, with added vcov entry.

See Also

fitCauchy, cauphylm, confint.cauphylm, confint.cauphyfit, vcov.cauphylm, vcov.cauphyfit

Examples

```
# Simulate tree and data
set.seed(1289)
phy <- ape::rphylo(20, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                    parameters = list(root.value = 10, disp = 0.1))
# Fit the data, without computing the Hessian at the estimated parameters.
fit <- fitCauchy(phy, dat, model = "cauchy", method = "reml", hessian = FALSE)
# Precompute the vcov matrix
fit <- compute_vcov(fit)
# Approximate confidence intervals
confint(fit)
```

---

fitCauchy *Model fitting for a Cauchy Process*

Description

Fit the Cauchy process on a phylogeny, using numerical optimization.

Usage

```
fitCauchy(
  phy,
  trait,
  model = c("cauchy", "lambda"),
  method = c("reml", "random.root", "fixed.root"),
  starting.value = list(x0 = NULL, disp = NULL, lambda = NULL),
  lower.bound = list(disp = 0, lambda = 0),
  upper.bound = list(disp = Inf, lambda = NULL),
  root.edge = 100,
```

```
  hessian = FALSE,
  optim = c("local", "global"),
  method.init.disp = c("Qn", "Sn", "MAD", "IQR")
)
```

Arguments

phy	a phylogenetic tree of class phylo.
trait	named vector of traits at the tips.
model	a model for the trait evolution. One of "cauchy" or "lambda" (see Details).
method	the method used to fit the process. One of reml (the default), fixed.root or random.root. See Details.
starting.value	starting value for the parameters of the Cauchy. This should be a named list, with x0 and disp the root starting value and the dispersion parameter. The default initial values are computed from standard statistics used on (independent) Cauchy variables, see Details.
lower.bound	named list with lower bound values for the parameters. See Details for the default values.
upper.bound	named list with upper bound values for the parameters. See Details for the default values.
root.edge	multiplicative factor for the root dispersion, equal to the length of the root edge. Ignored if method!=random.root.
hessian	if TRUE, then the numerical hessian is computed, for confidence interval computations. See compute_vcov.
optim	if "local", only a local optimization around the initial parameter values is performed (the default). If "global", a global maximization is attempted using the "MLSL" approach (see nloptr).
method.init.disp	the initialization method for the dispersion. One of "Qn", "Sn", "MAD", "IQR". Default to the "Qn" statistics. See Details.

Details

For the default model="cauchy", the parameters of the Cauchy Process (CP) are disp, the dispersion of the process, and x0, the starting value of the process at the root (for method="fixed.root").

The model assumes that each increment of the trait X on a branch going from node k to l follows a Cauchy distribution, with a dispersion proportional to the length t_l of the branch:

$$X_l - X_k \sim \mathcal{C}(0, \text{disp} \times t_l).$$

Unless specified by the user, the initial values for the parameters are taken according to the following heuristics:

x0: is the trimmed mean of the trait, keeping only 24% of the observations, as advocated in Rothenberg et al. 1964 (for method="fixed.root");

`disp`: is initialized from the trait centered and normalized by tip heights, with one of the following statistics, taken from Rousseeuw & Croux 1993:

`IQR`: half of the inter-quartile range (see `IQR`);

`MAD`: median absolute deviation with constant equal to 1 (see `mad`);

`Sn`: Sn statistics with constant 0.7071 (see `Sn`);

`Qn`: (default) Qn statistics with constant 1.2071 (see `Qn`).

Unless specified by the user, `x0` is taken to be unbounded, `disp` positive unbounded.

The `method` argument specifies the method used for the fit:

`method="reml"`: the dispersion parameter is fitted using the REML criterion, obtained by re-rooting the tree to one of the tips. See `logDensityTipsCauchy` for the default choice of the re-rooting tip;

`method="random.root"`: the root value is assumed to be a random Cauchy variable, centered at `x0=0`, and with a dispersion `disp_root = disp * root.edge`;

`method="fixed.root"`: the model is fitted conditionally on the root value `x0`, i.e. with a model where the root value is fixed and inferred from the data.

In the first two cases, the optimization is done on the dispersion only, while in the last case the optimization is on the root value and the dispersion.

The function uses `nloptr` for the numerical optimization of the (restricted) likelihood, computed with function `logDensityTipsCauchy`. It uses algorithms `BOBYQA` and `MLSL_LDS` for local and global optimization.

