Package: mcmc3r (via r-universe)

August 15, 2024

Type Package

Title Tools to work with MCMCtree

Version 0.5.6

Date 2024-03-27

Description Tools to work with MCMCtree, a program for Bayesian inference of species divergence times.

License MIT + file LICENSE

LazyData TRUE

Encoding UTF-8

Depends R (>= 3.5.0)

Suggests Morpho, Rdpack, knitr, rmarkdown, testthat (>= 0.11.0)

VignetteBuilder knitr

Imports ape, corpcor, coda

RoxygenNote 7.3.1

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

RemoteUrl https://github.com/dosreislab/mcmc3r

RemoteRef HEAD

RemoteSha 8d3f3b1c5736e5b73420a7b0bd60dcd2534db384

Contents

`A` *A matrix*

Description

Matrix which, once multiplied by its transpose, yields to the inverse of the estimate of the shrinkage correlation matrix, R.sh. The latter is used to transform the data set as it corrects the morphological data set for the corresponding character correlation

Usage

```
A
```

Format

A matrix of size n x n, where n = 87 (morphological traits: 29 landmarks x 3D coordinates):

n Number of traits for which the correlation values have been calculated, 87

`array2matrix` *Convert an array into a matrix*

Description

Convert an array with landmark points into an object of class matrix.

Usage

```
array2matrix(X, coords = c(2, 3))
```

Arguments

`X`	Array, `k` landmark points, `q` coordinates, and `s` specimens.
`coords`	Integer, 2 or 3, for 2D or 3D landmarks, respectively.

Details

The object `X`, class array, has format `k x q x s`, where `k` is the number of landmarks, `q` the number of coordinates, and `s` the number of specimens. See `C.arr.unal` for an example of the format of a 3D array and `data-raw/C.R` for the details about how to generate this object.

Value

An object of class matrix, with `s` rows, one for specimen, and `n` columns, one for each coordinate of the landmarks. Each landmark can be given in 2D or 3D. For instance, if the landmarks are 3D, the first 3 columns will be the coordinates x, y, and z for the first landmark, the next 3 columns for the second landmark, and so on.

specimens	lmk1.x	lmk1.y	lmk1.z	lmk2.x	lmk2.y	lmk2.z	...
Sp_1	0.143	-0.028	-0.044	0.129	0.028	-0.043	...
Sp_2	0.128	-0.024	-0.028	0.124	0.027	-0.025	...
...	...	...	...	...	...	...	...

See object `C` for a more detailed example of the format of the object that is returned and `data-raw/C.R` for the explanation about how to generate this object.

Note that if the names for the `s` matrices, one per specimen, in the array are not provided, i.e. the names for specimens are not given, the specimens in the returned matrix, `row.names(matrix)`, will be labelled as '1', '2', and so on.

Author(s)

Sandra Alvarez-Carretero and Mario dos Reis

See Also

`matrix2array`, `write.morpho`

`bayes.factors` *Calculate Bayes factors and posterior model probabilities*

Description

Calculate Bayes factors and posterior model probabilities

Usage

```
bayes.factors(..., prior = NULL, boot = TRUE, n = 10000, prob = 0.95)
```

Arguments

`...`	list of marginal likelihood objects, see details
`prior`	numeric, the prior model probabilities
`boot`	logical, whether to perform parametric boostrap of probabilities
`n`	numeric, number of bootstrap samples
`prob`	numeric, the probability used to calculate the boostrap CI

Details

Input is a list of marginal likelihood objects, with each object generated by either `stepping.stones()` or `gauss.quad()`. If `boot = TRUE`, parametric bootstrap is performed by assuming the log-marginal likelihood estimates are normally distributed with standard deviation equal to the standard error. The re-sampled `n` marginal log-likelihoods are used to estimate re-sampled posterior probabilities and to calculate an equal-tail bootstrap confidence interval for these.

Note that the length of `prior` should be the same as the number of models being compared. The `prior` is rescaled so that `sum(prior) == 1`.

Value

A list with elements `bf` and `logbf`, the Bayes factors and log-Bayes factors; `pr`, the posterior model probabilities; `prior` the prior model probabilities and, if `boot = TRUE`, `pr.ci` the equal-tail bootstrap confidence interval.

Author(s)

Mario dos Reis

Examples

```
# See Table 5 in dos Reis et al. (2018, Syst. Biol., 67: 594-615)
# Bayesian selection of relaxed clock models for the 1st and 2nd sites
# of mitochondrial protein-coding genes of primates
# Models: strick clock, independent-rates, and autocorrelated-rates
sc <- list(); sc$logml <- -16519.03; sc$se <- .01
ir <- list(); ir$logml <- -16480.58; ir$se <- .063
ar <- list(); ar$logml <- -16477.82; ar$se <- .035
bayes.factors(sc, ir, ar)
bayes.factors(sc, ir, ar, prior=c(.25,.5,.25))
bayes.factors(sc, ir, ar, prior=c(0,1,0))
```

block.boot *Generate block bootstrap replicates of sampled power likelihoods*

Description

Generate block bootstrap replicates of sampled power likelihoods

Usage

```
block.boot(R, p = 0.1, mcmcf = "mcmc.txt", betaf = "beta.txt", preff = "lnL")
```

Arguments

R	numeric, number of bootstrap replicates
p	numeric, block length, giving as a proportion of the MCMC sample size
mcmcf	character, mcmc output file name
betaf	character, file with beta values
preff	character, prefix for files storing boot replicates

Details

Block bootstrap replicates are generated using the stationary boostrap method of Politis and Romano (1994). The replicates are stored in files named using `preff` and the replicate number. For example, if `preff = "lnL"` (the default) then the files are lnL0.txt, lnL1.txt, lnL2.txt, ..., etc, with lnL0.txt corresponding to the original log-likelihood sample. Replicates are stored within the directories corresponding to the appropriate beta values. The collection of files can grow large quickly so you may want to use a small number of replicates (say R = 10 to R = 100).

This function uses code from the boot package by Canty and Ripley.

References

Álvarez-Carretero et al (2022) A species-level timeline of mammal evolution integrating phylogenomic data. *Nature*, 602: 263–267.

Politis and Romano (1994) The stationary boostrap. *J. Am. Stat. Assoc.*, 89: 1303–1313.

See Also

stepping.stones.boot and tsboot (from the boot package).

`C` *29 3D landmarks from the skulls of 19 carnivoran specimens after Procrustes analysis*

Description

A matrix containing the 29 3D landmarks collected from the skulls of 19 carnivoran specimens after carrying out a Procrustes analysis (PA). Please take a look at the description in morpho/data-raw/C.R to understand how this object was generated. This is the morphological alignment.

Usage

```
C
```

Format

A matrix with s = 19 rows and n = 87 columns (87/3 = 29 landmarks):

s Rows, specimens from which landmarks were collected, 19

n Columns, 87 traits (29 landmarks in 3D) after the PA

`C.arr.unal` *29 3D landmarks from the skulls of 19 carnivoran specimens before Procrustes analysis*

Description

A 3D array containing the 29 3D landmarks collected from the skulls of 19 carnivoran specimens before carrying out a Procrustes analysis (PA). Please take a look at the description in morpho/data-raw/C.R to understand how this object was generated.

Usage

```
C.arr.unal
```

Format

An array with k = 29 (landmarks), q = 3 (coordinates) and s = 19 (specimens):

k landmark points collected from 19 carnivoran specimens, 29

q coordinates in 3D or 2D, 3

s number of specimens, 18

`C.mat.unal` *29 3D landmarks from the skulls of 19 carnivoran specimens before Procrustes analysis*

Description

A matrix containing the 29 3D landmarks collected from the skulls of 19 carnivoran specimens before carrying out a Procrustes analysis (PA). Please take a look at the description in morpho/data-raw/C.R to understand how this object was generated.

