Package ‘pmc’

March 4, 2013

Version 0.0-8

Date 2012-02-22

Title Phylogenetic Monte Carlo

Author Carl Boettiger <cboettig@gmail.com>

Maintainer Carl Boettiger <cboettig@gmail.com>

Description Monte Carlo based model choice for applied phylogenetics of continuous traits

URL https://github.com/cboettig/pmc

BugReports https://github.com/cboettig/pmc/issues

License CC0

Type Package

Imports ape, geiger, snowfall, TreeSim, methods, ouch, reshape2

Depends ggplot2, ouch

Enhances ouch, geiger, ape

Collate 'format_data.R' 'pmc.R' 'powercurve.R' 'wrappers_geiger.R' 'wrappers_laser.R' 'wrappers_ouch.R' 'wrappers_treepar.R'

NeedsCompilation no

Repository CRAN

Date/Publication 2013-03-04 07:19:19
R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>anoles</td>
<td>2</td>
</tr>
<tr>
<td>convert</td>
<td>2</td>
</tr>
<tr>
<td>fitContinuous_object</td>
<td>3</td>
</tr>
<tr>
<td>format_data</td>
<td>4</td>
</tr>
<tr>
<td>matchformats</td>
<td>4</td>
</tr>
<tr>
<td>mrcaOUCH</td>
<td>5</td>
</tr>
<tr>
<td>paintBranches</td>
<td>5</td>
</tr>
<tr>
<td>plot.pmc</td>
<td>6</td>
</tr>
<tr>
<td>plot_pars</td>
<td>6</td>
</tr>
<tr>
<td>pmc</td>
<td>7</td>
</tr>
<tr>
<td>pmc_fit</td>
<td>8</td>
</tr>
<tr>
<td>simulate.fitContinuous</td>
<td>9</td>
</tr>
<tr>
<td>simulate_and_update</td>
<td>10</td>
</tr>
<tr>
<td>tree</td>
<td>10</td>
</tr>
<tr>
<td>treepalette</td>
<td>11</td>
</tr>
<tr>
<td>treepower</td>
<td>11</td>
</tr>
<tr>
<td>update.fitContinuous</td>
<td>12</td>
</tr>
</tbody>
</table>

Index 13

--

**anoles**

*The anoles data set*

*Description*

as from ouch with additional regimes added and minor formatting changes

--

**convert**

* toggles between ouch and ape format trees

*Description*

toggles between ouch and ape format trees

*Usage*

`convert(ot, regimes = NULL, safe = TRUE)`
Arguments

* tree: a phylogenetic tree in ouch or ape format
* regimes: if given in ape format, are appended to phylo format as phy$regimes. If the ouchtree is a fitted hansen object, regimes will automatically be imported from it unless other regime choice is given here.
* safe: mode for going from ape to ouch. After converting, this writes to a temporary nexus file and reads the tree back in, because phylo format does not have a unique specification for a unique tree, resulting in all kinds of silly problems when developers haven’t been careful. Defaults to true, and will clean up after itself.

Value

a phylogenetic tree in the opposite format

Description

Runs the fitContinuous function and returns and object with necessary information to run methods such as 'update' and 'simulate'. Call is the same as the original fitContinuous function.

Usage

```r
fitContinuous_object(tree, data, model = "BM",
                    bounds = NULL, meserr = NULL)
```

Arguments

* tree: a phylo object phylogenetic tree
* data: ape data phylo format
* model: the type of model to fit
* bounds: optional list of bounds for the parameter fits
* meserr: measurement error, if any

Value

an object of class fitContinuous
format_data
description
format data in ape format into ouch format

