Package ‘RPANDA’

January 9, 2020

Version  1.7
Date  2020-01-01
Type  Package
Title  Phylogenetic ANalyses of Diversification
Depends  R (>= 2.14.2), picante, methods
Suggests  testthat
Imports  ape, pspline, deSolve, igraph, TESS, cluster, fpc, pvclust, corpcor, phytools, mvMORPH (>= 1.1.0), stats, graphics, grDevices, utils, geiger, mvtnorm, Matrix, pracma, glassoFast, Rmpfr, fields, coda, parallel
Encoding  UTF-8
Author  Hélène Morlon [aut, cre, cph],
     Eric Lewitus [aut, cph],
     Fabien Condamine [aut, cph],
     Marc Manceau [aut, cph],
     Julien Clavel [aut, cph],
     Jonathan Drury [aut, cph],
     Olivier Billaud [aut, cph],
     Odile Maliet [aut, cph]
Maintainer  Hélène Morlon <morlon@biologie.ens.fr>
License  GPL-2
URL: https://github.com/hmorlon/PANDA
Repository: CRAN
RoxygenNote: 6.1.1
NeedsCompilation: yes
Date/Publication: 2020-01-09 12:30:05 UTC

R topics documented:

RPANDA-package ........................................ 4
ancestral ................................................. 5
Anolis.data ............................................. 7
Balaenopteridae ....................................... 8
BGB.examples ......................................... 8
BICompare ............................................. 9
Calomys ............................................... 10
Caprimulgidae ...................................... 11
Caprimulgidae_ClaDS2 ................................ 11
Cetacea ............................................... 12
ClaDS0_example ....................................... 13
c02 ..................................................... 14
c02_res ............................................... 15
coccolithophore ..................................... 16
CreateClassObject .................................. 16
CreateGeobyClassObject ............................ 18
CreateGeoObject .................................... 20
CreateGeoObject_BioGeoBEARS ................... 21
createModel .......................................... 23
createModelCoevolution .............................. 23
d13c .................................................... 24
fitTipData ........................................... 26
fitTipData-methods .................................. 27
fit_bd .................................................. 27
fit_ClaDS ............................................. 31
fit_ClaDS0 ........................................... 33
fit_coal_cst ......................................... 35
fit_coal_var .......................................... 37
fit_env ................................................ 39
fit_sgd ............................................... 42
fit_t_comp ........................................... 43
fit_t_comp_subgroup .................................. 45
fit_t_env ............................................. 48
fit_t_pl ............................................... 52
foraminifera ......................................... 54
getDataLikelihood ................................... 55
dataLikelihood-methods ............................. 56
dataLikelihood-methods ......................... 56
getMAPS_ClaDS ....................................... 56
R topics documented:

- getMAPS_ClaDS0 ........................................... 58
- getTipDistribution ...................................... 59
- getTipDistribution-methods ............................ 60
- GIC.fit_pl.rpanda ...................................... 61
- gic_criterion ............................................ 62
- greenalgae .............................................. 64
- InfTemp .................................................. 65
- JSDtree .................................................. 65
- JSDtree_cluster ........................................ 66
- JSDt_cluster ............................................ 67
- landplant ............................................... 69
- likelihood_bd .......................................... 69
- likelihood_coal_cst .................................... 71
- likelihood_coal_var .................................... 72
- likelihood_sgd .......................................... 73
- likelihood_subgroup_model ............................ 74
- likelihood_t_DD ......................................... 77
- likelihood_t_DD_geog ................................... 78
- likelihood_t_env ........................................ 79
- likelihood_t_MC ......................................... 81
- likelihood_t_MC_geog ................................... 82
- lines.fit_t.env ......................................... 84
- modelSelection .......................................... 85
- modelSelection-methods ............................... 86
- MPHiFFT ................................................ 86
- ostracoda ............................................... 87
- PhenotypicACDC-class ................................ 87
- PhenotypicADiag-class ................................ 89
- PhenotypicBM-class .................................... 90
- PhenotypicDD-class .................................... 91
- PhenotypicGMM-class .................................. 92
- PhenotypicModel-class ................................. 93
- PhenotypicOU-class .................................... 94
- PhenotypicPM-class .................................... 96
- Phocoenidae ............................................ 97
- Phyllostomidae ......................................... 97
- Phyllostomidae_genera ................................ 99
- plot.fit_t.env ........................................... 100
- plot_BICmpare .......................................... 102
- plot_ClaDS0_chains .................................... 103
- plot_ClaDS_chains ...................................... 104
- plot_ClaDS_phylo ....................................... 105
- plot_dtt ................................................ 106
- plot_fit_bd .............................................. 107
- plot_fit_env ............................................ 108
- plot_prob_dtt .......................................... 109
- plot_spectR ............................................. 111
Description

Implements macroevolutionary analyses on phylogenetic trees

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>RPANDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>1.5</td>
</tr>
<tr>
<td>Date</td>
<td>2018-01-29</td>
</tr>
<tr>
<td>License</td>
<td>GPL (&gt;= 2)</td>
</tr>
</tbody>
</table>

More information on the RPANDA package and worked examples can be found in Morlon et al. (2016)

Author(s)

Hélène Morlon <morlon@biologie.ens.fr>
Julien Clavel <clavel@biologie.ens.fr>
Fabien Condamine <fabien.condamine@gmail.com>
Jonathan Drury <jonathan.p.drury@gmail.com>
Eric Lewitus <elewitus@hivresearch.org>
Estimation of traits ancestral states.

Description
Reconstruct the ancestral states at the root (and possibly for each nodes) of a phylogenetic tree from models fit obtained using the \texttt{fit_t_XX} functions.

Usage

\texttt{ancestral(object)}
Arguments

object A model fit object obtained by the fit_t_XX class of functions.

Details

ancestral is an S3 method that reconstruct the ancestral states at the root and possibly for each
nodes of a phylogenetic tree from the models fit obtained by the fit_t_XX class of functions (e.g.,
fit_t_pl, fit_t_comp and fit_t_env). Ancestral states are estimated using generalized least
squares (GLS; Martins & Hansen 1997, Cunningham et al. 1998).

Value

a list with the following components

root the reconstructed ancestral states at the root
nodes the reconstructed ancestral states at each nodes (not yet implemented for all the
methods)

Note

The function is used internally in phyl.pca_pl.

Author(s)

J. Clavel

References

Clavel, J., Aristide, L., Morlon, H., 2019. A Penalized Likelihood framework for high-dimensional
phylogenetic comparative methods and an application to new-world monkeys brain evolution. Syst.

Cunningham C.W., Omland K.E., Oakley T.H. 1998. Reconstructing ancestral character states: a

incorporating phylogenetic information into the analysis of interspecific data. Am. Nat. 149:646-
667.

See Also

fit_t_pl, fit_t_env, phyl.pca_pl, GIC, gic_criterion

Examples

require(mvMORPH)
set.seed(1)
n <- 32 # number of species
p <- 31 # number of traits
tree <- pbtree(n=n)  # phylogenetic tree  
R <- Posdef(p)    # a random symmetric matrix (covariance)  

# simulate a dataset  
Y <- mvSIM(tree, model="BM1", nsim=1, param=list(sigma=R))  

# fit a multivariate BM with Penalized likelihood  
fit <- fit_t_pl(Y, tree, model="BM", method="RidgeAlt")  

# Perform the ancestral states reconstruction  
anc <- ancestral(fit)  

# retrieve the scores  
head(anc$nodes)
### BGB.examples

**Examples**

```
data(Anolis.data)  
plot(Anolis.data$phylo)  
print(Anolis.data$data)  
print(Anolis.data$geography.object)
```

<table>
<thead>
<tr>
<th>Balaenopteridae</th>
<th>Balaenopteridae phylogeny</th>
</tr>
</thead>
</table>

**Description**

Ultrametric phylogenetic tree of the 9 extant Balaenopteridae species

**Usage**

```
data(Balaenopteridae)
```

**Details**

This phylogeny was extracted from Steeman et al. Syst Bio 2009 cetacean phylogeny

**References**


**Examples**

```
data(Balaenopteridae)  
print(Balaenopteridae)  
plot(Balaenopteridae)
```

<table>
<thead>
<tr>
<th>BGB.examples</th>
<th>BioGeoBEARS stochastic maps</th>
</tr>
</thead>
</table>

**Description**

Phylogenies and example stochastic maps for Canidae (from an unstratified BioGeoBEARS analysis) and Ochotonidae (from a stratified BioGeoBEARS analysis)

**Usage**

```
data(BGB.examples)
```
References


See Also

CreateGeoObject_BioGeoBEARS

Examples

data(BGB.examples)
par(mfrow=c(1,2))
plot(BGB.examples$Canidae.phylo)
plot(BGB.examples$Ochotonidae.phylo)

BICompare

Identify modalities in a phylogeny

Description

Computes the BIC values for a specified number of modalities in the distance matrix of a phylogenetic tree and that of randomly bifurcating trees; identifies these modalities using k-means clustering.

Usage

BICompare(phylo,t,meth=c("ultrametric"))

Arguments

phylo an object of type 'phylo' (see ape documentation)
t the number of modalities to be tested
meth whether the randomly bifurcating "control" tree should be ultrametric or non-ultrametric

Value

a list with the following components:

BIC_test BIC values for finding t modalities in the distance matrix of a tree and the lowest five percent of 1000 random ("control") trees
clusters a vector specifying which nodes in the tree belong to each of t modalities
BSS/TSS the ratio of between-cluster sum of squares over total sum of squares
**Author(s)**

E Lewitus

**References**

Lewitus, E., Morlon, H., Characterizing and comparing phylogenies from their Laplacian spectrum, bioRxiv doi: http://dx.doi.org/10.1101/026476

**See Also**

plot_BICmpare, spectR, JSDtree

**Examples**

data(Cetacea)

#BICmpare(Cetacea,5)

---

**Calomys**

*Calomys phylogeny*

---

**Description**

Ultrametric phylogenetic tree of 11 of the 13 extant Calomys species

**Usage**

data(Calomys)

**Details**

This phylogeny is from Pigot et al. PloS Biol 2012

**References**


Manceau, M., Lambert, A., Morlon, H. (submitted)

**Examples**

data(Calomys)

print(Calomys)

plot(Calomys)
Caprimulgidae

The Caprimulgidae phylogeny.