Usage

```
C.mat.unal
```

Format

A matrix with s = 19 rows and n = 87 columns (87/3 = 29 landmarks):

s Rows, specimens from which landmarks were collected, 19

n Columns, 87 traits (29 landmarks in 3D) after the PA

`C.PS` *Object of class procSym output by Morpho after PA*

Description

Object of class procSym output by Morpho, which mean shape is later used to align the foxes ("Vulpes vulpes") specimens to it. Please take a look at the description in morpho/data-raw/C.R to understand how this object was generated.

Usage

```
C.PS
```

Format

Object procSym

... Check `procSym` for more details

calibrations *Calibration densities*

Description

Density, distribution, and quantile functions for calibrations used in MCMCtree.

Usage

```
dL(x, tL, p = 0.1, c = 1, pL = 0.025)

pL(q, tL, p = 0.1, c = 1, pL = 0.025)

qL(prob, tL, p = 0.1, c = 1, pL = 0.025)

dB(x, tL, tU, pL = 0.025, pU = 0.025)

dU(x, tU, pU = 0.025)
```

Arguments

`x`	numeric, vector of quantiles
`tL`	numeric, minimum age
`p`	numeric, mode parameter for truncated Cauchy
`c`	numeric, tail decay parameter for truncated Cauchy
`pL`	numeric, minimum probability bound
`q`	numeric, quantile
`prob`	numeric probability
`tU`	numeric, maximum age
`pU`	numeric, maximum probability bound

Details

Calculates the density, distribution and quantile functions for the minimum (dL) calibration, and the density function for the joint (dB) and maximum (dU) calibration bounds as implemented in MCMCtree. See Yang and Rannala (2007) and Inoue et al. (2010) for details.

Value

A vector of density, probability, or quantile values as appropriate.

Author(s)

Mario dos Reis

References

Yang and Rannala. (2006) Bayesian Estimation of Species Divergence Times Under a Molecular Clock Using Multiple Fossil Calibrations with Soft Bounds. *Mol. Biol. Evol.*, 23: 212–226.

Inoue, Donoghue and Yang (2010) The Impact of the Representation of Fossil Calibrations on Bayesian Estimation of Species Divergence Times. *Syst. Biol.*, 59: 74–89.

Examples

```
# Plot a minimum bound calibration density:
curve(dL(x, 1), from=0, to=10, n=5e2)

# Plot a joint bounds calibration density:
curve(dB(x, 1, 6), from=0, to=10, n=5e2)

# Plot a maximum bound calibration density:
curve(dU(x, 6), from=0, to=10, n=5e2)

# Probability and quantile function for minimum bound (or truncated-Cauchy):
qv <- 0:20
# calculate probability vector from quantiles:
pv <- pL(qv, tL=1)
# calculate quantiles back from probability vector:
# (note numerical error)
qL(pv, tL=1)
```

`carnivores` *Carnivores dataset*

Description

Dataset of carnivores containing alignments of morphlogical continuous characters, a phylogeny, and an MCMC matrix produced by MCMCtree.

Usage

```
carnivores
```

Format

`carnivores` is a list with elements:

`C.raw`, a matrix of 29 3D-landmarks measurements from 19 species;

`V.raw`, a matrix of 29 3D-landmarks for 21 common foxes (Vulpes vulpes);

`C.proc` and `V.proc`, the corresponding matrices after Procrustes alignment;

`var.foxes` and `R.sh`, with estimates of the variance vector and correlation matrix for the landmarks in the 21 foxes;

`M`, a matrix of transformed landmarks (using the foxes variances and correlation matrix, with Cholesky metric) for the 19 carnivores;

`pairedLM`, a matrix describing the symmetry of the landmarks;

`tree`, the phylogeny of the 19 carnivores; and

`mcmc`, an MCMC sample of divergence times and morphological and molecular rates from an MCMCtree analysis.

Source

Alvarez-Carretero S, Goswami A, Yang Z and dos Reis M. (2018) Bayesian estimation of species divergence times using correlated quantitative characters. *Syst. Biol.,* 68: 967–986.

`carnivores19x29.raw` *29 3D landmarks from the skulls of 19 carnivoran specimens*

Description

A dataset containing the 29 3D landmarks collected from the skulls of 19 carnivoran specimens This data.frame consists of a first column with the specimen labels used by MCMCtree, a second column with the voucher names of the specimens collected, and then 87 columns with the coordinates for each trait (29 landmarks x 3D coordinates). Please take a look at the description in morpho/data-raw/carnivores19x29.raw.R to understand how this object was generated.

Usage

```
carnivores19x29.raw
```

Format

A data.frame with s = 19 rows and p = 89 columns (2 info columns + 87 traits):

s Rows, specimens from which landmarks were collected, 19

p Columns, specimens information in 1st and 2nd column + 87 traits (29 landmarks in 3D)

`ctlMCMCtree` *Generating the control file to run MCMCtree when using morphological data*

Description

Function that outputs the control file to run MCMCtree. It can be used to run only with morphological data or morphological+molecular data (more than one partition in the alignment file) together.

Usage

```
ctlMCMCtree(
  filename,
  mol = FALSE,
  seed = -1,
  seqfile,
  treefile,
  mcmcfile = "mcmc.txt",
  outfile = "out.txt",
  ndata,
  seqtype = 0,
  usedata = 1,
  clock,
  RootAge,
  TipDate,
  alpha,
  ncatG,
  cleandata = 0,
  BDparas,
  kappa_gamma,
  alpha_gamma,
  rgene_gamma,
  sigma2_gamma,
  print = 2,
  burnin,
  sampfreq,
  nsample,
  model
)
```

Arguments

`filename`	Character, name for the output control file.
`mol`	Boolean, TRUE if you include molecular data, FALSE otherwise. Default = FALSE.
`seed`	Numeric, seed value. Default = -1 (assigns random seed using computer's current time).
`seqfile`	Character, path to the alignment file.
`treefile`	Character, path to the tree file.
`mcmcfile`	Character, path to the file with the report of MCMC runs. By default it generates a file called "mcmc.txt" in the directory where MCMCtree is run.
`outfile`	Character, path to the summary results file. By default it generates a file called "out.txt" in the directory where MCMCtree is run.
`ndata`	Numeric, number of partitions in the alignment file.
`seqtype`	Numeric, 0 for nucleotide sequences, 1 for codon sequences, and 2 for amino acid sequences. Default = 0.

`usedata`	Numeric, 1: Calculate the likelihood function in the normal way, 0: Likelihood is not calculated (likelihood = 1), 2 and 3: approximate likelihood calculation and ML estimation of branch lengths (see details). Default = 1.
`clock`	Numeric, 1: strict clock model, 2: independent rates model, 3: autocorrelated rates model (see details).
`RootAge`	Character, calibration for the root.
`TipDate`	Numeric, time unit to scale the estimated divergence times. See details.
`alpha`	Numeric, alpha value for the discrete-gamma model of rate variation. Only used if molecular data is available.
`ncatG`	Numeric, number of categories for the discrete-gamma model of rate variation. Only used if molecular data is available.
`cleandata`	Numeric, 0: alignment gaps and ambiguity characters are treated as missing data, 1: any site where at least one sequences has an alignment gap or ambiguity character is deleted (see details). Default = 0.
`BDparas`	Numeric, vector with the parameters controlling the birth-death-sequential-sampling (BDSS) process (see details).
`kappa_gamma`	Numeric, vector with the parameters for the substitution model parameter kappa (transition/transversion rate ratio). Only used if molecular data is available.
`alpha_gamma`	Numeric, vector with the parameters for the substitution model parameter gamma (gamma shape parameter for variable rates among sites). Only used if molecular data is available.
`rgene_gamma`	Numeric, vector with the parameters for the Dirichlet-gamma prior for the mean substitution rate (see details).
`sigma2_gamma`	Numeric, vector with the parameters for the Dirichlet-gamma prior for the rate drift parameter (see details).
`print`	Numeric, 0: results are printed to screen only, 1: MCMC is written to "mcmc-file" and the summary to the "outfile", 2: same as 1 but rates for branches for each partitions are appended to "outfile". Default = 2.
`burnin`	Numeric, number of iterations to be discarded (burn-in).
`sampfreq`	Numeric, number of iterations after which a sample will be collected.
`nsample`	Numeric, number of samples to be gathered.
`model`	Numeric, substitution model to be used (see details). 0:JC69, 1:K80, 2:F81, 3:F84, 4:HKY85, 5:T92, 6:TN93, 7:REV, 8:UNREST, 9:REVu; 10:UNRESTu. Only used if molecular data is available.