If `model="lambda"`, the CP is fit on a tree with branch lengths re-scaled using the Pagel's lambda transform (see `transf.branch.lengths`), and the `lambda` value is estimated using numerical optimization. The default initial value for the `lambda` parameter is computed using adequate robust moments. The default maximum value is computed using `phytools:::maxLambda`, and is the ratio between the maximum height of a tip node over the maximum height of an internal node. This can be larger than 1. The default minimum value is 0.

Value

An object of S3 class `cauphyfit`, with fields:

`x0`	the fitted starting value (for `method="fixed.root"`)
`disp`	the ml or reml estimate of the dispersion parameter
`lambda`	the ml or reml estimate of the lambda parameter (for `model="lambda"`)
`logLik`	the maximum of the log (restricted) likelihood
`p`	the number of parameters of the model
`aic`	the AIC value of the model
`trait`	the named vector of traits at the tips used in the fit
`y`	the named vector of traits at the tips used in the fit
`n`	the number of tips in the tree
`d`	the number of dependent variables
`call`	the original call of the function

model	the phylogenetic model (one of "cauchy" or "lambda")
phy	the phylogenetic tree
method	the method used (one of "reml", "fixed.root", "random.root")
random.root	TRUE if method="random.root"
reml	TRUE if method="reml"
root_tip_reml	name of the tip used to reroot the tree (for method="reml")

References

Bastide, P. and Didier, G. 2023. The Cauchy Process on Phylogenies: a Tractable Model for Pulsed Evolution. Systematic Biology. doi:10.1093/sysbio/syad053.

Rothenberg T. J., Fisher F. M., Tilanus C. B. 1964. A Note on Estimation from a Cauchy Sample. Journal of the American Statistical Association. 59:460–463.

Rousseeuw P.J., Croux C. 1993. Alternatives to the Median Absolute Deviation. Journal of the American Statistical Association. 88:1273–1283.

See Also

confint.cauphyfit, profile.cauphyfit, ancestral, increment, logDensityTipsCauchy, cauphylm, fitContinuous

Examples

```
# Simulate tree and data
set.seed(1289)
phy <- ape::rphylo(20, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                    parameters = list(root.value = 10, disp = 0.1))
# Fit the data
fit <- fitCauchy(phy, dat, model = "cauchy", method = "reml")
fit
# Approximate confidence intervals
confint(fit)
# Profile likelihood
pl <- profile(fit)
plot(pl)
```

---

hdi.ancestralCauchy *Highest (Posterior) Density Interval*

Description

This function takes an object of class ancestralCauchy, result of function ancestral or increment, and find the Highest (Posterior) Density Interval of reconstructed states for given nodes. It relies on function hdi from package HDInterval.

Usage

```
## S3 method for class 'ancestralCauchy'
hdi(object, credMass = 0.95, allowSplit = TRUE, node, ...)
```

Arguments

object	an object of class ancestralCauchy, result of function ancestral or increment.
credMass	a scalar between 0 and 1 specifying the mass within the credible interval.
allowSplit	if FALSE and the proper HDI is discontinuous, a single credible interval is returned, but this is not HDI. See hdi for details. Default to TRUE.
node	the vector of nodes where to plot the ancestral reconstruction. Can be missing, in which case all the nodes reconstructed in the ancestralCauchy
...	further arguments to be passed to plot.

Details

The function relies on the density method of the hdi function. Package HDInterval must be loaded in the workspace for this function to work. See documentation of this functions for more details on the definition and computation of the HDI.

The density is obtained on the grid of values defined by the ancestralCauchy object, which defaults to 100 values. See details in the documentation of the ancestral and increment functions.

NOTE: if the grid of values is too coarse (if it has too few values), then the result can be a poor approximation. Please make sure to use an appropriate grid in the reconstruction to get meaningful results (see example).

Value

A named list. Each item of the list is named after a node, and contains the HDI interval of the node, in the same format as in hdi: a vector of length 2 or a 2-row matrix with the lower and upper limits of the HDI, with an attribute credMass. If allowSplit=TRUE, the matrix has a row for each component of a discontinuous HDI and columns for begin and end. It has an additional attribute "height" giving the probability density at the limits of the HDI.