Description
The MCC phylogeny for the Caprimulgidae, from Jetz et al. (2012).

Usage
data("Caprimulgidae")

Source

Examples
data("Caprimulgidae")
plot(Caprimulgidae)

Caprimulgidae_ClaDS2
An example run of ClaDS2.

Description
An example of the run on the inference of ClaDS2 on the Caprimulgidae phylogeny, thinned every 10 iterations.

Usage
data("Caprimulgidae_ClaDS2")

Format
A list object with fields:

tree The Caprimulgidae phylogeny on which we ran the model.
sample_fraction The sample fraction for the clade.
sampler The chains obtained by running ClaDS2 on the Caprimulgidae phylogeny.

Details
The Caprimulgidae phylogeny was obtained from Jetz et al. (2012)
Author(s)
O. Maliet

Source

References

See Also
fit_ClaDS, plot_ClaDS_chains, getMAPS_ClaDS

Examples

data("Caprimulgidae_ClaDS2")

# plot the mcmc chains
plot_ClaDS_chains(Caprimulgidae_ClaDS2$sampler)

# extract the Maxima A Posteriori for each parameter
maps = getMAPS_ClaDS(Caprimulgidae_ClaDS2$sampler, thin = 1)
print(paste0("sigma = ", maps[1], " ; alpha = ", maps[2], " ; epsilon = ", maps[3], " ; l_0 = ", maps[4] ))

# plot the inferred branch specific speciation rates
plot_ClaDS_phylo(Caprimulgidae_ClaDS2$tree, maps[-(1:4)])

---

Cetacea

Description
Ultrametric phylogenetic tree for 87 of the 89 extant cetacean species

Usage
data(Cetacea)

Details
This phylogeny was constructed by Bayesian phylogenetic inference from six mitochondrial and nine nuclear genes. It was calibrated using seven paleontological age constraints and a relaxed molecular clock approach. See Steeman et al. (2009) for details.
Source

References
Steeman ME et al. (2009) Radiation of extant cetaceans driven by restructuring of the oceans *Syst Biol* 58:573-585

Examples
```r
data(Cetacea)
print(Cetacea)
plot(Cetacea)
```

---

**ClaDS0_example**  
*An example run of ClaDS0.*

Description
An example of the run on the inference of ClaDS0 on a simulated phylogeny, thinned every 10 iterations.

Usage
```r
data("ClaDS0_example")
```

Format
A list object with fields:
```
tree  The simulated phylogeny on which we ran the model.
speciation_rates  The simulated speciation rates.
Cl0_chains  The output of the run_ClaDS0 run.
```

References

See Also
`fit_ClaDS0`
Examples

data(ClaDS0_example)

# plot the resulting chains for the first 4 parameters
plot_ClaDS0_chains(ClaDS0_example$Cl0_chains, param = 1:4)

# extract the Maximum A Posteriori for each of the parameters
MAPS = getMAPS_ClaDS0(ClaDS0_example$tree,
                       ClaDS0_example$Cl0_chains,
                       thin = 10)

# plot the simulated (on the left) and inferred speciation rates (on the right)
# on the same color scale
plot_ClaDS_phylo(ClaDS0_example$tree,
                 ClaDS0_example$speciation_rates,
                 MAPS[-(1:3)])

co2

co2 data since the Jurassic

Description

Atmospheric co2 data since the Jurassic

Usage

data(co2)

Details

Atmospheric co2 data since the Jurassic taken from Mayhew et al., (2008, 2012) and derived from the GeoCarb-III model (Berner and Kothavala, 2001). The data are reported as the ratio of the mass of co2 at time t to that at present. The format is a dataframe with the two following variables:

age  a numeric vector corresponding to the geological age, in Myrs before the present
co2  a numeric vector corresponding to the estimated co2 at that age

References


co2_res

Examples

data(co2)
plot(co2)

---

co2_res  
*co2 data since the beginning of the Cenozoic*

Description

Atmospheric co2 data since the beginning of the Cenozoic

Usage

data(co2_res)

Details

Implied co2 data since the beginning of the Cenozoic taken from Hansen et al., (2013). The data are the amount of co2 in ppm required to yield observed global temperature throughout the Cenozoic:

- `age` a numeric vector corresponding to the geological age, in Myrs before the present
- `co2` a numeric vector corresponding to the estimated co2 at that age

Source


References

Steeman ME et al.(2009) Radiation of extant cetaceans driven by restructuring of the oceans *Syst Biol* 58:573-585


Examples

data(Cetacea)
print(Cetacea)
plot(Cetacea)
Coccolithophore diversity since the Jurassic

Description
Coccolithophore fossil diversity since the Jurassic

Usage
data(coccolithophore)

Details
Coccolithophore fossil diversity since the Jurassic compiled from the Neptune Database (Lazarus, 1994) and Paleobiology Database (https://paleobiodb.org/). Diversity curves are estimated at the genus level using shareholder quorum subsampling (Alroy, 2010) at two-million-year bins. The format is a dataframe with the two following variables:

- age  a numeric vector corresponding to the geological age, in Myrs before the present
- coccolithophore a numeric vector corresponding to the estimated coccolithophore change at that age

References

Examples
data(coccolithophore)
plot(coccolithophore)

CreateClassObject Create class object

Description
This function returns names of internode intervals, named descendants of each node, and a class object formatted in a way that can be passed to CreateGeobyClassObject

Usage
CreateClassObject(simmap,rnd=5)
CreateClassObject

Arguments

- simmap: stochastic map from `make.simmap` in `phytools`
- rnd: integer indicating the number of decimal places to which times should be rounded (default value is 5) (see `round`)

Details

This function formats the class object so that it can be correctly passed to the numerical integration performed in `fit_t_comp_subgroup`.

Value

A list with the following components:

- `class.object`: a list of matrices specifying the state of each branch during each internode interval (see Details)
- `times`: a vector containing the time since the root of the tree at which nodes or changes in biogeography occur (used internally in other functions)
- `spans`: a vector specifying the distances between times (used internally in other functions)

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com

References


See Also

`fit_t_comp_subgroup`, `CreateGeobyClassObject`

Examples

data(Anolis.data)

# Create a make.simmap object
require(phytools)
geo<-c(rep("cuba",7),rep("hispaniola",9),"puerto_rico")
names(geo)<-Anolis.data$phylo$tip.label
stochastic.map<-phytools::make.simmap(Anolis.data$phylo, geo, model="ER", nsim=1)
CreateClassObject(stochastic.map)
CreateGeobyClassObject

Create merged biogeography-by-class object

Description
Create a merged biogeography-by-class object to be passed to fit_t_comp_subgroup using a stochastic map created from any model in BioGeoBEARS (see documentation in BioGeoBEARS package) and a simmap object from phytools (see documentation in phytools package).

Usage
CreateGeobyClassObject(phylo, simmap, trim.class, ana.events, clado.events, stratified=FALSE, rnd=5)

Arguments

- **phylo**: the object of type 'phylo' (see ape documentation) used to build ancestral range stochastic maps in BioGeoBEARS
- **simmap**: a phylo object created using make.simmap in phytools
- **trim.class**: category in the simmap object that represents the subgroup of interest (see Details and Examples)
- **ana.events**: the "ana.events" table produced in BioGeoBEARS that lists anagenetic events in the stochastic map
- **clado.events**: the "clado.events" table produced in BioGeoBEARS that lists cladogenetic events in the stochastic map
- **stratified**: logical indicating whether the ancestral biogeography stochastic map was built from a stratified analysis in BioGeoBEARS
- **rnd**: an integer value indicating the number of decimals to which values should be rounded in order to reconcile class and geo.objects (default is 5)

Details
This function merges a class object (which reconstructs group membership through time) and a stochastic map of ancestral biogeography (to reconstruct sympatry through time), such that lineages can only interact when they belong to the same subgroup AND are sympatric.

This allows fitting models of competition where only sympatric members of a subgroup can compete (e.g., all lineages that share similar diets or habitats).

This function should be used to format the geography object so that it can be correctly passed to the numerical integration performed in fit_t_comp_subgroup.
Value

Returns a list with the following components:

- **map**: a simmap object with phylogeny trimmed to subgroup of interest (including all branches determined to belong to that subgroup)
- **geography.object**: a list with the following components:
  - **geography.matrix**: a list of matrices specifying both sympatry & group membership (==1) or allopatry and/or non-membership in the focal subgroup (==0) for each species pair for each internode interval (see Details)
  - **times**: a vector containing the time since the root of the tree at which nodes or changes in biogeographyXsubgroup membership occur (used internally in other functions)
  - **spans**: a vector specifying the distances between times (used internally in other functions)

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com

References


See Also

- `fit_t_comp_subgroup`
- `CreateGeoObject_BioGeoBEARS`
- `CreateClassObject`

Examples

data(BGB.examples)

Canidae.phylo<-BGB.examples$Canidae.phylo
names(dummy.group)<-Canidae.phylo$tip.label

Canidae.simmap<-phytools::make.simmap(Canidae.phylo,dummy.group)

#build GeobyClass object with "A" as the focal group

Canidae.geobyclass.object<-CreateGeobyClassObject(phylo=Canidae.phylo, simmap=Canidae.simmap, trim.class="A", ana.events=BGB.examples$Canidae.ana.events, ...
CreateGeoObject

Create biogeography object

Description

This function returns names of internode intervals, named descendants of each node, and a geography object formatted in a way that can be passed to fit_t_comp.