Details

For more information, please check the MCMCtree tutorial and the PAML documentation.

Examples

```
# First create objects with the path to alignment and tree files and then
# call the function to generate the control file. The parameters not passed
# to the function are used as the default values
```

```
tree <- system.file( "extdata", "19s.trees", package = "morpho")
aln  <- system.file( "extdata", "seqfile.aln", package = "morpho")

# Uncomment the following lines followed by "##" and change the path in the filename so
# it is saved where you want
##ctlMCMCtree( filename = "../mcmctree.ctl", mol = FALSE, seqfile = aln, treefile = tree,
##ndata = 1, clock = 2, TipDate = 1, RootAge = c("B(37.3, 66.0, 0.025, 0.025)"),
##BDparas = c( 1, 1, 0, 0.001 ), rgene_gamma = c( 2, 5 ),
##sigma2_gamma = c( 2, 2 ), burnin = 50000, sampfreq = 50, nsample = 20000 )
```

---

dBD *Birth-death process with species sampling*

Description

Kernel density function for the birth-death process with species sampling.

Usage

```
dBD(x, lambda, mu, rho, t1 = 1)
```

Arguments

`x`	numeric, vector of quantiles.
`lambda`	numeric, birth rate.
`mu`	numeric, death rate.
`rho`	numeric, proportion of species sampled.
`t1`	numeric, age of the phylogeny's root.

Details

MCMCtree uses the BD kernel to generate the prior on node ages for those nodes without fossil calibrations. You can look at the examples below for some suggestions. Note rho must be between 0 and 1. The special case mu = lambda, rho=0 gives a uniform density. See Yang and Rannala (2006) for full details.

Value

A numeric vector of probability densities.

Author(s)

Mario dos Reis

References

Yang and Rannala. (2006) Bayesian Estimation of Species Divergence Times Under a Molecular Clock Using Multiple Fossil Calibrations with Soft Bounds. *Mol. Biol. Evol.*, 23: 212–226.

Yang (2014) Molecular Evolution: A Statistical Approach. *Oxford University Press*

Examples

```
# Reproduce Fig. 10.10 from Yang (2014)
# (a) lambda = mu = 1, rho = 0 (uniform density):
curve(dBD(x, 1, 1, 0), xlim=c(0, 1), ylim=c(0, 4), xaxs="i", yaxs="i")

# (b) lambda = 10, mu = 5, rho = 0.01 (old node ages, useful for diversified
# sampling):
curve(dBD(x, 10, 5, .01), from=0, to=1, lty=2, add=TRUE)

# (c) lambda = 10, mu = 5, rho = 0.001 (old node ages, useful for diversified
# sampling):
curve(dBD(x, 10, 5, .001), from=0, to=1, lty=3, add=TRUE)

# (d) lambda = 10, mu = 5, rho = 0.99 (young node ages, useful for dense
# sampling of diverse phylogenies):
curve(dBD(x, 10, 5, .99), from=0, to=1, lty=4, add=TRUE)
```

`gauss.quad` *Estimate marginal likelihood by thermodynamic integration*

Description

Estimate marginal likelihood by thermodynamic integration and Gauss-Legendre quadrature from a sample of `n` power posterior MCMC chains sampled with mcmctree (or bpp).

Usage

```
gauss.quad(mcmcf = "mcmc.txt", betaf = "beta.txt", se = TRUE)
```

Arguments

`mcmcf`	character, mcmc output file name
`betaf`	character, file with beta values
`se`	logical, whether to calculate the standard error

Details

The MCMC samples should be stored in a directory structure created by `make.bfctlf` with `method = "gauss-quad"`. The function will read the stored log-likelihood values and calculate the log-marginal likelihood.

Numerical integration is done using Gauss-Legendre quadrature. See Rannala and Yang (2017) for details (also dos Reis et al. 2017, Appendix 2).

Value

A list with components `logml`, the log-marginal likelihood estimate; `se`, the standard error of the estimate; `mean.logl`, the mean of log-likelihood values sampled for each beta; and `b`, the beta values used.

Author(s)

Mario dos Reis

References

Rannala B and Yang Z. (2017) Efficient Bayesian species tree inference under the multispecies coalescent. *Systematic Biology* 66: 823-842.

dos Reis et al. (2017) Using phylogenomic data to explore the effects of relaxed clocks and calibration strategies on divergence time estimation: Primates as a test case. *bioRxiv*

See Also

`make.bfctlf` to prepare directories and mcmctree or bpp control files to calculate the power posterior.

`hominids` *A BPP A00 MCMC sample for an hominid phylogeny*

Description

This dataset contains the results from the BPP A00 analysis of hominid evolution from Angelis and dos Reis (2015).

Usage

```
hominids
```

Format

`hominids` is a list with elements `mcmc`, a dataframe with 20,000 rows and 8 columns, and `tree`, an object of class `phylo` from the `ape` package.

`mcmc` is a posterior sample from a BPP A00 MCMC analysis containing the relative divergence times (tau's) and nucleotide diversities (theta's) for the four species ape (hominid) phylogeny.

`tree` contains the phylogeny with node ages given as the posterior means of the tau's in `mcmc`.

Source

K. Angelis and M. dos Reis (2015) *The impact of ancestral population size and incomplete lineage sorting on Bayesian estimation of species divergence times.* Curr. Zool., 61: 874–885.

See Also

microcebus

`lmk_imp` *Import various landmark files for different specimens at once*

Description

Import more than one csv file with landmark points and return an array object with dimensions p x k x n, being p the number of landmarks, k the dimension of the coordinates (2D or 3D), and n the number of specimens (the number of files, as each file contains the landmarks for one specimen).

Usage

```
lmk_imp(path = NULL, lmk.names = FALSE)
```

Arguments

`path`	Character, absolute path to the directory with the csv files with the landmark points are.
`lmk.names`	Logical, TRUE if there is an extra column for landmark names, FALSE otherwise (see details).

Details

Note that all files need to be comma separated files (csv).

If `lmk.names = TRUE`, the format expected for 3D landmarks is the following:

landmarks	x	y	z
lmk_1	0.143	-0.028	-0.030
lmk_2	0.128	-0.024	-0.035
...	...	...	

Otherwise, if `lmk.names = TRUE`, then the format is:

x	y	z
0.143	-0.028	-0.030
0.128	-0.024	-0.035
...	...	...

Note that you can always have 2D landmarks, so the format is the same but the column with the z landmarks will not appear in the files.

Author(s)

Sandra Alvarez-Carretero

`logL.boot`	*Estimate marginal likelihood from bootstrap replicates*

Description

Estimate marginal likelihood from bootstrap replicates

Usage

```
stepping.stones.boot(R, betaf = "beta.txt", preff = "lnL")

gauss.quad.boot(R, betaf = "beta.txt", preff = "lnL")
```

Arguments

`R`	numeric, number of bootstrap replicates used
`betaf`	character, file with beta values
`preff`	character, prefix for files storing boot replicates

Details

`stepping.stones.boot` and `gauss.quad.boot` are used to calculate the marginal likelihoods on bootstrap replicates using the stepping stones and gaussian quadrature methods respectively. The replicates must have been generated using block.boot.

Value

A list with components `logml`, the original log-marginal likelihood estimate, `logmlR`, the vector of log-marginal likelihood estimates on the boostrap replicates, `se` and `ci`, the standard error and 95% credibility interval of `logml` calculated on the bootstrap replicates, and `b`, the beta values used.

See Also

block.boot, stepping.stones and gauss.quad.