See Also

plot.ancestralCauchy, ancestral, increment, fitCauchy

Examples

```
# Lizard dataset
data(lizards)
attach(lizards)
# Fit CP
fit_CP <- fitCauchy(phy, svl, model = "cauchy", method = "reml")
# Reconstruct increments for some branches
inc <- increment(fit_CP, node = c(142, 151), n_cores = 1)
# HDI
library(HDInterval)
```

```
inc_int <- hdi(inc)
plot(inc, intervals = inc_int, type = "l")
# HDI of edge ending at node 142 is unimodal
inc_int[["142"]]
# HDI of edge ending at node 151 is bimodal
inc_int[["151"]]
# If the grid is coarse, the result is meaningless
inc <- increment(fit_CP, node = c(151), n_cores = 1, n_values = 10)
inc_int <- hdi(inc)
plot(inc, intervals = inc_int, type = "l")
```

`increment` *Posterior density of an increment*

Description

Compute the posterior density of a branch increment under a fitted Cauchy process on a phylogenetic tree.

Usage

```
increment(x, ...)

## S3 method for class 'cauphylm'
increment(x, node, values, n_values = 100, n_cores = 1, ...)

## S3 method for class 'cauphyfit'
increment(x, node, values, n_values = 100, n_cores = 1, ...)
```

Arguments

`x`	an object of class `fitCauchy` or `cauphylm`.
`...`	other arguments to be passed to the method.
`node`	vector of nodes ending the branches for which to compute the posterior density of the increment. If not specified, the reconstruction is done on all the possible edges.
`values`	the vector of values where the density should be computed. If not specified, the reconstruction is done for a grid of `n_values` values between `-1.5 * maxdiff` and `1.5 * maxdiff`, where `maxdiff` is the difference between the larger and smaller tip value.
`n_values`	the number of point for the grid of values. Default to `100`. Ignored if `values` is provided.
`n_cores`	number of cores for the parallelization. Default to 1.

Details

This function assumes a Cauchy Process on the tree with fitted parameters (see fitCauchy), and computes the posterior ancestral density of trait increments at branches (ie, the difference between the traits value at the end and beginning of the branch), conditionally on the vector of tip values.

It computes the posterior density on all the points in values, that should be refined enough to get a good idea of the density curve.

Value

an object of S3 class ancestralCauchy, which is a matrix of posterior increment values, with nodes in rows and values in columns.

Methods (by class)

- increment(cauphylm): cauphylm object
- increment(cauphyfit): fitCauchy object

References

Bastide, P. and Didier, G. 2023. The Cauchy Process on Phylogenies: a Tractable Model for Pulsed Evolution. Systematic Biology. doi:10.1093/sysbio/syad053.

See Also

fitCauchy, cauphylm, plot.ancestralCauchy, plot_asr, ancestral, hdi.ancestralCauchy

Examples

```
set.seed(1289)
# Simulate tree and data
phy <- ape::rphylo(10, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                    parameters = list(root.value = 10, disp = 0.1))
# Fit the data
fit <- fitCauchy(phy, dat, model = "cauchy", method = "reml")
# Reconstruct the ancestral increments
inc <- increment(fit)
plot_asr(fit, inc = inc, offset = 3)
plot(inc, node = c(3, 8), type = "l")
# Refine grid for edges ending at tips 3 and 8
inc2 <- increment(fit, node = c(3, 8), values = seq(-3, 3, 0.01))
plot(inc2, type = "l")
# Find HDI
library(HDInterval)
hdi_inc <- hdi(inc2)
hdi_inc
plot(inc2, interval = hdi_inc, type = "l")
```

lizards *Greater Antillean Anolis lizard dataset*

Description

A dataset containing the dated phylogeny and the log snout-to-vent length for Greater Antillean Anolis lizard species, taken from Mahler et al. 2013.

Usage

```
lizards
```

Format

A data frame with 53940 rows and 10 variables:

phy Bayesian maximum clade credibility chronogram for Greater Antillean Anolis from Mahler et al. 2013

svl Natural log-transformed species average snout-to-vent length

ecomorph Ecomorph assignments for each of the species

Source

doi:10.5061/dryad.9g182

References

Mahler, D. Luke; Ingram, Travis; Revell, Liam J.; Losos, Jonathan B. (2013), Data from: Exceptional convergence on the macroevolutionary landscape in island lizard radiations, Dryad, Dataset, https://doi.org/10.5061/dryad.9g182

logDensityTipsCauchy *Log Density of a Cauchy Process*

Description

Compute the log density of the vector of trait at the tips of the phylogenetic tree, assuming a Cauchy process.