Usage

CreateGeoObject(phylo, map)

Arguments

phylo an object of type 'phylo' (see ape documentation)
map either a matrix modified from phylo$edge or a phylo object created using make.simmap (see Details and Examples)

Details

This function should be used to format the geography object so that it can be correctly passed to the numerical integration performed in fit_t_comp.

The map can either be a matrix formed by specifying the region in which each branch specified by phylo$edge existed, or a stochastic map stored as a phylo object output from make.simmap (see Examples).

Value

a list with the following components:

geography.object a list of matrices specifying sympatry (1) or allopatry (0) for each species pair for each internode interval (see Details)
times a vector containing the time since the root of the tree at which nodes or changes in biogeography occur (used internally in other functions)
spans a vector specifying the distances between times (used internally in other functions)
Create Geo Object BioGeoBEARS

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com

References


See Also

fit_t_comp

Examples

data(Anolis.data)
#Create a geography.object with a modified edge matrix
#First, specify which region each branch belonged to:
Anolis.regions=c(rep("cuba",14),rep("hispaniola",17),"puerto_rico")
Anolis.map<-cbind(Anolis.data$phylo$edge,Anolis.regions)
CreateGeoObject(Anolis.data$phylo,map=Anolis.map)

#Create a geography.object with a make.simmap object
#First, specify which region each branch belonged to:
require(phytools)
geo<-c(rep("cuba",7),rep("hispaniola",9),"puerto_rico")
names(geo)<-Anolis.data$phylo$tip.label
stochastic.map<-phytools::make.simmap(Anolis.data$phylo,geo,model="ER",nsim=1)
CreateGeoObject(Anolis.data$phylo,map=stochastic.map)

CreateGeoObject_BioGeoBEARS

Create biogeography object using a stochastic map from BioGeoBEARS

Description

Create biogeography object using a stochastic map created from any model in BioGeoBEARS (see documentation in BioGeoBEARS package).

Usage

CreateGeoObject_BioGeoBEARS( full.phylo, trimmed.phylo = NULL, ana.events, clado.events, stratified=FALSE)
Arguments

full.phylo  the object of type 'phylo' (see ape documentation) that was used to construct the stochastic map in BioGeoBEARS

trimmed.phylo  if the desired biogeography object excludes some species that were initially included in the stochastic map, this specifies a phylo object for the trimmed set of species

ana.events  the "ana.events" table produced in BioGeoBEARS that lists anagenetic events in the stochastic map

clado.events  the "clado.events" table produced in BioGeoBEARS that lists cladogenetic events in the stochastic map

stratified  logical indicating whether the stochastic map was built from a stratified analysis in BioGeoBEARS

Value

a list with the following components:

geography.object  a list of matrices specifying sympatry (1) or allopatry (0) for each species pair for each internode interval (see Details)

times  a vector containing the time since the root of the tree at which nodes or changes in biogeography occur (used internally in other functions)

spans  a vector specifying the distances between times (used internally in other functions)

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com

References


See Also

fit_t_comp CreateGeoObject

Examples

data(BGB.examples)
createModel

Creation of a PhenotypicModel

Description

Creates an object of class PhenotypicModel, intended to represent a model of trait evolution on a specific tree. Distinct keywords correspond to different models, using one phylogenetic tree.

Usage

createModel(tree, keyword)

Arguments

tree an object of class ‘phylo’ as defined in the R package ‘ape’.

keyword a string specifying the model. Available models include "BM", "BM_from0", "BM_from0_driftless", "OU", "OU_from0", "ACDC", "DD", "PM", "PM_OUless".
**createModelCoevolution**

Creation of a PhenotypicGMM

**Description**

Creates an object of class PhenotypicGMM, a subclass of the class PhenotypicModel intended to represent the Generalist Matching Mutualism model of trait evolution on two specific trees.

**Usage**

createModelCoevolution(tree1, tree2, keyword)

**Arguments**

- **tree1**: an object of class ‘phylo’ as defined in the R package ‘ape’.
- **tree2**: an object of class ‘phylo’ as defined in the R package ‘ape’.
- **keyword**: a string object. Default value "GMM" returns an object of class PhenotypicGMM, which takes advantage of faster distribution computation. Otherwise, a "PhenotypicModel" is returned, and the computation of the tip distribution will take much longer.
Value

an object of class "PhenotypicModel" or "PhenotypicGMM".

Author(s)

M Manceau

References


Examples

#Loading example trees
tree1 <- read.tree(text=newick1)
newick2 <- "((X:1.5,Y:1.5):3,Z:4.5);"
tree2 <- read.tree(text=newick2)

#Creating the model
modelGMM <- createModelCoevolution(tree1, tree2)

#Printing basic or full informations on the model definitions
show(modelGMM)
print(modelGMM)

#Simulates tip trait data
dataGMM <- simulateTipData(modelGMM, c(0,0,5,-5, 1, 1), method=2)

d13c data since the Jurassic

Description

Benthic d13c weathering ratio since the Jurassic

Usage

data(d13c)

Details

Ratio of stable carbon isotopes since the Jurassic calculated by Hannisdal and Peters (2011) and Lazarus et al. (2014) from marine carbonates. The format is a dataframe with the two following variables:

age  a numeric vector corresponding to the geological age, in Myrs before the present
d13c  a numeric vector corresponding to the estimated d13c at that age
References


Examples

data(d13c)
plot(d13c)

fitTipData

Maximum likelihood estimators of a model's parameters

Description

Finds the maximum likelihood estimators of the parameters, returns the likelihood and the inferred parameters.

Usage

fitTipData(object, data, params0, GLSstyle, v)

Arguments

object an object of class 'PhenotypicModel'.
data vector of tip trait data.
params0 vector of parameters used to initialize the optimization algorithm. Default value is NULL, in which case the optimization procedure starts with the vector 'params0' specified within the 'model' object.
GLSstyle boolean specifying the way the mean trait value at the root is estimated. Default value is FALSE in which case the mean at the root is considered as any other parameter. If TRUE, the mean value at the root is estimated with the GLS method, as explained, e.g. in Hansen 1997.
v boolean specifying the verbose mode. Default value : FALSE.

Details

Warning : This function uses the standard R optimizer "optim". It may not always converge well. Please double check the convergence by trying distinct parameter sets for the initialisation.

Value

value A numerical value : the lowest -log( likelihood ) value found during the optimization procedure.
inferredParams The maximum likelihood estimators of the model’s parameters.
convergence An integer code specifying the convergence of the optim function. Please refer to the optim function help files.
Author(s)
M Manceau

References

Examples

#Loading an example tree

#Creating the models
modelBM <- createModel(tree, 'BM')

#Simulating tip traits under the model:
dataBM <- simulateTipData(modelBM, c(0,0,0,1))

#Fitting the model to the data
fitTipData(modelBM, dataBM, v=TRUE)

fitTipData-methods  ~ ~ Methods for Function fitTipData ~ ~

Description

~ ~ Methods for function fitTipData ~ ~

Methods

signature(object = "PhenotypicModel") This is the only method available for this function. Same behaviour for any PhenotypicModel.

fit_bd

Maximum likelihood fit of the general birth-death model

Description

Fits the birth-death model with potentially time-varying rates and potentially missing extant species to a phylogeny, by maximum likelihood. Notations follow Morlon et al. PNAS 2011.
fit_bd(phylo, tot_time, f.lamb, f.mu, lamb_par, mu_par, f = 1,
meth = "Nelder-Mead", cst.lamb = FALSE, cst.mu = FALSE,
expo.lamb = FALSE, expo.mu = FALSE, fix.mu = FALSE,
dt=0, cond = "crown")

Usage

Arguments

- **phylo**
  - an object of type 'phylo' (see ape documentation)
- **tot_time**
  - the age of the phylogeny (crown age, or stem age if known). If working with crown ages, tot_time is given by max(node.age(phylo)$ages).
- **f.lamb**
  - a function specifying the hypothesized functional form (e.g. constant, linear, exponential, etc.) of the variation of the speciation rate $\lambda$ with time. Any functional form may be used. This function has two arguments: the first argument is time; the second argument is a numeric vector of the parameters of the time-variation (to be estimated).
- **f.mu**
  - a function specifying the hypothesized functional form (e.g. constant, linear, exponential, etc.) of the variation of the extinction rate $\mu$ with time. Any functional form may be used. This function has two arguments: the first argument is time; the second argument is a numeric vector of the parameters of the time-variation (to be estimated).
- **lamb_par**
  - a numeric vector of initial values for the parameters of f.lamb to be estimated (these values are used by the optimization algorithm). The length of this vector is used to compute the total number of parameters in the model, so to fit a model with constant speciation rate (for example), lamb_par should be a vector of length 1. Otherwise aic values will be wrong.
- **mu_par**
  - a numeric vector of initial values for the parameters of f.mu to be estimated (these values are used by the optimization algorithm). The length of this vector is used to compute the total number of parameters in the model, so to fit a model without extinction (for example), mu_par should be empty (vector of length 0). Otherwise aic values will be wrong.
- **f**
  - the fraction of extant species included in the phylogeny
- **meth**
  - optimization to use to maximize the likelihood function, see optim for more details.
- **cst.lamb**
  - logical: should be set to TRUE only if f.lamb is constant (i.e. does not depend on time) to use analytical instead of numerical computation in order to reduce computation time.
- **cst.mu**
  - logical: should be set to TRUE only if f.mu is constant (i.e. does not depend on time) to use analytical instead of numerical computation in order to reduce computation time.
- **expo.lamb**
  - logical: should be set to TRUE only if f.lamb is exponential to use analytical instead of numerical computation in order to reduce computation time.
- **expo.mu**
  - logical: should be set to TRUE only if f.mu is exponential to use analytical instead of numerical computation in order to reduce computation time.
- **fix.mu**
  - logical: if set to TRUE, the extinction rate $\mu$ is fixed and will not be optimized.
dt

de the default value is 0. In this case, integrals in the likelihood are computed using R "integrate" function, which can be quite slow. If a positive dt is given as argument, integrals are computed using a piece-wise constant approximation, and dt represents the length of the intervals on which functions are assumed to be constant. For an exponential dependency of the speciation rate with time, we found that dt=1e-3 gives a good trade-off between precision and computation time.

cond

c conditioning to use to fit the model:
  • FALSE: no conditioning (not recommended);
  • "stem": conditioning on the survival of the stem lineage (use when the stem age is known, in this case tot_time should be the stem age);
  • "crown" (default): conditioning on a speciation event at the crown age and survival of the 2 daughter lineages (use when the stem age is not known, in this case tot_time should be the crown age).