`make.beta`	*Make beta values for marginal likelihood calculation*

Description

Make appropriate beta values

Usage

```
make.beta(n, method = c("step-stones", "gauss-quad"), a = 5)
```

Arguments

`n`	numeric, number of beta points
`method`	character, the method to choose the beta points, see details
`a`	numeric, exponent for stepping stones beta generation, see details

Details

If `method = "step-stones"`, the beta values are given by the formula

$$\beta_i = \left(\frac{i-1}{n}\right)^a.$$

Values of `a` between 5 to 8 appear appropriate. Large `a` values produce beta values close to zero.

If `method = "gauss-quad"`, the beta values are calculated according to the `n` Gauss-Legendre quadrature rule (see Rannala and Yang, 2017).

Value

Numeric vector with `n` beta values

Author(s)

Mario dos Reis

References

Rannala B and Yang Z (2017) Efficient Bayesian species tree inference under the multispecies coalescent. *Systematic Biology*, 66: 823–842.

See Also

The generated beta values are suitable input for `make.bfctlf`.

`make.bfctlf` *Prepare mcmctree or bpp control files for marginal likelihood calculation*

Description

Prepare mcmctree or bpp control files for marginal likelihood calculation

Usage

```
make.bfctlf(beta, ctlf = "mcmctree.ctl", betaf = "beta.txt")
```

Arguments

`beta`	numeric vector of beta values
`ctlf`	character, mcmctree or bpp control file template
`betaf`	character, file onto which to write selected beta values

Details

This function generates a set of `n` directories each containing a modified `ctlf` control file with the appropriate beta value to run mcmctree (or bpp) to obtain MCMC samples under the required power-posterior distribution. For the general theory of marginal likelihood calculation with power posteriors see Yang (2014).

The beta values are printed to `betaf`.

Author(s)

Mario dos Reis

References

Yang Z (2014) *Molecular Evolution: A Statistical Approach*. Oxford University Press. Pages 256–260.

See Also

`make.beta`, `stepping.stones`

`matrix2array` *Convert a matrix into an array*

Description

Convert a matrix with landmark points into an object of class array.

Usage

```
matrix2array(X, coords = c(2, 3))
```

Arguments

`X`	Matrix of size `s x n`, `n` landmark points for `s` specimens (see details).
`coords`	Numeric, 2 or 3 for 2D or 3D landmarks, respectively.

Details

The matrix has format `s x n`, with `s` rows, one per specimen, and `n` columns, one for each coordinate of the landmarks. Each landmark can be given in 2D or 3D. For instance, if the landmarks are 3D, the first 3 columns will be the coordinates x, y, and z for the first landmark, the next 3 columns for the second landmark; and so on:

specimens	lmk1.x	lmk1.y	lmk1.z	lmk2.x	lmk2.y	lmk2.z	...
Sp_1	0.143	-0.028	-0.044	0.129	0.028	-0.043	...
Sp_2	0.128	-0.024	-0.028	0.124	0.027	-0.025	...
...	...	...	...	...	...	...	...

See object `C` for an example of its format and `data-raw/C.R` to see how this object is generated.

Value

An object of class array with format `k x q x s`, where `k` is the number of landmarks, `q` the number of coordinates, and `s` the number of specimens.

Note that if the matrix provided does not have rownames, the specimens in the returned array (names for the 's' matrices accessed through the array, i.e. `dimnames( array )[ 3 ]` will be labelled as '1', '2', and so on. See object `C.arr.unal` for an example of the format of the object that is returned and `data-raw/C.R` for the description of how to obtain this object.

Author(s)

Sandra Alvarez-Carretero and Mario dos Reis

See Also

`array2matrix`, `write.morpho`

`mcmc.sum` *Calculate summaries from an MCMC run*

Description

Calculate summaries from an MCMC run

Usage

```
mcmc.sum(mcmc)
```

Arguments

`mcmc`	a data frame with MCMCtree's output

Details

The data frame should have headers and should contain the output from the mcmc.txt file generated by MCMCtree.

Value

A data frame with the mean, median, and 95

`mcmc2anc` *Ancestral character reconstruction from an MCMC sample*

Description

Obtain the ancestral reconstruction of characters from an MCMC sample using Felsenstein (1973) model of continuous character evolution.

Usage

```
mcmc2anc(tree, M, mcmc, time.name, rate.name, tip.ages = NULL)
```

Arguments

`tree`	a rooted, strictly bifurcating phylogeny
`M`	s x k matrix of k continuous morphological measurements for s species
`mcmc`	data frame with MCMC output from MCMCtree
`time.name`	character vector of length one
`rate.name`	character vector of length one
`tip.ages`	numeric, the ages of the terminal taxa in the tree

Details

The function first calculates the mean of divergence times and morphological rates in the MCMC sample. These are used to reconstruct the branch lengths in units of morphological evolution, and then Eq. (7) in Felsenstein (1973) is used to calculate the ancestral reconstruction on the tree.

Note `time.name` is the name format used for the node ages in the MCMC dataframe, usually of the form `time.name = "t_"`. Similarly `rate.name` is the name format used in the MCMC sample for the rates, of the form `rate.name = "r_g1_"` where the subscript in "g" must be the partition number containing the morphological rates. Note `tree` must be the same used by MCMCtree when generating the MCMC sample, right down to its Newick representation. Taxon names in `tree` and `M` must match.

Value

A n x k matrix with the ancestral reconstruction for the k characters at the n internal nodes of the phylogeny.

Author(s)

Mario dos Reis

References

Felsenstein J (1973) Maximum-likelihood estimation of evolutionary trees from continuous characters. *Am J Hum Genet,* 25: 471–492.

See Also

write.morpho

Examples

```
data(carnivores)
# calculate tip ages correctly, as they're needed by the function:
tips_info <- strsplit(carnivores$tree$tip.label, "\\^" )
back.ages <- as.numeric(unlist(tips_info)[seq(from=2, to=2*19, by=2)])
back.ages <- max(back.ages) - back.ages
C <- carnivores$C.proc
rownames(C) <- rownames(carnivores$M)
recM = mcmc2anc(carnivores$tree, C, mcmc=carnivores$mcmc, time.name="t_",
       rate.name="r_g1_", tip.ages=back.ages)

x <- seq(1, 87, by=3); y <- seq(2, 87, by=3)
# Plot landmarks for the 19 carnivores:
plot(carnivores$C.proc[,x], carnivores$C.proc[,y], pch='+', cex=.5)
# plot ancestral reconstruction at the root (node 20):
points(recM["20",x], recM["20",y], pch=19, col="red")
# mean shape
mS <- apply(carnivores$C.proc, 2, mean)
points(mS[x], mS[y], pch=19, col="blue")

# Convert reconstruction to an array, as is the standard in
# morphometrics software
## Not run:
recA <- matrix2array(recM, 3)
options(rgl.printRglwidget = TRUE)
rgl::plot3d(recA[,,"20"], ty='s', size=2, col="red", aspect=FALSE)

## End(Not run)
```

---

mcmc2densitree *Plot a densi-tree from an MCMC sample*

Description

Plot a densi-tree from an MCMC sample from a BPP or MCMCTree analysis

Usage

```
mcmc2densitree(
  tree,
  mcmc,
  time.name,
  thin,
  col = "blue",
  alpha = 1,
  y.offset = 0,
  pfract = 0.1,
  plot.labels = TRUE,
  cex.lab = 1,
  axis = TRUE,
  add = FALSE,
  tip.ages = NULL
)
```

Arguments

`tree`	an object of class phylo.
`mcmc`	data frame with an MCMC sample from MCMCTree or a BPP A00 analysis.
`time.name`	character vector of length one.
`thin`	numeric, the fraction of MCMC samples to keep.
`col`	character, the color for branches.
`alpha`	numeric, between 0 and 1, the branch color transparency.
`y.offset`	numeric, the vertical offset for plotting the tree.
`pfract`	numeric, how much of the plotting space to used for plotting the tip labels. If `pfrac = 1`, the same amount of space is used for the tree and the labels. Use large values if your tip labels are long.
`plot.labels`	logical, whether to plot the tip labels. Ignored if `add = TRUE`.
`cex.lab`	numeric, the relative character size for the tip labels.
`axis`	logical, whether to plot the x axis.
`add`	logical, if TRUE add the trees to an existing plot, otherwise create a new plot.
`tip.ages`	numeric, the ages of the tips, with the most recent tip having age zero, and the oldest tip having the largest age. If `NULL`, tips are assumed to have all age zero.