Usage

```
logDensityTipsCauchy(
  tree,
  tipTrait,
  root.value = NULL,
  disp,
  method = c("reml", "random.root", "fixed.root"),
  rootTip = NULL,
  do_checks = TRUE
)
```

Arguments

tree	a phylogenetic tree of class phylo.
tipTrait	a names vector of tip trait values, with names matching the tree labels.
root.value	the root starting value of the process.
disp	the dispersion value.
method	the method used to compute the likelihood. One of reml (the default), fixed.root or random.root. See Details.
rootTip	the tip used to re-root the tree, when the REML method is used. If NULL, the tip with the smallest distance to all the others is used (see Details). Ignored in method != "reml".
do_checks	if FALSE, the entry parameters are not checked for consistency. This can be useful when doing multiple calls to the function, as in numerical optimization. Default to TRUE.

Details

The parameters of the Cauchy Process (CP) are disp, the dispersion of the process, and root.value, the starting value of the process at the root (for method="fixed.root").

The model assumes that each increment of the trait X on a branch going from node k to l follows a Cauchy distribution, with a dispersion proportional to the length t_l of the branch:

$$X_l - X_k \sim \mathcal{C}(0, \text{disp} \times t_l).$$

The method argument specifies the type of likelihood that is computed:

method="reml": the dispersion parameter is fitted using the REML criterion, obtained by re-rooting the tree to one of the tips. The default tip used to reroot the tree is: rootTip = which.min(colSums(cophenetic.phylo(tree))). Any tip can be used, but this default empirically proved to be the most robust numerically;

method="random.root": the root value is assumed to be a random Cauchy variable, centered at root.value=0, and with a dispersion disp_root = disp * root.edge;

method="fixed.root": the model is fitted conditionally on the root value root.value, i.e. with a model where the root value is fixed and inferred from the data.

Value

the log density value.

See Also

fitCauchy

Examples

```
phy <- ape::rphylo(5, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy", parameters = list(root.value = 0, disp = 1))
logDensityTipsCauchy(phy, dat, 0, 1, method = "fixed.root")
```

---

plot.ancestralCauchy *Plot for class* ancestralCauchy

Description

This function takes an object of class ancestralCauchy, result of function ancestral or increment, and plots the reconstructed states for given nodes.

Usage

```
## S3 method for class 'ancestralCauchy'
plot(x, node, n_col, intervals = NULL, ...)
```

Arguments

x	an object of class ancestralCauchy, result of function ancestral or increment.
node	the vector of nodes where to plot the ancestral reconstruction. Can be missing, in which case all the nodes reconstructed in the ancestralCauchy object are plotted.
n_col	the number of columns on which to display the plot. Can be missing, in which case a default number is used.
intervals	a list of HDI intervals produced by function hdi.ancestralCauchy. If the HDI of a plotted node is in the list, then it is plotted by the function.
...	further arguments to be passed to plot.

Value

None.

See Also

plot_asr, ancestral, increment, fitCauchy

Examples

```
set.seed(1289)
# Simulate tree and data
phy <- ape::rphylo(10, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                    parameters = list(root.value = 10, disp = 0.1))
# Fit the data
fit <- fitCauchy(phy, dat, model = "cauchy", method = "reml")
# Reconstruct the ancestral values
inc <- increment(fit, node = c(3, 8), values = seq(-3, 3, 0.01))
plot(inc, type = "l")
anc <- ancestral(fit, node = c(12, 17), n_values = 1000)
plot(anc, type = "l")
```

---

plot.profile.cauphyfit

Plot for class `profile.cauphyfit`

Description

This function takes an object of class `profile.cauphyfit`, and plots the profile likelihood for each parameter.

Usage

```
## S3 method for class 'profile.cauphyfit'
plot(x, n_col, ...)
```

Arguments

`x`	an object of class `profile.cauphyfit`
`n_col`	the number of columns on which to display the plot. Can be left blank.
`...`	further arguments to be passed to `plot`.

Value

None.

See Also

`profile.cauphyfit`, `fitCauchy`.

Examples

```
phy <- ape::rphylo(5, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy", parameters = list(root.value = 0, disp = 1))
fit <- fitCauchy(phy, dat, model = "cauchy", method = "fixed.root")
pr <- profile(fit)
plot(pr)
```

---

plot_asr *Plot Ancestral States Reconstructions*

Description

Plot the ancestral states reconstructions from a fitted Cauchy model.