Details

The lengths of lamb_par and mu_par are used to compute the total number of parameters in the model, so to fit a model with constant speciation rate (for example), lamb_par should be a vector of length 1. Otherwise aic values will be wrong. In the f.lamb and f.mu functions, the first argument (time) runs from the present to the past. Hence, if the parameter controlling the variation of $\lambda$ with time is estimated to be positive (for example), this means that the speciation rate decreases from past to present.

Value

a list with the following components

model the name of the fitted model
LH the maximum log-likelihood value
aicc the second order Akaike's Information Criterion
lamb_par a numeric vector of estimated f.lamb parameters, in the same order as defined in f.lamb
mu_par a numeric vector of estimated f.mu parameters, in the same order as defined in f.mu (if fix.mu is FALSE)

Author(s)

H Morlon

References


See Also

plot_fit_bd, plot_dtt, likelihood_bd

Examples

# Some examples may take a little bit of time. Be patient!

data(Cetacea)
tot_time<-max(node.age(Cetacea)$ages)

# Fit the pure birth model (no extinction) with a constant speciation rate
f.lamb <-function(t,y){y[1]}
f.mu<-function(t,y){0}
lamb_par<-c(0.09)
mu_par<-c()
result_cst <- fit_bd(Cetacea,tot_time,f.lamb,f.mu,lamb_par,mu_par,
# f=87/89,cst.lamb=TRUE,fix.mu=TRUE,dt=1e-3)
result_cst$model <- "pure birth with constant speciation rate"

# Fit the pure birth model (no extinction) with exponential variation
# of the speciation rate with time
f.lamb <-function(t,y){y[1]*exp(y[2]*t)}
f.mu<-function(t,y){0}
lamb_par<-c(0.05, 0.01)
mu_par<-c()
result_exp <- fit_bd(Cetacea,tot_time,f.lamb,f.mu,lamb_par,mu_par,
# f=87/89,expo.lamb=TRUE,fix.mu=TRUE,dt=1e-3)
result_exp$model <- "pure birth with exponential variation in speciation rate"

# Fit the a pure birth model (no extinction) with linear variation of
# the speciation rate with time
f.lamb <-function(t,y){y[1]+y[2]*t}
f.mu<-function(t,y){0}
lamb_par<-c(0.09, 0.001)
mu_par<-c()
result_lin <- fit_bd(Cetacea,tot_time,f.lamb,f.mu,lamb_par,mu_par,f=87/89,fix.mu=TRUE,dt=1e-3)
result_lin$model <- "pure birth with linear variation in speciation rate"

# Fit a birth-death model with exponential variation of the speciation
# rate with time and constant extinction
f.lamb<-function(t,y){y[1]*exp(y[2]*t)}
f.mu <-function(t,y){y[1]}
lamb_par <- c(0.05, 0.01)
mu_par <-c(0.005)
result_bexp_d cst <- fit_bd(Cetacea,tot_time,f.lamb,f.mu,lamb_par,mu_par,
# f=87/89,expo.lamb=TRUE,cst.mu=TRUE,dt=1e-3)
result_bexp_d cst$model <- "birth-death with exponential variation in speciation rate
# and constant extinction"

# Find the best model
#index <- which.min(c(result_cst$aicc, result_exp$aicc, result_lin$aicc,result_bexp_d cst$aicc))
#rbind(result_cst, result_exp, result_lin, result_bexp_d cst)[index,]
fit_ClaDS

Fit ClaDS to a phylogeny

Description

Perform the inference of branch-specific speciation rates and the model’s hyper parameters for the model with constant extinction rate (ClaDS1) or constant turnover rate (ClaDS2).

Usage

```r
fit_ClaDS(tree, sample_fraction, iterations, thin = 50, file_name = NULL, it_save = 1000, model_id = "ClaDS2", nCPU = 1, mcmcSampler = NULL, ...)
```

Arguments

- `tree` An object of class `phylo`
- `sample_fraction` The sampling fraction for the clade on which the inference is performed.
- `iterations` Number of steps in the MCMC, should be a multiple of `thin`.
- `thin` Number of iterations between two chain state’s recordings.
- `file_name` Name of the file in which the result will be saved. Use `file_name = NULL` (the default) to disable this option.
- `it_save` Number of iteration between each backup of the result in `file_name`.
- `model_id` "ClaDS1" for constant extinction rate, "ClaDS2" (the default) for constant turnover rate.
- `nCPU` The number of CPUs to use. Should be either 1 or 3.
- `mcmcSampler` Optional output of `fit_ClaDS` to continue an already started run.
- `...` Optional arguments, see details.

Details

This function uses a blocked Differential Evolution (DE) MCMC sampler, with sampling from the past of the chains (Ter Braak, 2006; ter Braak and Vrugt, 2008). This sampler is self-adaptive because proposals are generated from the past of the chains. In this sampler, three chains are run simultaneously. Block updates is implemented by first drawing the number of parameters to be updated from a truncated geometric distribution with mean 3, drawing uniformly which parameter to update, and then following the normal DE algorithm.

The available optional arguments are:

- `Nchain` Number of MCMC chains (default to 3).
- `res_ClaDS0` The output of ClaDS0 to use as a startpoint. If `NULL` (the default) a random startpoint is used for the branch-specific speciation rates for each chain.
The starting value for lambda_0 (not used if res_ClaDS0 != NULL).

s0  The starting value for sigma (not used if res_ClaDS0 != NULL).

nlambda  Number of subdivisions for the rate space discretization (use in the likelihood computation). Default to 1000.

nt  Number of subdivisions for the time space discretization (use in the likelihood computation). Default to 30.

Value

A 'list' object with fields:

- post: The posterior function.
- startvalue: The starting value for the MCMC.
- numPars: The number of parameter in the model, including the branch-specific speciation rates.
- Nchain: The number of MCMC chains ran simultaneously.
- currentLPs: The current values of the logposterior for th Nchains chains.
- proposalGenerator: The proposal distribution for the MCMC sampler.
- former: The last output of post for each of the chains.
- thin: Number of iterations between two chain state’s recordings.
- alpha_effect: A vector of size nrow(tree$edge), where the ith element is the number of branches on the path from the crown of the tree and branch i (used internally in other functions).
- consoleupdates: The frequency at which the sampler state should be printed.
- likelihood: The likelihood function, used internally.
- relToAbs: A function mapping the relative changes in speciation rates to the absolute speciation rates for the object phylo, used internally.

Author(s)

O. Maliet

References


**fit_ClaDS0**

Infer ClaDS0’s parameter on a phylogeny

**Description**

Infer branch-specific speciation rates and the model’s hyper parameters for the pure-birth model

**Usage**

```r
fit_ClaDS0(tree, name, pamhLocalName = "pamhLocal",
           iteration = 1e+07, thin = 20000, update = 1000,
           adaptation = 10, seed = NULL, nCPU = 3)
```

**Examples**

```r
data("Caprimulgidae")

sample_fraction = 0.61

sampler = fit_ClaDS(Caprimulgidae, sample_fraction, 1000, thin = 50,
        file_name = NULL, model_id="ClaDS2", nCPU = 1)
plot_ClaDS_chains(sampler)

# continue the same run
sampler = fit_ClaDS(Caprimulgidae, sample_fraction, 50, mcmcSampler = sampler)

# plot the result of the analysis (saved in "Caprimulgidae_ClaDS2", after thinning)
data("Caprimulgidae_ClaDS2")

# plot the mcmc chains
plot_ClaDS_chains(Caprimulgidae_ClaDS2$sampler)

# extract the Maxima A Posteriori for each parameter
maps = getMAPS_ClaDS(Caprimulgidae_ClaDS2$sampler, thin = 1)
print(paste0("sigma = ", maps[1], " ; alpha = ",
               maps[2], " ; epsilon = ", maps[3], " ; l_0 = ", maps[4] ))

# plot the inferred branch specific speciation rates
plot_ClaDS_phylo(Caprimulgidae_ClaDS2$tree, maps[-(1:4)])
```
**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tree</td>
<td>An object of class 'phylo'.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the file in which the results will be saved. Use name = NULL to disable this option.</td>
</tr>
<tr>
<td>pamhLocalName</td>
<td>The function is writing in a text file to make the execution quicker, this is the name of this file.</td>
</tr>
<tr>
<td>iteration</td>
<td>Number of iterations after which the gelman factor is computed and printed. The function stops if it is below 1.05</td>
</tr>
<tr>
<td>thin</td>
<td>Number of iterations between two chain state’s recordings.</td>
</tr>
<tr>
<td>update</td>
<td>Number of iterations between two adjustments of the proposal parameters during the adaptation phase of the sampler.</td>
</tr>
<tr>
<td>adaptation</td>
<td>Number of times the proposal is adjusted during the adaptation phase of the sampler.</td>
</tr>
<tr>
<td>seed</td>
<td>An optional seed for the MCMC run.</td>
</tr>
<tr>
<td>nCPU</td>
<td>The number of CPUs to use. Should be either 1 or 3.</td>
</tr>
</tbody>
</table>

**Details**

This function uses a Metropolis within Gibbs MCMC sampler with a bactrian proposal (ref) with an initial adaptation phase. During this phase, the proposal is adjusted "adaptation" times every "update" iterations to reach a goal acceptance rate of 0.3.