Usage

format_data(tree, traits, species_names = NULL, regimes = NULL)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>a phylogenetic tree of class &quot;phylo&quot;, ape format, or a tree in ouch format</td>
</tr>
<tr>
<td>traits</td>
<td>a numeric with trait values, or a matrix or data frame of traits, rownames matching species or handed in</td>
</tr>
<tr>
<td>species_names</td>
<td>in the order of entries in traits, if not given in rownames.</td>
</tr>
<tr>
<td>regimes</td>
<td>the column in traits containing regime labels</td>
</tr>
</tbody>
</table>

Details

Should become an internal function to handle data conversion to ape-type

Value

the ouch-formatted tree, traits, and regimes

See Also

convert to toggle between formats, including regime paintings

matchformats

description
Internal helper function to match data formats

Usage

matchformats(A, B, sim)
Arguments

A    model with simulate & update methods
B    another model with simulate & update methods
sim  simulations produced by model A

Value

A list of two data sets. The first is in a format appropriate to update model A, the second is in the format appropriate to update model B

<table>
<thead>
<tr>
<th>mrcaOUCH</th>
<th>Finds the most recent common ancestor on an OUCH formatted tree</th>
</tr>
</thead>
</table>

Description

Finds the most recent common ancestor on an OUCH formatted tree

Usage

mrcaOUCH(cladeVector, tree)

Arguments

cladeVector  vector of species for which you want to find the most recent common ancestor
tree         an ouchtree class phylogeny

Value

the node number of the most recent common ancestor

<table>
<thead>
<tr>
<th>paintBranches</th>
<th>Paints branches with regimes changing at nodes specified</th>
</tr>
</thead>
</table>

Description

Paints branches with regimes changing at nodes specified

Usage

paintBranches(regimeShiftNodes, tree, regimeTitles = NULL)
Arguments

Arguments

Arguments

Arguments

Arguments

Arguments

Arguments

Arguments

Arguments

Arguments

Arguments

Arguments

Arguments
**pmc**

**Arguments**
- `object` a pmc object fit

**Value**
- a ggplot2 plot object

---

**pmc**

Performs a phylogenetic monte carlo between modelA and modelB

---

**Description**

Simulates data under each model and returns the distribution of likelihood ratio, \( L(B)/L(A) \), under for both simulated datasets.

**Usage**

```r
pmc(tree, data,
modelA = c("BM", "OU", "lambda", "kappa", "delta", "EB", "white", "trend", "hansen", "brown"),
modelB = c("BM", "OU", "lambda", "kappa", "delta", "EB", "white", "trend", "hansen", "brown"),
optionsA = list(), optionsB = list(), nboot = 20, ...)
```

**Arguments**

- `tree` A phylogenetic tree. Can be phylo (ape) or ouch tree
- `data` The data matrix
- `modelA` a model from the list, or a custom model, see details
- `modelB` any other model from the list, or custom model, see details
- `optionsA` a named list of arguments for fitting model A
- `optionsB` a named list of arguments for fitting model B
- `nboot` number of bootstrap replicates to use
- `...` additional arguments

**Details**

Possible models are all models from fitContinuous & ouch

The return value includes all parameters estimated under the structure "par_dists", a data frame with columns "value", (the numerical value of the parameter estimated) "parameter" (a factor indicating the name of the parameter in the model, i.e. lambda), "comparison", (one of AA, AB, BA, or BB, where the first letter indicates the class of model estimated, and the second indicates the model used to simulate the data on which the estimate is based. Hence AA is the bootstrap of model A, and BB the bootstrap of model B. Note that the cross-comparisons can also be informative). Finally "rep" indicates the replicate number for the simulation. See examples for plotting and calculating statistics from this data frame.
pmc_fit

Fit any model used in PMC

Description

The fitting function used by pmc to generalize fitting to any model

Usage

pmc_fit(tree, data, model, options = list())
Arguments

- **tree**: a phylogenetic tree. can be ouch or ape format
- **data**: trait data in ape or ouch format
- **model**: the name of the model to fit, see details for a list of currently supported types
- **options**: whatever additional options would be provided to the model fit, see details

Details

options should include all parameters required by the fit method. Currently methods available are `fitContinuous` (see geiger package) and `hansen` (see the ouch package).