Details

The function will reduce the MCMC sample to `dim(mcmc)[1] * thin` observations. Then the node ages in each observarion are used to plot each tree in the sample. For a tree with `s` species. The y coordinates of the tips are given by `0:(s - 1) + y.offset`.

The `tree` must be rooted, strictly bifurcating, and be the same tree used to genarate the BPP (A00) or MCMCTree MCMC samples.

Author(s)

Mario dos Reis

Examples

```
data(microcebus)
mcmc2densitree(microcebus$tree, microcebus$mcmc, time.name="tau_", thin=0.05,
 alpha=0.01, col="blue")
 title(xlab="Distance (substitutions per site)")

# data(hominids) TODO: Fix this example (add msc2time.t function)
# Calibrate the hominid phylogeny with a uniform fossil calibration of
# between 6.5 to 10 Ma for the human-chimp divergence, and plot the
# calibrated sample
#calmsc <- msc2time.t(mcmc=hominids$mcmc, node="7humanchimp", calf=runif,
#   min=6.5, max=10)
# mcmc2densitree(hominids$tree, calmsc, "t_", thin=0.05, alpha=0.01)
# title(xlab="Divergence time (Ma)")
```

---

microcebus *A BPP A00 MCMC sample for a mouse lemur phylogeny*

Description

This dataset contains the results from the BPP A00 analysis of mouse lemur evolution in Madagascar from Yoder et al. (2016).

Usage

```
microcebus
```

Format

`microcebus` is a list with elements `mcmc`, a dataframe with 20,000 rows and 12 columns, and `tree`, an object of class `phylo` from the `ape` package.

`mcmc` is a posterior sample from a BPP A00 MCMC analysis containing the relative divergence times (tau's) and nucleotide diversities (theta's) for the six species mouse lemur (*Microcebus* spp) phylogeny.

`tree` contains the phylogeny with node ages given as the posterior means of the tau's in `mcmc`.

Source

A. D. Yoder, C. R. Campbell, M. B. Blanco, M. dos Reis, J. U. Ganzhorn, S. M. Goodman, K. E. Hunnicutt, P. A. Larsen, P. M. Kappeler, R. M. Rasoloarison, J. M. Ralison, D. L. Swofford, and D. W. Weisrock. (2016) *Geogenetic patterns in mouse lemurs (genus Microcebus) reveal the ghosts of Madagascar's forests past.* Proc. Nat. Acad. Sci. USA., 113: 8049–8056.

See Also

hominids

---

proc2MCMCtree *Procrustes alignment output in MCMCtree format for Bayesian inference*

Description

Perform a Procrustes alignment with the procSym and align2procSym and generate a morphological alignment in MCMCtree format.

Usage

```
proc2MCMCtree(
  data,
  popdata,
  sp.data,
  sp.popdata,
  filename,
  coords,
  method = c("eigen", "chol"),
  ages,
  ...
)
```

Arguments

`data`	Matrix or array, object with the data set with one specimen per species. See details.
`popdata`	Matrix or array, containing the specimens for the population of species with a larger variation (more than one specimen for this species). See details.
`sp.data`	Numeric, position of the specimen in the `data` object also present in the `popdata` object (see details).
`sp.popdata`	Numeric, position of the specimen in the `popdata` object also present in the `data` object (see details).
`filename`	Character, name for the output file.
`coords`	Numeric, 2 or 3 for 2D or 3D landmarks, respectively.
`method`	(Optional) character, either `"eigen"` or `"col"`, method used to decompose the inverse of the shrinkage correlation matrix. See details.
`ages`	(Optional) list or vector, ages of the species included in the morpholical alignment.
`...`	(Optional) Further arguments passed to procSym (see details)

Details

If the objects `data` and `popdata` are of class "matrix", then they have `s` or `ps` rows, respectively, and `n` columns as of the amount of characters. These data sets are supposed to contain landmarks, which can be given in 2D or 3D. For instance, if the landmarks are 3D, the first 3 columns will be the coordinates x, y, and z for the first landmark, the next 3 columns for the second landmark; and so on.

specimens	lmk1.x	lmk1.y	lmk1.z	lmk2.x	lmk2.y	lmk2.z	...
Sp_1	0.143	-0.028	-0.044	0.129	0.028	-0.043	...
Sp_2	0.128	-0.024	-0.028	0.124	0.027	-0.025	...
...	...	...	...	...	...	...	...

You can load the object `C.mat.unal` to see an example of this format and also take a look at the `data-raw/C.R` file for details about how to generate it. Otherwise, if the objects `data` and `popdata` are of class "array", they have format `k` x `q` x `s` or `k` x `q` x `ps`, respectively, where `k` is the number of landmarks, `q` the number of coordinates, `s` the number of specimens for object `data`, and `ps` the number of specimens in a sampled population of one species for object `popdata`. You can load the object `C.arr.unal` to see an example of this array format and also take a look at the `data-raw/C.R` file for details about how to generate it.

The names of the specimens will be taken from either the row names of the data matrices, if `data` and `popdata` objects are of class "matrix", or from the names of the third dimension of the array, if these objects are of class "array". If no names are found, the specimens will be labelled as "Specimen_1", "Specimen_2", and so on.

Note that you are providing a population matrix with the landmarks collected from more than on specimen belonging to the same species. Therefore, it is assumed that population noise is accounted for. Furthermore, the landmarks of one of these specimens are also present in the morphological alignment. First, a Procrusts alignment is performed with the data set with one specimen per species (`data`), and then all the specimens in `popdata` except for the specimen common in `data`, which position is specified in the argument `sp.popdata`, are aligned to the mean shape of the PA generated with `data`. Take a look at `data.raw/V.R` for more details.

The logarithm of the determinant of the correlation matrix is going to be printed in the output file to later be used by MCMCtree during the likelihood calculation.

The function `procSym` can perform a Procrustes superimposition alignment given a dataset of landmarks in array format and allows the user to pass different options to the arguments defined. The current function runs with the default parameters in `procSym` but allows the user to pass three arguments to `proc2MCMCtree` through `...`: scale, reflect, and pairedLM. If you are thinking of using these arguments, read the documentation in `procSym`, otherwise do not pass further arguments to `proc2MCMCtree` and let it run `procSym` in default mode.

Value

`$dataPS`	Object of class `symproc` output by the function `Morpho::procSym`. The user can access the rest of the items in this object by checking <your_object>$dataPS$<available_Morpho_objects It contains the array with the morphological alignment generated by this function, which corresponds to the specimens in the object passed to `data`. This can be accessed by loading <your_object>$dataPS$rotated.

`$popdataPS`	Array with the morphological alignment with the object passed to `popdata`, i.e. the alignment with the population sample
`$M`	Matrix with the morphological alignment with the specimens of all species. Note that this alignment is not corrected for character correlation nor population noise. The corrected alignment is only printed in the output alignment file.
`$Rsh`	Estimated shrinkage correlation matrix.
`$c`	Estimated population variance.