To monitor convergence, 3 independant MCMC chains are run simultaneously and the Gelman statistics is computed every "iteration" iterations. The inference is stopped when the maximum of the one dimentional Gelman statistics (computed for each of the parameters) is below 1.05.

**Value**

A mcmc.list object with the three MCMC chains.

**Author(s)**

O. Maliet

**References**


**See Also**

getMAPS_ClaDS0, plot_ClaDS0_chains, fit_ClaDS
Examples

```r
set.seed(1)

obj = sim_ClaDS(
  lambda_0=0.1,
  mu_0=0.5,
  sigma_lamb=0.7,
  alpha_lamb=0.90,
  condition="taxa",
  taxa_stop = 20,
  prune_extinct = TRUE)

tree = obj$tree
speciation_rates = obj$lamb[as.numeric(obj$rates)]
extinction_rates = obj$mu[as.numeric(obj$rates)]

plot_ClaDS_phylo(tree, speciation_rates)

sampler = fit_ClaDS0(tree = tree,
  name="ClaDS0_example.Rdata",
  nCPU=1,
  pamhLocalName = "local",
  iteration=500000,
  thin=2000,
  update=1000, adaptation=5)

# extract the Maximum A Posteriori for each of the parameters
MAPS = getMAPS_ClaDS0(tree, sampler, thin = 10)

# plot the simulated (on the left) and inferred speciation rates (on the right)
# on the same color scale
plot_ClaDS_phylo(tree, speciation_rates, MAPS[-(1:3)])
```

---

**fit_coal_cst**

*Maximum likelihood fit of the equilibrium model*

**Description**

Fits the equilibrium diversity model with potentially time-varying turnover rate and potentially missing extant species to a phylogeny, by maximum likelihood. The implementation allows only exponential time variation of the turnover rate, although this could be modified using expressions in Morlon et al. PLoSB 2010. Notations follow Morlon et al. PLoSB 2010.

**Usage**

```r
fit_coal_cst(phylo, tau0 = 1e-2, gamma = 1, cst.rate = FALSE,
  meth = "Nelder-Mead", N0 = 0)
```
Arguments

- **phylo**: an object of type ‘phylo’ (see ape documentation)
- **tau0**: initial value of the turnover rate at present (used by the optimization algorithm)
- **gamma**: initial value of the parameter controlling the exponential variation in turnover rate (used by the optimization algorithm)
- **cst.rate**: logical: should be set to TRUE to fit an equilibrium diversity model with time-constant turnover rate (known as the Hey model, model 1 in Morlon et al. PLoSB 2010). By default, a model with exponential time-varying rate exponential is fitted (model 2 in Morlon et al. PLoSB 2010).
- **meth**: optimization to use to maximize the likelihood function, see `optim` for more details.
- **N0**: Number of extant species. With default value(0), N0 is set to the number of tips in the phylogeny. That is, the phylogeny is assumed to be 100% complete.

Details

This function fits models 1 (when cst.rate=TRUE) and 2 (when cst.rate=FALSE) from the PLoSB 2010 paper. Likelihoods arising from these models are directly comparable to likelihoods from the `fit_coal_var` function, thus allowing to test support for equilibrium versus expanding diversity scenarios. Time runs from the present to the past. Hence, if gamma is estimated to be positive (for example), this means that the speciation rate decreases from past to present.

Value

- a list with the following components
  - **model**: the name of the fitted model
  - **LH**: the maximum log-likelihood value
  - **aicc**: the second order Akaike’s Information Criterion
  - **tau0**: the estimated turnover rate at present
  - **gamma**: the estimated parameter controlling the exponential variation in turnover rate (if cst.rate is FALSE)

Author(s)

H Morlon

References

**See Also**

likelihood_coal_cst, fit_coal_var

**Examples**

```r
data(Cetacea)
result <- fit_coal_cst(Cetacea, tau0=1.e-3, gamma=-1, cst.rate=FALSE, N0=89)
print(result)
```

---

**fit_coal_var**  
Fit birth-death model using a coalescent approach

**Description**

Fits the expanding diversity model with potentially time-varying rates and potentially missing extant species to a phylogeny, by maximum likelihood. The implementation allows only exponential time variation of the speciation and extinction rates, although this could be modified using expressions in Morlon et al. PloSB 2010. Notations follow Morlon et al. PLoSB 2010.