Usage

```
plot_asr(
  x,
  anc = NULL,
  inc = NULL,
  common_colorscale = FALSE,
  x.legend = "topleft",
  y.legend = NULL,
  adj = c(0.5, 0.5),
  piecol = NULL,
  width.node = NULL,
  height.node = NULL,
  width.edge = NULL,
  height.edge = NULL,
  style = "bars",
  offset = 1,
  scaling = 1,
  x.lim = NULL,
  x.intersp = NULL,
  ...
)
```

Arguments

x	a cauphylm or fitCauchy object.
anc	(optional) an object of class ancestralCauchy, obtained with ancestral.
inc	(optional) an object of class ancestralCauchy, obtained with increment.
common_colorscale	If both plotted, should the ancestral states and the increment be represented by the same color scale ? Default to FALSE.

`x.legend`, `y.legend`
: the x and y co-ordinates to be used to position the legend. They can be specified by keyword or in any way which is accepted by `legend`.

`adj`
: one or two numeric values specifying the horizontal and vertical, respectively, justification of the text or symbols. By default, the text is centered horizontally and vertically. If a single value is given, this alters only the horizontal position of the text.

`piecol`
: a list of colours (given as a character vector) to be used by `thermo` or `pie`; if left `NULL`, a series of colours given by the function `rainbow` is used.

`width.node`, `height.node`, `width.edge`, `height.edge`
: parameters controlling the aspect of thermometers for the nodes and the edges; by default, their width and height are determined automatically.

`style`
: a character string specifying the type of graphics; can be abbreviated (see details).

`offset`
: offset of the tip labels (can be negative).

`scaling`
: the scaling factor to apply to the data.

`x.lim`
: a numeric vector of length one or two giving the limit(s) of the x-axis. If `NULL`, this is computed with respect to various parameters such as the string lengths of the labels and the branch lengths. If a single value is given, this is taken as the upper limit.

`x.intersp`
: character interspacing factor for horizontal (x) spacing between symbol and legend text (see `legend`).

`...`
: other parameters to be passed on to `plot.phylo` or `phydataplot`.

Details

The main plot is done with `plot.phylo`, the node annotation use `nodelabels`, and the tip data plot use `phydataplot`. Please refer to these functions for the details of the parameters.

The width of each color in the thermo plots approximately represents the weight of each node of the distribution, that is estimated by numerically integrating the density function around each mode. Function `findpeaks` is first used to find the modes and estimate their starting and ending points. Then function `trapz` estimates the integral of the density around the mode.

For an exact representation of a node posterior density, please plot it separately, using function `plot.ancestralCauchy`.

Value

None.

See Also

`cauphylm`, `fitCauchy`, `ancestral`, `increment`, `plot.phylo`, `phydataplot`, `nodelabels`

Examples

```
set.seed(1289)
# Simulate tree and data
phy <- ape::rphylo(10, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                    parameters = list(root.value = 10, disp = 0.1))
# Fit the data
fit <- fitCauchy(phy, dat, model = "cauchy", method = "reml")
# Reconstruct the ancestral states and increments
inc <- increment(fit, n_values = 100)
anc <- ancestral(fit, n_values = 100)
plot_asr(fit, inc = inc, anc = anc, offset = 3,
         width.node = 0.8, height.node = 0.5,
         width.edge = 1.5, height.edge = 0.2,
         x.legend = "topright")
```

---

posteriorDensityAncestral

Posterior density of a node

Description

Compute the posterior density of a set of node values under a Cauchy process on a phylogenetic tree.

Usage

```
posteriorDensityAncestral(
  node,
  vals,
  tree,
  tipTrait,
  root.value = NULL,
  disp,
  method = c("reml", "random.root", "fixed.root")
)
```

Arguments

node	the node for which to compute the posterior density.
vals	the table of values where the density should be computed.
tree	a phylogenetic tree of class phylo.
tipTrait	a names vector of tip trait values, with names matching the tree labels.
root.value	the root starting value of the process.

disp	the dispersion value.
method	the method used to compute the likelihood. One of reml (the default), fixed.root or random.root. See Details.

Details

This function is internally called by ancestral, which is the preferred way of doing ancestral reconstruction on a fitted object.

Value

the posterior density value.