Author(s)

Sandra Alvarez-Carretero and Mario dos Reis

See Also

`write.morpho`, `matrix2array`, `array2matrix`

Examples

```
## Not run:
# A. Use the unaligned, but processed, carnivoran data (data = C.mat.unal) and
#    vulpes data (popdata = V.mat.unal) to obtain a morphological alignment.
#    The fox specimen that is common in V.mat.unal and C.mat.unal is the
#    one in the first row of V.mat.unal (sp.popdata = 1) and the one in position
#    13 in C.mat.unal (sp.data = 13). The method to
#    decompose the estimate shrinkage correlation matrix (internally calculated
#    in this function) is the Cholesky decomposition (method = c("chol")).
#    We do not add ages.
     right    <- c( 11, 22, 13, 19, 15, 20, 24, 5, 7, 2, 9, 26, 29 )
     left     <- c( 10, 21, 12, 17, 14, 18, 23, 4, 6, 1, 8, 25, 28 )
     pairedLM <- cbind( left, right )
     obj.aln <- proc2MCMCtree( data = C.mat.unal, popdata = V.mat.unal, sp.data = 13,
                               sp.popdata = 1, filename = "./seqfile.aln", coords = 3,
                               method = c("chol"), pairedLM = pairedLM )

# B. Use the unaligned, but processed, carnivoran data (data = C.mat.unal) and
#    vulpes data (popdata = V.mat.unal) to obtain a morphological alignment.
#    The fox specimen that is common in V.mat.unal and C.mat.unal is the
#    one in the first row of V.mat.unal (sp.popdata = 1) and the one in position
#    13 in C.mat.unal (sp.data = 13). The method to
#    decompose the estimate shrinkage correlation matrix (internally calculated
#    in this function) is the Cholesky decomposition (method = c("chol")).
#    We add ages.
     ages <- list( sp1  = 13.135, sp2  =  0.0285, sp3  = 11.95,  sp4  = 35.55,
                   sp5  = 25.615, sp6  = 14.785,  sp7  = 28.55,  sp8  =  0,
                   sp9  =  0,     sp10 =  0,      sp11 =  0,     sp12 =  0,
                   sp13 =  0,     sp14 =  0,      sp15 =  0,     sp16 =  0,
                   sp17 =  6.65,  sp18 =  0.0285, sp19 =  0 )
     right    <- c( 11, 22, 13, 19, 15, 20, 24, 5, 7, 2, 9, 26, 29 )
     left     <- c( 10, 21, 12, 17, 14, 18, 23, 4, 6, 1, 8, 25, 28 )
     pairedLM <- cbind( left, right )
     obj.aln <- proc2MCMCtree( data = C.mat.unal, popdata = V.mat.unal, sp.data = 13,
                               sp.popdata = 1, filename = "./seqfile.aln", coords = 3,
                               method = c("chol"), pairedLM = pairedLM, ages = ages )

## End(Not run)
```

`R.sh` *Estimated shrinkage correlation matrix*

Description

Estimated shrinkage correlation matrix obtained after using the `corpcor::cor.shrink` package.

Usage

```
R.sh
```

Format

A matrix of size n x n, where n = 87 (morphological traits, 29 landmarks x 3D coordinates):

n Number of traits for which the correlation values have been calculated, 87

`sim.morpho` *Simulate a continuous morphological alignment*

Description

Simulate a continuous morphological alignment using `rTraitCont` and later allowing to account for population noise and correlation among characters.

Usage

```
sim.morpho(tree, n, c = 0, R, ...)
```

Arguments

`tree`	Phylo, object with a phylogenetic tree (see `rTraitCont`).
`n`	Numeric, number of morphological traits to be simulated.
`c`	(Optional) numeric, vector with variances for the speccies within a population to add as population noise to the simulated morphological traits (see details).
`R`	(Optional) matrix, correlation matrix (see details).
`...`	Further arguments passed to `rTraitCont` (see details).

Details

The function `rTraitCont` simulates continuous traits and can take different parameters to adjust the simulation (e.g. the model, the rate drift, etc.). These parameters are the ones the user can pass to the argument `...` in `sim.morpho`. The default values that `sim.morpho` uses are `model = "BM"`, `sigma = 1`, `ancestor = F`, and `root.value = 0`. For this kind of simulation, `sim.morpho` allows only `ancestor = F`, so please do not change this parameter. In the `rTraitCont` package, the parameter `model` can be `model = BM`, `model = OU`, or a function `model = FUN` provided by the user. Currently, `sim.morpho` supports only the first two.

The parameter `c` contains the population noise, which is used to simulate the noise matrix. Each parameter follows a normal distribution, `x ~ N(0,c)`. If the variances are assumed to be the same for all characters within the species, then the length of `c` is 1 and equals to the value of this variance. If it differs, then a vector of length `n` has to be provided specifying the variance for each of the characters.

The simulated noise is later added to the morphological data previously generated, so we obtain the noisy matrix. If a correlation matrix, `R`, is provided, then it is added to the noisy matrix. See object `sim.R` for an example of its format and `data-raw/sim.R` to understand how it can be generated. Note that the correlation matrix needs to be of class "matrix" and symmetric.

Value

`M`	Matrix with the simulated morphological continuous data accounting for noise and, if provided, for population variance and trait correlation too.

Author(s)

Sandra Alvarez-Carretero and Mario dos Reis

See Also

`write.morpho`

Examples

```
# A) Simulation setup: Simulate a morphological alignment
#    with n = 100 continuous characters for a phylogeny
#    defined in object 'tree', with the default parameters in
#    'sim.morpho' to run 'rTraitCont'.
#    Population noise and character correlation are not considered,
#    i.e. c = 0 and R not provided.

     sim.morpho( tree = sim.tree, n = 100 )


# B) Simulation setup: Simulate a morphological alignment
#    with n = 100 continuous characters for a phylogeny
#    defined in object 'tree', but with different parameters
#    than the default ones in 'sim.morpho' to run
#    'rTraitCont'. Population noise and trait correlation are not
#    considered, i.e. c = 0 and R not provided.
```

```
     sim.morpho( tree  = sim.tree, n = 100,
                         model = "OU", sigma = 0.2, alpha = 2 )


# C) Simulation setup: Simulate a morphological alignment
#    with n = 100 continuous characters for a phylogeny
#    defined in object 'tree', with the default parameters in
#    'sim.morpho' to run 'rTraitCont'.
#    Population noise is low, c = 0.25, but trait correlation is not
#    considered, i.e. R not provided.

     sim.morpho( tree = sim.tree, n = 100, c = 0.25 )


# D) Simulation setup: Simulate a morphological alignment
#    with n = 100 continuous characters for a phylogeny
#    defined in object 'tree', with the default parameters in
#    'sim.morpho' to run 'rTraitCont'.
#    Population noise is low, c = 0.25, and a correlation
#    matrix to simulate trait correlation (rho = 0.50) is provided.

     sim.morpho( tree = sim.tree, n = 100, c = 0.25, R = sim.R )
```

---

sim.pop *Simulate a population matrix*

Description

Simulate a population sample and return a list with (i) a matrix of size `s x n`, `s` specimens and `n` characters, (ii) a vector with the estimated population variances for each character, and (iii) the estimated shrinkage correlation matrix if the true correlation matrix is provided.

Usage

```
sim.pop(psample, n, c, R)
```

Arguments

psample	Numeric, number of specimens the simulated population sample should include.
n	Numeric, number of morphological traits to be simulated.
c	Numeric, vector with the variances for the species within a population (see details).
R	(Optional) matrix, correlation matrix. (see details).

Details

The parameter `c` is the population noise and it is used to sample `n` characters for each of the `psample` specimens from a normal distribution `x ~ N(0,c)`. If the population noise is assumed to be the same for all the characters within the species, then the length of `c` is 1 and equals to the value of this variance. If it differs, then a vector of length `n` has to be provided specifying the variance for each of the characters

If a correlation matrix, `R`, is provided, then it is added to the population matrix. Note that the correlation matrix needs to be of class "matrix" and symmetric. You can take a look at `data-raw/sim.R.R` to follow the commands used to generate this matrix, object `R.sim`, which is used in the examples.