**Usage**

```r
fit_coal_var(phylo, lamb0 = 0.1, alpha = 1, mu0 = 0.01, beta = 0, meth = "Nelder-Mead", N0 = 0, cst.lamb = FALSE, cst.mu = FALSE, fix.eps = FALSE, mu.0 = FALSE, pos = TRUE)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phylo</td>
<td>an object of type ‘phylo’ (see ape documentation)</td>
</tr>
<tr>
<td>lamb0</td>
<td>initial value of the speciation rate at present (used by the optimization algorithm)</td>
</tr>
<tr>
<td>alpha</td>
<td>initial value of the parameter controlling the exponential variation in speciation rate (used by the optimization algorithm)</td>
</tr>
<tr>
<td>mu0</td>
<td>initial value of the extinction rate at present (used by the optimization algorithm)</td>
</tr>
<tr>
<td>beta</td>
<td>initial value of the parameter controlling the exponential variation in extinction rate.</td>
</tr>
<tr>
<td>meth</td>
<td>optimization to use to maximize the likelihood function, see <code>optim</code> for more details.</td>
</tr>
<tr>
<td>N0</td>
<td>Number of extant species. With default value(0), N0 is set to the number of tips in the phylogeny. That is, the phylogeny is assumed to be 100% complete.</td>
</tr>
<tr>
<td>cst.lamb</td>
<td>logical: should be set to TRUE only if f.lamb is constant (i.e. does not depend on time, models 3, 4b &amp; 5 in Morlon et al. PloSB 2010) to use analytical instead of numerical computation in order to reduce computation time.</td>
</tr>
<tr>
<td>cst.mu</td>
<td>logical: should be set to TRUE only if f.mu is constant (i.e. does not depend on time, models 3 &amp; 4a in Morlon et al. PloSB 2010) to use analytical instead of numerical computation in order to reduce computation time.</td>
</tr>
</tbody>
</table>
**Details**

The function fits models 3 to 6 from the PloSB 2010 paper. Likelihoods arising from these models are computed using the coalescent approximation and are directly comparable to likelihoods from the `fit_coal_cst` function, thus allowing to test support for equilibrium versus expanding diversity scenarios.

These models can be fitted using the options specified below:

- **model 3:**
  
  with `cst.lamb=TRUE & cst.mu=TRUE`

- **model 4a:**
  
  with `cst.lamb=FALSE & cst.mu=TRUE`

- **model 4b:**
  
  with `cst.lamb=TRUE & cst.mu=FALSE`

- **model 4c:**
  
  with `cst.lamb=FALSE, cst.mu=FALSE & fix.eps=TRUE`

- **model 4d:**
  
  with `cst.lamb=FALSE, cst.mu=FALSE & fix.eps=FALSE`

- **model 5:**
  
  with `cst.lamb=TRUE & mu.0=TRUE`

- **model 6:**
  
  with `cst.lamb=FALSE & mu.0=TRUE`

Time runs from the present to the past. Hence, if alpha is estimated to be positive (for example), this means that the speciation rate decreases from past to present.

**Value**

a list with the following components

- `model` the name of the fitted model
- `LH` the maximum log-likelihood value
- `aic` the second order Akaike’s Information Criterion
- `model.parameters` the estimated parameters

**Author(s)**

H Morlon
References


See Also

likelihood_coal_var, fit_coal_cst

Examples

data(Cetacea)
result <- fit_coal_var(Cetacea, lamb0=0.01, alpha=-0.001, mu0=0.0, beta=0, N0=89)
print(result)

fit_env

Maximum likelihood fit of the environmental birth-death model

Description

Fits the environmental birth-death model with potentially missing extant species to a phylogeny, by maximum likelihood. Notations follow Morlon et al. PNAS 2011 and Condamine et al. ELE 2013.

Usage

fit_env(phylo, env_data, tot_time, f.lamb, f.mu, lamb_par, mu_par, df= NULL, f = 1, meth = "Nelder-Mead", cst.lamb = FALSE, cst.mu = FALSE, expo.lamb = FALSE, expo.mu = FALSE, fix.mu = FALSE, dt=0, cond = "crown")

Arguments

phylo an object of type 'phylo' (see ape documentation)
env_data environmental data, given as a data frame with two columns. The first column is time, the second column is the environmental data (temperature for instance).
tot_time the age of the phylogeny (crown age, or stem age if known). If working with crown ages, tot_time is given by max(node.age(phylo)$ages).
f.lamb a function specifying the hypothesized functional form of the variation of the speciation rate \( \lambda \) with time and the environmental variable. Any functional form may be used. This function has three arguments: the first argument is time; the second argument is the environmental variable; the third argument is a numeric vector of the parameters controlling the time and environmental variation (to be estimated).
fit_env

f.mu a function specifying the hypothesized functional form of the variation of the extinction rate \( \mu \) with time and the environmental variable. Any functional form may be used. This function has three arguments: the first argument is time; the second argument is the environmental variable; the second argument is a numeric vector of the parameters controlling the time and environmental variation (to be estimated).

lamb_par a numeric vector of initial values for the parameters of f.lamb to be estimated (these values are used by the optimization algorithm). The length of this vector is used to compute the total number of parameters in the model, so to fit a model with constant speciation rate (for example), lamb_par should be a vector of length 1. Otherwise aic values will be wrong.

mu_par a numeric vector of initial values for the parameters of f.mu to be estimated (these values are used by the optimization algorithm). The length of this vector is used to compute the total number of parameters in the model, so to fit a model without extinction (for example), mu_par should be empty (vector of length 0). Otherwise aic values will be wrong.

df the degree of freedom to use to define the spline. As a default, smooth.spline(env_data[,1], env_data[,2])$df is used. See sm.spline for details.

f the fraction of extant species included in the phylogeny

meth optimization to use to maximize the likelihood function, see optim for more details.

cst.lamb logical: should be set to TRUE only if f.lamb is constant (i.e. does not depend on time or the environmental variable) to use analytical instead of numerical computation in order to reduce computation time.

cst.mu logical: should be set to TRUE only if f.mu is constant (i.e. does not depend on time or the environmental variable) to use analytical instead of numerical computation in order to reduce computation time.

expo.lamb logical: should be set to TRUE only if f.lamb is an exponential function of time (and does not depend on the environmental variable) to use analytical instead of numerical computation in order to reduce computation time.

expo.mu logical: should be set to TRUE only if f.mu is an exponential function of time (and does not depend on the environmental variable) to use analytical instead of numerical computation in order to reduce computation time.

fix.mu logical: if set to TRUE, the extinction rate \( \mu \) is fixed and will not be optimized.

dt the default value is 0. In this case, integrals in the likelihood are computed using R "integrate" function, which can be quite slow. If a positive dt is given as argument, integrals are computed using a piece-wise constant approximation, and dt represents the length of the intervals on which functions are assumed to be constant. We found that 1e-3 generally provides a good trade-off between precision and computation time.

cond conditioning to use to fit the model:

- FALSE: no conditioning (not recommended);
- "stem": conditioning on the survival of the stem lineage (use when the stem age is known, in this case tot_time should be the stem age);
• "crown" (default): conditioning on a speciation event at the crown age and survival of the 2 daughter lineages (use when the stem age is not known, in this case tot_time should be the crown age).

Details
The lengths of lamb_par and mu_par are used to compute the total number of parameters in the model, so to fit a model with constant speciation rate (for example), lamb_par should be a vector of length 1. Otherwise aic values will be wrong. In the f.lamb and f.mu functions, time runs from the present to the past.

Value
a list with the following components

- **model**: the name of the fitted model
- **LH**: the maximum log-likelihood value
- **aicc**: the second order Akaike's Information Criterion
- **lamb_par**: a numeric vector of estimated f.lamb parameters, in the same order as defined in f.lamb
- **mu_par**: a numeric vector of estimated f.mu parameters, in the same order as defined in f.mu (if fix.mu is FALSE)

Note
The speed of convergence of the fit might depend on the degree of freedom chosen to define the spline.

Author(s)
H Morlon and F Condamine

References

See Also
[plot_fit_env, fit_bd, likelihood_bd]
Examples

data(Cetacea)
tot_time<-max(node.age(Cetacea)$ages)
data(InfTemp)
dof<-smooth.spline(InfTemp[,1], InfTemp[,2])$df

# Fits a model with lambda varying as an exponential function of temperature
# and mu fixed to 0 (no extinction). Here t stands for time and x for temperature.
f.lamb <-function(t,x,y){y[1] * exp(y[2] * x)}
f.mu<-function(t,x,y){0}
lamb_par<-c(0.10, 0.01)
mu_par<-c()
result_exp <- fit_env(Cetacea,InfTemp,tot_time,f.lamb,f.mu,lamb_par,mu_par,
  # f=87/89,fix.mu=TRUE,df=dof,dt=1e-3)

Description

Fits the SGD model with exponential growth of the metacommunity, by maximum likelihood. Notations follow Manceau et al. (2015)

Usage

fit_sgd(phylo, tot_time, par, f=1, meth = "Nelder-Mead")

Arguments

phylo an object of type 'phylo' (see ape documentation)
tot_time the age of the phylogeny (crown age, or stem age if known). If working with crown ages, tot_time is given by max(node.age(phylo)$ages)
par a numeric vector of initial values for the parameters (b,d,nu) to be estimated (these values are used by the optimization algorithm)
f the fraction of extant species included in the phylogeny
meth optimization to use to maximize the likelihood function, see optim for more details.

Value

a list with the following components

model the name of the fitted model
LH the maximum log-likelihood value
aicc the second order Akaike’s Information Criterion
par a numeric vector of estimated values of b (birth), b-d (growth) and nu (mutation)
**Note**

While b-d and nu can in general be well estimated, the likelihood surface is quite flat with respect to b, such that the estimated b can vary a lot depending on the choice of the initial parameter values. Estimates of b should not be trusted.

**Author(s)**

M Manceau

**References**


**See Also**

likelihood_sgd

**Examples**

```r
# Some examples may take a little bit of time. Be patient!
data(Calomys)
tot_time <- max(node.age(Calomys)$ages)
par_init <- c(1e7, 1e7-0.5, 1)
#fit_sgd(Calomys, tot_time, par_init, f=11/13)
```

---

**Description**

Fits matching competition (MC), diversity dependent linear (DDlin), or diversity dependent exponential (DDexp) models of trait evolution to a given dataset and phylogeny.

**Usage**

```r
fit_t_comp(phylo, data, model=c("MC","DDexp","DDlin"), pars=NULL, geography.object=NULL)
```

**Arguments**

- **phylo** an object of type 'phylo' (see ape documentation)
- **data** a named vector of trait values with names matching phylo$tip.label
- **model** model chosen to fit trait data, "MC" is the matching competition model of Nuismer & Harmon 2014, "DDlin" is the diversity-dependent linear model, and "DDexp" is the diversity-dependent exponential model of Weir & Mursleen 2013.
- **geography.object** a list of sympatry through time created using CreateGeoObject
- **pars** vector specifying starting parameter values for maximum likelihood optimization. If unspecified, default values are used (see Details)
Details

If unspecified, `par` takes the default values of \( \text{var(data)}/\max(\text{nodeHeights(phylo)}) \) for `sig2` and 0 for either `S` for the matching competition model, `b` for the linear diversity dependence model, or `r` for the exponential diversity dependence model. Values can be manually entered as a vector with the first element equal to the desired starting value for `sig2` and the second value equal to the desired starting value for either `S`, `b`, or `r`. Note: since likelihood optimization uses `sig` rather than `sig2`, and since the starting value for `r` is exponentiated to stabilize the likelihood search, if you input a `par` value, the first value specifying `sig2` should be the `log(sqrt())` of the desired `sig2` starting value.

Value

a list with the following elements:

- `LH` maximum log-likelihood value
- `aic` Akaike Information Criterion value
- `aicc` AIC value corrected for small sample size
- `free.parameters` number of free parameters from the model
- `sig2` maximum-likelihood estimate of `sig2` parameter
- `S` maximum-likelihood estimate of `S` parameter of matching competition model (see Note)
- `b` maximum-likelihood estimate of `b` parameter of linear diversity dependence model
- `r` maximum-likelihood estimate of `r` parameter of exponential diversity dependence model
- `z0` maximum-likelihood estimate of `z0`, the value at the root of the tree
- `convergence` convergence diagnostics from `optim` function (see `optim` documentation)

Note

In current version, the `S` parameter is restricted to take on negative values in `MC + geography ML` optimization.

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com
Julien Clavel

References


See Also

`sim_t_comp`, `CreateGeoObject`, `likelihood_t_MC`, `likelihood_t_MC_geog`, `likelihood_t_DD`, `likelihood_t_DD_geog`

Examples

```r
data(Anolis.data)
geography.object<-Anolis.data$geography.object
pPC1<-Anolis.data$data
phylo<-Anolis.data$phylo

# Fit three models without biogeography to pPC1 data
MC.fit<-fit_t_comp(phylo, pPC1, model="MC")
DDlin.fit<-fit_t_comp(phylo, pPC1, model="DDlin")
DDexp.fit<-fit_t_comp(phylo, pPC1, model="DDexp")

# Now fit models that incorporate biogeography, NOTE these models take longer to fit
MC.geo.fit<-fit_t_comp(phylo, pPC1, model="MC", geography.object=geography.object)
DDlin.geo.fit<-fit_t_comp(phylo, pPC1, model="DDlin", geography.object=geography.object)
DDexp.geo.fit<-fit_t_comp(phylo, pPC1, model="DDexp", geography.object=geography.object)
```

Description

Fits matching competition (MC), diversity dependent linear (DDlin), or diversity dependent exponential (DDexp) models of trait evolution to a given dataset, phylogeny, and stochastic maps of both subgroup membership and biogeography.

Usage