Value

a pmc_model object, anything that has methods "simulate", "update", getParameters, and getLikelihood

Examples

```r
## a geiger example
require(geiger) # just to load the data
data(geospiza)
attach(geospiza)
lambdaFit<-pmc_fit(geospiza.tree, geospiza.data, model="lambda")
## Or a single trait at a time:
lambdaFit<-pmc_fit(geospiza.tree, geospiza.data[1], model="lambda")
## an ouch example
require(ouch) # just for the data,
data(bimac)
tree <- with(bimac, ouchtree(node, ancestor, time/max(time), species))
ou.3 <- pmc_fit(data=log(bimac['size']), tree, model="hansen",
                list(regimes=bimac['OU.3'], sqrt.alpha=1, sigma=1))
```

---

**simulate.fitContinuous**

*simulate method*

Description

simulate method

Usage

```r
## S3 method for class 'fitContinuous'
simulate(object, nsim = 1, 
  seed = NULL, ...)
```
Arguments

- object: a fitContinuous object
- nsim: number of sims (currently always 1)
- seed: an optional seed for the simulations (not implemented)
- ... additional arguments, not implemented for fitContinuous simulations

Details

intended as an internal function, though an available S3 method

Value

simulated dataset

simulate_and_update  Internal helper function Simulates under model A, updates both A and B based on that data

Description

Internal helper function Simulates under model A, updates both A and B based on that data

Usage

simulate_and_update(A, B)

Arguments

- A: model with simulate & update methods
- B: another model with simulate & update methods

Value

a list of with fit of A on data simulated under A, the fit on B on data simulated under A, and the simulated data itself

tree  The phylogeny for the anoles data set

Description

The bimaculus phylogeny, as from the ouch package
treepalette  

**Description**

flexible regime coloring for trees

**Usage**

```r
treepalette(tree, regimes = NULL, 
             colormap = c("rainbow", "heat.colors", "terrain.colors", "topo.colors", "cm.colors", "gray"), 
             custom = NULL, rev = FALSE)
```

**Arguments**

- `tree`: A phylo class or ouch-class tree
- `regimes`: in ouch format, (only needed if not already given in tree)
- `colormap`: a standard colormap color, see the list
- `custom`: a custom colormap
- `rev`: logical, reverse the order of the colormap colors

**Value**

a set of colors than can be passed as edge.color in plot.phylo

---

treepower  

**Description**

compares BM against OU of increasing value of alpha

**Usage**

```r
treepower(tree, nboot = 100, cpu = 2, threshold = 0.95, 
           alpha = seq(0.001, 30, length = 100), 
           method = "hansen")
```

**Arguments**

- `tree`: an ouchtree class phylogeny
- `nboot`: number of bootstraps to use in power simulation
- `cpu`: number of cpu’s available
- `threshold`: to use for the power calculation
- `alpha`: a sequence of alpha values to try
- `method`: currently ignored as only comparisons to OU through hansen are allowed
update.fitContinuous

Details

function can take a while to run all comparisons.

Value

a power object, see examples in the vignette

Description

update a fitContinuous model

Usage

## S3 method for class 'fitContinuous'
update(object, ...)

Arguments

object the output of fitContinuous
...
data on which update will be based

Value

the MLE estimated model for the given data
Index

Topic datasets
  anoles, 2
  tree, 10

Topic data
  anoles, 2
  tree, 10

anoles, 2
convert, 2, 4

fitContinuous_object, 3
format_data, 4

matchformats, 4
mrcaOUCH, 5

paintBranches, 5
plot.pmc, 6
plot_pars, 6
pmc, 7
pmc_fit, 8

simulate.fitContinuous, 9
simulate_and_update, 10

tree, 10
treepalette, 11
treepower, 11

update.fitContinuous, 12