See Also

ancestral, fitCauchy

Examples

```
phy <- ape::rphylo(5, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy", parameters = list(root.value = 0, disp = 1))
posteriorDensityAncestral(7, 0.1, phy, dat, disp = 1)
```

---

posteriorDensityIncrement

Posterior density of an increment

Description

Compute the posterior density of a set of branch increments under a Cauchy process on a phylogenetic tree.

Usage

```
posteriorDensityIncrement(
  node,
  vals,
  tree,
  tipTrait,
  root.value = NULL,
  disp,
  method = c("reml", "random.root", "fixed.root")
)
```

Arguments

node	the node ending the branch for which to compute the posterior density of the increment.
vals	the table of values where the density should be computed.
tree	a phylogenetic tree of class phylo.
tipTrait	a names vector of tip trait values, with names matching the tree labels.
root.value	the root starting value of the process.
disp	the dispersion value.
method	the method used to compute the likelihood. One of reml (the default), fixed.root or random.root. See Details.

Details

This function is internally called by increment, which is the preferred way of doing ancestral reconstruction on a fitted object.

Value

the posterior density value.

See Also

increment, fitCauchy

Examples

```
set.seed(1289)
phy <- ape::rphylo(5, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy", parameters = list(root.value = 0, disp = 1))
posteriorDensityIncrement(2, 0.1, phy, dat, disp = 1)
```

---

print.cauphyfit *Generic Methods for S3 class* cauphyfit.

Description

Generic Methods for S3 class cauphyfit.

Usage

```
## S3 method for class 'cauphyfit'
print(x, digits = max(3, getOption("digits") - 3), ...)

## S3 method for class 'cauphyfit'
vcov(object, ...)

## S3 method for class 'cauphyfit'
logLik(object, ...)

## S3 method for class 'logLik.cauphyfit'
AIC(object, k = 2, ...)

## S3 method for class 'cauphyfit'
AIC(object, k = 2, ...)

## S3 method for class 'cauphyfit'
confint(object, parm, level = 0.95, ...)

## S3 method for class 'cauphyfit'
coef(object, ...)
```

Arguments

x	an object of class "phylolm".
digits	number of digits to show in summary method.
...	further arguments to methods.
object	an object of class cauphyfit.
k	numeric, the penalty per parameter to be used; the default k = 2 is the classical AIC.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required.

Value

Same value as the associated methods from the stats package:

vcov an estimated covariance matrix, see compute_vcov;

logLik an object of class logLik;

AIC a numeric value;

confint a matrix (or vector) with columns giving lower and upper confidence limits for each parameter;

coef coefficients extracted from the model;

See Also

fitCauchy, vcov, logLik AIC, confint, coef, predict, predict.phylolm

Examples

```
# Simulate tree and data
set.seed(1289)
phy <- ape::rphylo(20, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                    parameters = list(root.value = 10, disp = 0.1))
# Fit the data
fit <- fitCauchy(phy, dat, model = "cauchy", method = "reml")
fit
# vcov matrix
vcov(fit)
# Approximate confidence intervals
confint(fit)
# log likelihood of the fitted object
logLik(fit)
# AIC of the fitted object
AIC(fit)
# coefficients
coef(fit)
```

---

print.cauphylm *Generic Methods for S3 class* cauphylm.

Description

Generic Methods for S3 class cauphylm.

Usage

```
## S3 method for class 'cauphylm'
print(x, digits = max(3, getOption("digits") - 3), ...)

## S3 method for class 'cauphylm'
vcov(object, ...)

## S3 method for class 'cauphylm'
logLik(object, ...)

## S3 method for class 'logLik.cauphylm'
AIC(object, k = 2, ...)

## S3 method for class 'cauphylm'
AIC(object, k = 2, ...)
```

```
## S3 method for class 'cauphylm'
predict(object, newdata = NULL, se.fit = FALSE, ...)

## S3 method for class 'cauphylm'
confint(object, parm, level = 0.95, ...)

## S3 method for class 'cauphylm'
coef(object, ...)
```

Arguments

x	an object of class "phylolm".
digits	number of digits to show in summary method.
...	further arguments to methods.
object	an object of class cauphylm.
k	numeric, the penalty per parameter to be used; the default k = 2 is the classical AIC.
newdata	an optional data frame to provide the predictor values at which predictions should be made. If omitted, the fitted values are used. Currently, predictions are made for new species whose placement in the tree is unknown. Only their covariate information is used. The prediction for the trend model is not currently implemented.
se.fit	A switch indicating if standard errors are required.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required.