```r
fit_t_comp_subgroup(full.phylo, ana.events, clado.events, stratified=FALSE, map=data,
trim.class, model=c("MC", "DDexp", "DDlin"), par=NULL, method="Nelder-Mead",
bounds=NULL)
```

Arguments

- `full.phylo`: an object of type 'phylo' (see ape documentation) containing all of the tips used to estimate ancestral biogeography in BioGeoBEARS
- `ana.events`: the "ana.events" table produced in BioGeoBEARS that lists anagenetic events in the stochastic map
clado.events  the "clado.events" table produced in BioGeoBEARS that lists cladogenetic events in the stochastic map
stratified   logical indicating whether the stochastic map was built from a stratified analysis in BioGeoBEARS
map         a phylo object created using make.simmap in phytools that contains reconstructed subgroup membership
data         a named vector of trait values for subgroup members with names matching full.phylo$tip.label
trim.class  subgroup whose members are competing
model       model chosen to fit trait data, "MC" is the matching competition model of Nuismer & Harmon 2014, "DDlin" is the diversity-dependent linear model, and "DDexp" is the diversity-dependent exponential model of Weir & Mursleen 2013.
par          vector specifying starting parameter values for maximum likelihood optimization. If unspecified, default values are used (see Details)
method      optimization algorithm to use (see optim)
bounds      (optional) list of bounds to pass to optimization algorithm (see details at optim)

Details

If unspecified, par takes the default values of var(data)/max(nodeHeights(phylo)) for sig2 and 0 for either S for the matching competition model, b for the linear diversity dependence model, or r for the exponential diversity dependence model. Values can be manually entered as a vector with the first element equal to the desired starting value for sig2 and the second value equal to the desired starting value for either S, b, or r. Note: since likelihood optimization uses sig rather than sig2, and since the starting value for sig is exponentiated to stabilize the likelihood search, if you input a par value, the first value specifying sig2 should be the log(sqrt()) of the desired sig2 starting value.

Value

a list with the following elements:
LH             maximum log-likelihood value
aic            Akaake Information Criterion value
aicc           AIC value corrected for small sample size
free.parameters number of free parameters from the model
sig2           maximum-likelihood estimate of sig2 parameter
S              maximum-likelihood estimate of S parameter of matching competition model (see Note)
b              maximum-likelihood estimate of b parameter of linear diversity dependence model
r              maximum-likelihood estimate of r parameter of exponential diversity dependence model
z0             maximum-likelihood estimate of z0, the value at the root of the tree
convergence    convergence diagnostics from optim function (see optim documentation)
Note

In current version, the S parameter is restricted to take on negative values in MC + geography ML optimization.

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com

References


See Also

*likelihood_subgroup_model CreateGeobyClassObject*

Examples

data(BGB.examples)

Canidae.phylo<-BGB.examples$Canidae.phylo
names(dummy.group)<-Canidae.phylo$tip.label

Canidae.simmap<-phytools::make.simmap(Canidae.phylo,dummy.group)

set.seed(123)
Canidae.data<-rnorm(length(Canidae.phylo$tip.label))
names(Canidae.data)<-Canidae.phylo$tip.label
Canidae.A<-Canidae.data[which(dummy.group=="A")]

fitA<-fit_t_comp_subgroup(full.phylo=Canidae.phylo,ana.events=BGB.examples$Canidae.ana.events, clado.events=BGB.examples$Canidae.clado.events,stratified=FALSE,map=Canidae.simmap, data=Canidae.A,trim.class="A",model="DDexp")
Maximum likelihood fit of the environmental model of trait evolution

Description

Fits model of trait evolution for which evolutionary rates depends on an environmental function, or more generally a time varying function.

Usage

```r
fit_t_env(phylo, data, env_data, error=NULL, model=c("EnvExp", "EnvLin"), method="Nelder-Mead", control=list(maxit=20000), ...)
```

Arguments

- `phylo`: An object of class ‘phylo’ (see ape documentation)
- `data`: A named vector of phenotypic trait values.
- `env_data`: Environmental data, given as a time continuous function (see, e.g. splinefun) or a data frame with two columns. The first column is time, the second column is the environmental data (temperature for instance).
- `error`: A named vector with standard error (SE) for each species. Default is NULL, if NA, then the SE is estimated from the data.
- `model`: The model describing the functional form of variation of the evolutionary rate $\sigma^2$ with time and the environmental variable. Default models are "EnvExp" and "EnvLin" (see details). An user defined function of any functional form may be used (forward in time). This function has three arguments: the first argument is time; the second argument is the environmental variable; the third argument is a numeric vector of the parameters controlling the time and environmental variation (to be estimated). See the example below.
- `method`: Methods used by the optimization routine (see ?optim for details).
- `control`: Max. bound for the number of iteration of the optimizer; other options can be fixed on the list (see ?optim).
- `...`: Arguments to be passed to the function. See details.

Details

`fit_t_env` allows fitting environmental models of trait evolution. The default models `EnvExp` and `EnvLin` represents models for which the evolutionary rates are changing as a function of environmental changes though times as defined below.

**EnvExp:**

$$\sigma^2(t) = \sigma^2_0 e^{(\beta T(t))}$$

**EnvLin:**

$$\sigma^2(t) = \sigma^2_0 + \beta T(t)$$
Users defined models should have the following form (see also examples below):

```r
fun <- function(t, env, param){ param*env(t)}
```

- **t**: is the time parameter.
- **env**: is a time function of an environmental variable. See for instance object created by `splinefun` when interpolating coordinate of points.
- **param**: is a vector of parameters to estimate.

For instance, the `EnvExp` function can be coded as:

```r
fun <- function(t, env, param){ param[1]*exp(param[2]*env(t))}
```

where `param[1]` is the $\sigma^2$ parameter and `param[2]` is the $\beta$ parameter. Note that in this later case, two starting values should be provided in the `param` argument.

**e.g.**:

```r
sigma=0.1
beta=0
```

```r
fit_t_env(tree, data, env_data=InfTemp, model=fun, param=c(sigma, beta))
```

The various options are passed through "...".

- **param**: The starting values used for the model. Must match the total number of parameters of the specified models. If "error=NA", a starting value for the SE to be estimated must be provided with user-defined models.
- **scale**: scale the amplitude of the environmental curve between 0 and 1. This may improve the parameters search in some situations.
- **df**: the degree of freedom to use for defining the spline. As a default, smooth.spline(env_data[,1], env_data[,2])$df is used. See `sm.spline` for details.
- **upper**: the upper bound for the parameter search when the "L-BFGS-B" method is used. See `optim` for details.
- **lower**: the lower bound for the parameter search when the "L-BFGS-B" method is used. See `optim` for details.
- **sig2**: can be used instead of `param` to define the starting sigma value only
- **beta**: can be used instead of `param` to define the beta starting value only
- **maxdiff**: difference in time between tips and present day for phylogenetic trees with no contemporaneous species (default is 0)

**Value**

- a list with the following components
  - **LH**: the maximum log-likelihood value
  - **aic**: the Akaike's Information Criterion
  - **aicc**: the second order Akaike's Information Criterion
  - **free.parameters**: the number of estimated parameters
param a numeric vector of estimated parameters, sigma and beta respectively for the defaults models. In the same order as defined by the user if a customized model is provided
root the estimated root value
convergence convergence status of the optimizing function; "0" indicates convergence (See ?optim for details)
hess.value reliability of the likelihood estimates calculated through the eigen-decomposition of the hessian matrix. "0" means that a reliable estimate has been reached
env.func the environmental function
tot.time the root age of the tree
model the fitted model (default models or user specified)
SE the estimated SE for species mean when "error=NA"

Note
The users defined function is evaluated forward in time i.e.: from the root to the tips (time = 0 at the (present) tips). The speed of convergence of the fit might depend on the degree of freedom chosen to define the spline.

Author(s)
J. Clavel

References

See Also
plot.fit_t.env, likelihood_t_env

Examples
data(Cetacea)
data(InfTemp)

# Simulate a trait with temperature dependence on the Cetacean tree
set.seed(123)

trait <- sim_t_env(Cetacea, param=c(0.1,-0.2), env_data=InfTemp, model="EnvExp", root.value=0, step=0.001, plot=TRUE)

## Fit the Environmental-exponential model
# Fit the environmental model
result1=fit_t_env(Cetacea, trait, env_data=InfTemp, scale=TRUE)
plot(result1)

# Add to the plot the results from different smoothing of the temperature curve
result2=fit_t_env(Cetacea, trait, env_data=InfTemp, df=10, scale=TRUE)
lines(result2, col="red")

result3=fit_t_env(Cetacea, trait, env_data=InfTemp, df=50, scale=TRUE)
lines(result3, col="blue")

## Fit the environmental linear model
fit_t_env(Cetacea, trait, env_data=InfTemp, model="EnvLin", df=50, scale=TRUE)

## Fit user defined model (note that several other environmental variables
## can be simultaneously encapsulated in this function through the env argument)

# We define the function for the model
my_fun<-function(t, env_cont, param){
  param[1]*exp(param[2]*env_cont(t))
}
res<-fit_t_env(Cetacea, trait, env_data=InfTemp, model=my_fun,
  param=c(0.1,0), scale=TRUE)
# Retrieve the parameters and compare to 'result1'
res
plot(res, col="red")

## Fit user defined environmental function
require(pspline)
spline_result <- sm.spline(x=InfTemp[,1],y=InfTemp[,2], df=50)
env_func <- function(t)(predict(spline_result,t))
t<-unique(InfTemp[,1])

# We build the interpolated smoothing spline function
env_data<-splinefun(t,env_func(t))

# We then fit the model
fit_t_env(Cetacea, trait, env_data=env_data)

## Various parameterization (box constraints, df, scaling of the curve...) example
fit_t_env(Cetacea, trait, env_data=InfTemp, model="EnvLin", method="L-BFGS-B",
  scale=TRUE, lower=-30, upper=20, df=10)

## A very general model...

# We define the function for the Early-Burst/AC model:
maxtime = max(branching.times(Cetacea))

# sigma^2*e^(r*t)
my_fun_ebac <- function(t, env_cont, param){
  time = (maxtime - t)
  param[1]*exp(param[2]*time)
}
res<-fit_t_env(Cetacea, trait, env_data=InfTemp, model=my_fun_ebac,
               param=c(0.1,0), scale=TRUE)
res # note that "r" is positive: it's the AC model (~OU model on ultrametric tree)

---

**fit_t_pl**

*High-dimensional phylogenetic models of trait evolution*

**Description**

Fits high-dimensional model of trait evolution on trees through penalized likelihood. A phylogenetic Leave-One-Out Cross-Validated log-likelihood (LOOCV) is used to estimate model parameters.

**Usage**