Value

`$P`	Matrix with the simulated population sample
`$var`	Vector with the estimated variances
`$Rsh`	Estimated shrinkage correlation matrix, only returned if `R` is provided

Author(s)

Sandra Alvarez-Carretero and Mario dos Reis

See Also

`sim.morpho`, `write.morpho`

Examples

```
# A) Simulation setup: Simulate a population with
#    psample = 20 specimens, and sample n = 100 characters with
#    a low population noise, c = 0.25.

     sim.pop( psample = 20, n = 100, c = 0.25 )

# B) Simulation setup: Simulate a population with
#    psample = 20 specimens, and sample n = 100 characters with
#    a low population noise, c = 0.25, and a low trait correlation
#    rho = 0.50 (correlation matrix that follows
#    the constant correlation model, i.e. all non-diagonal values
#    equal to rho).

     sim.pop( psample = 20, n = 100, c = 0.25, R = sim.R )
```

`sim.R` *Correlation matrix for simulations*

Description

True correlation matrix simulated to be used in the examples detailed in the `sim.pop()` function. The matrix follows the constant correlation model, hence all values outside the diagonal are rho = 0.50. The size is `p x p`, being n = 100 the number of characters.

Usage

```
sim.R
```

Format

A matrix of size n x n, where n = 100 (morphological traits, the 100 simulated continuous traits):

n Number of simulated continuous traits for which the correlation values have been calculated, 100

`sim.tree` *Simulated 8-species tree*

Description

Simulated 8-species tree of class "phylo". The function `read.tree` was used to generate this object.

Usage

```
sim.tree
```

Format

An object of class "phylo" with a simulated species tree with s = 8 species. It contains the following components:

edge Two-column matrix with 14 rows, where every row is one edge in the tree, the first column is the ancestor node and the second column its daughter node

edge.length A numeric vector with the branch lengths of the tree

Nnode Numeric, the number of (internal) nodes

tip.label A vector with the names of the tips, class "character"

`stepping.stones` *Estimate marginal likelihood by stepping stones*

Description

Estimate the marginal likelihood using the stepping stones method from a sample of `n` power posterior MCMC chains sampled with mcmctree (or bpp).

Usage

```
stepping.stones(mcmcf = "mcmc.txt", betaf = "beta.txt", se = TRUE)
```

Arguments

`mcmcf`	character, mcmc output file name
`betaf`	character, file with beta values
`se`	logical, whether to calculate the standard error

Details

The MCMC samples should be stored in a directory structure created by `make.bfctlf` with `method = "step-stones"`. The function will read the stored log-likelihood values and calculate the log-marginal likelihood.

If `se = TRUE`, an approximation based on the Delta method is used to calculate the standard error (see Xie et al. 2011). Warnings are given if the approximation appears unreliable.

Value

A list with components `logml`, the log-marginal likelihood estimate; `se`, the standard error of the estimate; `mean.logl`, the mean of log-likelihood values sampled for each beta; and `b`, the beta values used.

Author(s)

Mario dos Reis

References

Xie et al. (2011) Improving marginal likelihood estimation for Bayesian phylogenetic model selection. *Systematic Biology*, 60: 150–160.

See Also

`make.bfctlf` to prepare directories and mcmctree or bpp control files to calculate the power posterior.

treeMCMCtree	*Generating a tree file in MCMCtree format when using morphological data*

Description

Outputs a tree file to be used in MCMCtree. The path to this file needs to be written next to the option "treefile = " in the control file for MCMCtree.

Usage

```
treeMCMCtree(tree, aln, filename)
```

Arguments

`tree`	Character, path to the file with the tree topology in Newick format, without branch lengths (see details).
`aln`	Character, path to the alignment file output by `write.morpho` or `proc2MCMCtree`.
`filename`	Character, name for the output file.

Details

If you used `write.morpho` or `proc2MCMCtree` and passed a vector or a list with the ages of the specimens in your alignment, you will see that the names in the output file with this alignment are followed by a "^" and a value. This value is transformed to the age you input within MCMCtree, as if it was a tip date, and used to estimate the divergence times of the species in your phylogeny. This function generates a tree file with the same species names than in the morphological alignment, i.e. with the ages to be used by MCMCtree next to the names of each specimen. Remember to write the path to the tree file next to the "treefile = " option in the control file to run MCMCtree. Remember that the names without "^" followed by the ages have to be the same in the file you pass to parameter "tree" than the ones you have in the alignment file passed to "aln".

Examples

```
# Use the file with the tree topology (no branch lengths) and the
# file with the morphological alignment that are saved in the
# inst/extdata directory to generate a tree file with the
# ages used by MCMCtree included.
# We call the output tree file "treefile.txt".
   tree <- system.file( "extdata", "19s.trees", package = "mcmc3r")
   aln  <- system.file( "extdata", "seqfile.aln", package = "mcmc3r")
   treeMCMCtree( tree = tree, aln = aln, filename = "treefile.txt" )
```

`V` *29 3D landmarks from the skulls of 19 carnivoran specimens after Procrustes analysis*

Description

A matrix containing the 29 3D landmarks collected from the skulls of 21 "Vulpes vulpes" specimens after carrying out a Procrustes analysis (PA). Please take a look at the description in morpho/data-raw/V.R to understand how this object was generated.

Usage

```
V
```

Format

A matrix with s = 21 rows and n = 87 columns (87/3 = 29 landmarks):

s Rows, specimens from which landmarks were collected, 21

n Columns, 87 traits (29 landmarks in 3D) after the PA

`V.arr.unal` *29 3D landmarks from the skulls of 21 Vulpes vulpes specimens before Procrustes analysis*

Description

A 3D array containing the 29 3D landmarks collected from the skulls of 21 "Vulpes vulpes" specimens before carrying out a Procrustes analysis (PA). Please take a look at the description in morpho/data-raw/V.R to understand how this object was generated.

Usage

```
V.arr.unal
```

Format

An array with k = 29 (landmarks), q = 3 (coordinates) and s = 21 (specimens):

k landmark points collected from 21 foxes specimens, 29

q coordinates in 3D or 2D, 3

s number of specimens, 21

`V.mat.unal` *29 3D landmarks from the skulls of 19 carnivoran specimens before Procrustes analysis*

Description

A matrix containing the 29 3D landmarks collected from the skulls of 21 "Vulpes vulpes" specimens before carrying out a Procrustes analysis (PA). Please take a look at the description in morpho/data-raw/V.R to understand how this object was generated.

Usage

```
V.mat.unal
```

Format

A matrix with s = 21 rows and n = 87 columns (87/3 = 29 landmarks):

s Rows, specimens from which landmarks were collected, 21

n Columns, 87 traits (29 landmarks in 3D) after the PA

`V.PS.nov1` *Object of class array output by Morpho after PA*

Description

Object of class array output by Morpho after being aligned to the mean shape of the 19 carnivoran species previously generated (object "C.PS"). Please take a look at the description in morpho/data-raw/V.R to understand how this object was generated.

Usage

```
V.PS.nov1
```

Format

Object procSym

... Check `procSym` for more details

`var.foxes` *Vector with the population variance of Vulpes vulpes*

Description

Vector with 87 within-species variances calculated from the object V. Please take a look at the description in morpho/data-raw/var.foxes.R to understand how this object was generated.

Usage

```
var.foxes
```

Format

A vector with i = 87 variances regarding the "Vulpes vulpes" population:

i Number of variances for the foxes population, 87

`vulpes21x29.raw` *29 3D landmarks from the skulls of 21 Vulpes vulpes specimens*

Description

A dataset containing the 29 3D landmarks collected from the skulls of 19 carnivoran specimens This data.drame consists of a first column with the specimens labels used by MCMCtree and then 87 columns with the landmarks (29 landmarks x 3 coordinates). Please take a look at the description in morpho/data-raw/vulpes21x29.raw.R to understand how this object was generated.

Usage

```
vulpes21x29.raw
```

Format

A data.frame with n = 21 rows and p = 88 columns (info column + 87 coordinates):

n Rows, specimens from which landmarks were collected, 21

p Columns, information about the taxa (1st column) and 87 coordinates (29 landmarks in 3D)

write.morpho *Generate a file with a morphological alignment for MCMCtree*

Description

Generate an alignment file with quantitative characters in MCMCtree format. The option "seqfile" in the control file used by MCMCtree should read the path to the file output by this function.