Value

Same value as the associated methods from the stats package:

vcov an estimated covariance matrix, see compute_vcov;

logLik an object of class logLik;

AIC a numeric value;

confint a matrix (or vector) with columns giving lower and upper confidence limits for each parameter;

coef coefficients extracted from the model;

predict a vector of predicted values.

See Also

cauphylm, vcov, logLik AIC, confint, coef, predict, predict.phylolm

Examples

```
# Simulate tree and data
set.seed(1289)
phy <- ape::rphylo(20, 0.1, 0)
error <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                      parameters = list(root.value = 0, disp = 0.1))
x1 <- ape::rTraitCont(phy, model = "BM", sigma = 0.1, root.value = 0)
trait <- 3 + 2*x1 + error
# Fit the data
fit <- cauphylm(trait ~ x1, phy = phy)
fit
# vcov matrix
vcov(fit)
# Approximate confidence intervals
confint(fit)
# log likelihood of the fitted object
logLik(fit)
# AIC of the fitted object
AIC(fit)
# predicted values
predict(fit)
# coefficients
coef(fit)
```

---

profile.cauphyfit *Method for Profiling* cauphyfit *Objects*

Description

Investigates the profile log-likelihood function for a fitted model of class cauphyfit.

Usage

```
## S3 method for class 'cauphyfit'
profile(fitted, which = 1:npar, level = 0.8, npoints = 100, ...)
```

Arguments

fitted	the cauphyfit fitted model object.
which	the original model parameters which should be profiled. This can be a numeric or character vector. By default, all parameters are profiled.
level	highest confidence level for parameters intervals, computed using the approximated Hessian (see compute_vcov).
npoints	number of points to profile the likelihood for each parameter.
...	further arguments passed to or from other methods.

Details

This function computes a confidence interval for the parameters using `confint.cauphyfit`, and then computes the likelihood function on a grid with `npoints` values evenly spaced between the bounds of the interval, for each parameter one by one, all other parameters being fixed.

Value

An object of class `profile.cauphyfit`, which is a list with an element for each parameter being profiled. The elements are data-frames with two variables:

`par.vals`**:** a matrix of parameter values for each fitted model.

`profLogLik`**:** the profile log likelihood.

See Also

`fitCauchy`, `plot.profile.cauphyfit`, `profile`.

Examples

```
phy <- ape::rphylo(5, 0.1, 0)
dat <- rTraitCauchy(n = 1, phy = phy, model = "cauchy", parameters = list(root.value = 0, disp = 1))
fit <- fitCauchy(phy, dat, model = "cauchy", method = "reml")
pr <- profile(fit)
plot(pr)
```

---

rTraitCauchy *Cauchy Trait Simulation*

Description

Simulate a continuous trait using the Cauchy Process

Usage

```
rTraitCauchy(
  n = 1,
  phy,
  model = c("cauchy", "lambda", "kappa", "delta"),
  parameters = NULL
)
```

Arguments

`n`	number of independent replicates
`phy`	a phylogeny in `ape` `phylo` format.
`model`	a phylogenetic model. Default is "cauchy", for the Cauchy process. Alternative are "lambda", "kappa", and "delta".
`parameters`	list of parameters for the model (see Details).

Details

The default choice of parameters is as follow:

`model = cauchy root.value = 0, disp = 1`

`model = lambda root.value = 0, disp = 1, lambda = 1`

`model = kappa root.value = 0, disp = 1, kappa = 1`

`model = delta root.value = 0, disp = 1, delta = 1`

Value

If n=1, a numeric vector with names from the tip labels in the tree. For more than 1 replicate, a matrix with the tip labels as row names, and one column per replicate.

See Also

rTrait, rTraitCont

Examples

```
set.seed(1289)
phy <- ape::rphylo(40, 0.01, 0)
# One trait
y <- rTraitCauchy(n = 1, phy = phy, model = "cauchy",
                  parameters = list(root.value = 0, disp = 0.1))
y
plot(phy, x.lim = c(0, 750))
phydataplot(y, phy, offset = 150)
# Many trait
y <- rTraitCauchy(n = 10, phy = phy, model = "cauchy",
                  parameters = list(root.value = 0, disp = 0.1))
head(y)
```

Index