Usage

```
write.morpho(
  M,
  filename,
  c = 0,
  R = diag(1, dim(M)[2]),
  method,
  A = NULL,
  names,
  ages,
  scale = 1
)
```

Arguments

`M`	Matrix, `s` rows (specimens) and `n` morphological continuous characters (see details).
`filename`	Character, name for the output file.
`c`	Numeric, vector of variances within the species of a population (see details). If not provided, `c = 0` (no population noise).
`R`	Matrix, correlation matrix. Requires `method` (see details). If not provided, `R = I` (no character correlation).
`method`	(Optional) character, either `"eigen"` or `"chol"`, method used to decompose the inverse of the shrinkage correlation matrix. Requires `R` (see details).
`A`	(Optional) matrix, decomposed matrix. Requires `R` but not `method` (see details).
`names`	(Optional) list or character, species name included in the morphological alignment (see examples B and C).
`ages`	(Optional) list or numeric, ages of the species included in the morphological alignment (see example C).
`scale`	numeric, all characters are multiplied by `scale` before the morphological matrix is printed to file. Useful to re-scale the characters so they have the required variance (rate).

Details

The matrix `M` has `s` rows, one for each specimen, and `n` columns regarding the characters. If the data set contains landmarks, they can be given in 2D or 3D. For instance, if the landmarks are 3D, the first 3 columns will be the coordinates x, y, and z for the first landmark; the next 3 columns for the second landmark; and so on.

specimens	lmk1.x	lmk1.y	lmk1.z	lmk2.x	lmk2.y	lmk2.z	...
Sp_1	0.143	-0.028	-0.044	0.129	0.028	-0.043	...
Sp_2	0.128	-0.024	-0.028	0.124	0.027	-0.025	...
...	...	...	...	...	...	...	...

See descriptions in `data-raw/carnivores19x29.raw.R`, `data-raw/vulpes21x29.raw.R`, `data-raw/C.R`, `data-raw/V.R`, to know how to obtain the morphological alignment used in `write.morpho`. Note that the explanation starts with the processing of raw data.

If the data set contains a set of `n` morphological continuous characters, e.g. from a simulated data set, the file should look like

specimens	char.1	char.2	char.3	char.4	...
Sp_1	0.143	-0.028	-0.044	0.129	...
Sp_2	0.128	-0.024	-0.028	0.124	...
...	...	...	...	...	...

Note that if a list with the specimens names is not passed to the parameter `names`, the name for each species will be "Species_1", "Species_2", and so on.

The object `c` can be of length 1, if all characters have the same variance, or a vector of length `n` with the variance of each of the characters. For the latter, you can take a look at object `var.foxes`, which has been generated following the steps explained in `data-raw/var.foxes.R`.

The object `R` has to be a symmetric and positive definite object of class matrix, i.e. `class( R ) = "matrix"`. See `R.sh` for an example of its format. You can also read the description in `data-raw/R.shrinkage.R` for the details about how to generate this matrix.

The logarithm of the determinant of the correlation matrix is going to be printed in the output file to later be used by MCMCtree during the likelihood calculation.

If a correlation matrix `R` is provided, `write.morpho` can use either the `method = "chol"` or `method = "eigen"` to get a matrix `A` such that $\mathrm{R}^{-1} = \mathrm{A}^{\mathrm{T}}\mathrm{A}$. This matrix A^{T} is later used to transform the morphological data to account for the correlation in this data set, so that the transformed characters in `Z`, $\mathrm{Z} = \mathrm{MA}^{\mathrm{T}}$, are independent. Alternatively, this matrix `A` can also be provided by the user. You can read the description to generate this matrix in `data-raw/A.R` to understand how to generate this matrix, which is also available as object `A`. If you decide to use a matrix `A`, it will be used to transform the data and no decomposition will be performed, thus saving computational time when large matrices are to be used.

Author(s)

Sandra Alvarez-Carretero and Mario dos Reis

See Also

matrix2array, array2matrix, sim.morpho

Examples

```
# A.1) Providing the morphological alignment (M = C) and
#      the name for the output file. This does not account for
#      correlation nor population noise

       write.morpho(  M = C, filename = "seqfile.aln" )

# A.2) Providing the morphological alignment (M = C), the population
#      noise (c = 0.25), and the name for the output file. Note that
#      c = 0.25 means that the population noise for all the characters
#      is c = 0.25, i.e. it will be considered as if
#      length( c ) = p characters, being all of them 0.25.

       write.morpho(  M = C, c = 0.25,
                      filename = "seqfile.aln" )

# A.3) Providing the morphological alignment (M = C), the population
#      noise (c = 0.25), the (estimate of the) correlation matrix (R),
#      the method to decompose R ("chol" in this example),
#      and the name for the output file. Note that the R matrix needs
#      to be invertible, otherwise the data will not be able to be
#      transformed accounting for correlation.

       write.morpho(  M = C, c = 0.25, R = R.sh,
                      method = "chol", filename = "seqfile.aln" )

# A.4) Providing the morphological alignment (M = C), a vector with
#      the population noise for each character (c = var.foxes),
#      the (shrinkage estimate of the) correlation matrix (R = R.sh),
#      the method to decompose R ("chol" in this example),
#      and the name for the output file. Note that the matrix passed
#      to argument R needs to be invertible, otherwise the data will
#      not be able to be transformed accounting for correlation.

       write.morpho(  M = C, c = var.foxes, R = R.sh,
                      method = "chol", filename = "seqfile.aln" )

# A.5) Providing the morphological alignment (C), a vector with
#      the population noise for each character (c = var.foxes),
#      the (shrinkage estimate of the) correlation matrix (R = R.sh),
#      the A matrix to transform the data, and the name for the
#      output file. Note that as the A matrix is provided, the matrix
#      passed to R will not be decomposed, hence the argument "method"
#      is not needed.

       write.morpho(  M = C, c = var.foxes, R = R.sh,
                      A = A, filename = "seqfile.aln" )
```

```
# B) Scenario A.5 but providing a list with the
#    names of the species

    names <- list( sp1  = "Ael_sp.", sp2  = "Can_dir", sp3  = "Epi_hay", sp4  = "Hes_sp.",
                  sp5  = "Par_jos", sp6  = "Tom_sp.", sp7  = "Enh_pah", sp8  = "Cuo_alp",
                  sp9  = "Spe_ven", sp10 = "Can_lup", sp11 = "Cer_tho", sp12 = "Oto_meg",
                  sp13 = "Urs_ame", sp14 = "Ail_ful", sp15 = "Nan_bin", sp16 = "Par_her",
                    sp17 = "Hia_won", sp18 = "Smi_fat", sp19 = "Vul_vul"
                  )

     write.morpho( M = C, c = var.foxes, R = R.sh,
                   A = A, filename = "seqfile.aln", names = names )

# C) Scenario A.5 but providing a vector of type character with the names of
#    the specimens and a list with their corresponding ages. Please
#    keep the same order in both lists, so the first specimen in the
#    list name corresponds to the first age in the age list, and so on.

     names <- c( "Ael_sp.", "Can_dir", "Epi_hay", "Hes_sp.",
                 "Par_jos", "Tom_sp.", "Enh_pah", "Cuo_alp",
                 "Spe_ven", "Can_lup", "Cer_tho", "Oto_meg",
                 "Urs_ame", "Ail_ful", "Nan_bin", "Par_her",
                 "Hia_won", "Smi_fat", "Vul_vul"
                  )

     ages <- list( sp1  = 13.135, sp2  =  0.0285, sp3  = 11.95,  sp4  = 35.55,
                   sp5  = 25.615, sp6  = 14.785,  sp7  = 28.55,  sp8  =  0,
                   sp9  =  0,     sp10 =  0,      sp11 =  0,     sp12 =  0,
                   sp13 =  0,     sp14 =  0,      sp15 =  0,     sp16 =  0,
                   sp17 =  6.65,  sp18 =  0.0285, sp19 =  0
                  )

     write.morpho( M = C, c = var.foxes, R = R.sh,
                   A = A, filename = "seqfile.aln",
                   names = names, ages = ages )
```

Index