```r
fit_t_pl(Y, tree, model=c("BM", "OU", "EB", "lambda"),
          method=c("RidgeAlt", "RidgeArch", "RidgeAltapprox",
                   "LASSO", "LASSOapprox"),
          targM=c("null", "Variance", "unitVariance"),
          REML=TRUE, up=NULL, low=NULL,
          tol=NULL, starting=NULL, SE=NULL,
          scale.height=TRUE, ...)
```

**Arguments**

- **Y**: A matrix of phenotypic traits values (the variables are represented as columns)
- **tree**: An object of class 'phylo' (see ape documentation)
- **model**: The evolutionary model. "BM" is Brownian Motion, "OU" is Ornstein-Uhlenbeck, "EB" is Early Burst, and "lambda" is Pagel’s lambda transformation.
- **method**: The penalty method. "RidgeArch": Archetype (linear) Ridge penalty, "RidgeAlt": Quadratic Ridge penalty, "LASSO": Least Absolute Selection and Shrinkage Operator. "RidgeAltapprox" and "LASSOapprox" are fast approximations of the LOOCV for the Ridge quadratic and LASSO penalties
- **targM**: The target matrix used for the Ridge regularizations. "null" is a null target, "Variance" for a diagonal unequal variance target, "unitVariance" for an equal diagonal target. Only works with "RidgeArch", "RidgeAlt", and "RidgeAltapprox" methods.
- **REML**: Use REML (default) or ML for estimating the parameters.
- **up**: Upper bound for the parameter search of the evolutionary model (optional).
- **low**: Lower bound for the parameter search of the evolutionary model (optional).
function `fit_t_pl` allows fitting various multivariate evolutionary models to high-dimensional datasets (where the number of variables $p$ is larger than $n$). Models estimates are more accurate than maximum likelihood methods. Models fit can be compared using the GIC criterion (see `?GIC`). Details about the methods are described in Clavel et al. (2019).

### Value

A list with the following components:

- `loocv`: the (negative) cross-validated penalized likelihood
- `model.par`: the evolutionary model parameter estimates
- `gamma`: the regularization/tuning parameter of the penalized likelihood
- `corrstruct`: a list with the transformed variables and the phylogenetic tree with branch length stretched to the model estimated parameters
- `model`: the evolutionary model
- `method`: the penalization method
- `p`: the number of traits
- `n`: the number of species
- `targM`: the target used for Ridge Penalization
- `R`: a list with the estimated evolutionary covariance matrix and it's inverse
- `REML`: logical indicating if the REML (TRUE) or ML (FALSE) method has been used
- `variables`: $Y$ is the input dataset and $tree$ is the input phylogenetic tree
- `SE`: the estimated standard error

### Note

The LASSO is computationally intensive. Please wait! For highly-dimensional datasets you should favor the "RidgeArch" method to speed up the computations. The Ridge penalties with "null" or "unitVariance" targets are rotation invariants.

### Author(s)

J. Clavel
References


See Also

ancestral, phyl.pca_pl, GIC.fit_pl.rpanda, gic_criterion mvgls

Examples

```r
require(mvMORPH)
set.seed(1)
n <- 32 # number of species
p <- 31 # number of traits
tree <- pbtree(n=n) # phylogenetic tree
R <- Posdef(p) # a random symmetric matrix (covariance)

# simulate a dataset
Y <- mvSIM(tree, model="BM1", nsim=1, param=list(sigma=R))

# fit the model
fit_t_pl(Y, tree, model="BM", method="RidgeAlt")

# try on rotated axis (using PCA)
trans <- prcomp(Y, center=FALSE)
fit_t_pl(trans$x, tree, model="BM", method="RidgeAlt")

# Estimate the SE (similar to Pagel's lambda for BM).
# Advised with empirical datasets
fit_t_pl(Y, tree, model="BM", method="RidgeAlt", SE=TRUE)
```

foraminifera

Foraminifera diversity since the Jurassic

Description

Foraminifera fossil diversity since the Jurassic

Usage

data(foraminifera)
getDataLikelihood

Details

Foraminifera fossil diversity since the Jurassic compiled from the Neptune Database (Lazarus, 1994) and Paleobiology Database (https://paleobiodb.org/). Diversity curves are estimated at the genus level using shareholder quorum subsampling (Alroy, 2010) at two-million-year bins. The format is a dataframe with the two following variables:

age a numeric vector corresponding to the geological age, in Myrs before the present
foraminifera a numeric vector corresponding to the estimated foraminifera change at that age

References


Examples

data(foraminifera)
plot(foraminifera)

---

dataForamInput

Description

Computes -log( likelihood ) of tip trait data under a given set of parameters, and for a specified model of trait evolution.

Usage

ggetDataLikelihood(object, data, params, v)

Arguments

object an object of class 'PhenotypicModel'.
data vector of tip trait data.
params vector of parameters, given in the same order as in the 'model' object.
v boolean specifying the verbose mode. Default value : FALSE.

Value

A numerical value : -log( likelihood ) of the model.

Author(s)

M Manceau
getMAPS_ClaDS

References

Examples

#Loading an example tree

#Creating the models
modelBM <- createModel(tree, 'BM')

dataBM <- simulateTipData(modelBM, c(0,0,0,1))

#Likelihood of the data :
getDataLikelihood(modelBM, dataBM, c(0,0,0,1))

generateTree

Description

Methods

signature(object = "PhenotypicModel") This is the only method available for this function. Same behaviour for any PhenotypicModel.

getMAPS_ClaDS

Description

Extract the MAPs (Maximum A Posteriori) for the marginal posterior distributions estimated with fit_ClaDS

Usage

getMAPS_ClaDS(sampler, burn = 1/2, thin = 1)
getMAPS_ClaDS

Arguments

sampler The output of a fit_ClaDS run.
burn Number of iterations to drop in the beginning of the chains.
thin Thinning parameter, one iteration out of "thin" is kept to compute the MAPs.

Value

A vector MAPS containing the MAPs for the marginal posterior distribution for each of the model’s parameters.


MAPS[-(1:4)] are the estimated branch-specific speciation rates, given in the same order as the edges of the phylogeny on which the inference was performed.

Author(s)

O. Maliet

References


See Also

fit_ClaDS, plot_ClaDS_chains, getMAPS_ClaDS0

Examples

data("Caprimulgidae_ClaDS2")

MAPS = getMAPS_ClaDS(Caprimulgidae_ClaDS2$sampler, thin = 1)

print(paste0("sigma = ", MAPS[1], " ; alpha = ",
        MAPS[2], " ; epsilon = ", MAPS[3], " ; l_0 = ", MAPS[4] ))

plot_ClaDS_phylo(Caprimulgidae_ClaDS2$tree, MAPS[-(1:4)])
getMAPS_ClaDS0

*Gets the Maximum A Posteriori for each ClaDS0 parameter*

---

**Description**

Extract the MAPs (Maximum A Posteriori) for the marginal posterior distributions estimated with run_ClaDS0.

**Usage**

```r
getMAPS_ClaDS0(phylo, sampler, burn=1/2, thin=1)
```

**Arguments**

- `phylo` An object of class 'phylo'.
- `sampler` The output of a run_ClaDS0 run.
- `burn` Number of iterations to drop in the beginning of the chains.
- `thin` Thinning parameter, one iteration out of "thin" is kept to compute the MAPs.

**Value**

A vector MAPS containing the MAPs for the marginal posterior distribution for each of the model’s parameters.

MAPS[1:3] are the estimated hyperparameters, with MAPS[1] the sigma parameter (new rates stochasticity), MAPS[2] the alpha parameter (new rates trend), and MAPS[3] the initial speciation rate lambda_0.

MAPS[-(1:3)] are the estimated branch-specific speciation rates, given in the same order as the phylo$edges.

**Author(s)**

O. Maliet

**References**


**See Also**

`fit_ClaDS0`, `plot_ClaDS0_chains`, `getMAPS_ClaDS`
Examples

```r
set.seed(1)

obj = sim_ClaDS( lambda_0=0.1,
                 mu_0=0.5,
                 sigma_lamb=0.7,
                 alpha_lamb=0.90,
                 condition="taxa",
                 taxa_stop = 20,
                 prune_extinct = TRUE)

tree = obj$tree
speciation_rates = obj$lamb[obj$rates]
extinction_rates = obj$mu[obj$rates]
data("ClaDS0_example")

# extract the Maximum A Posteriori for each of the parameters
MAPS = getMAPS_ClaDS0(ClaDS0_example$tree,
                       ClaDS0_example$Cl0_chains,
                       thin = 10)

# plot the simulated (on the left) and inferred speciation rates (on the right)
# on the same color scale
plot_ClaDS_phylo(ClaDS0_example$tree,
                 ClaDS0_example$speciation_rates,
                 MAPS[-(1:3)])
```

getTipDistribution

Distribution of tip trait values.

### Description

Computes the mean and variance of the tip trait distribution under a specified model of trait evolution.

### Usage

```r
getTipDistribution(object, params, v)
```

### Arguments

- **object**
  - an object of class 'PhenotypicModel'
- **params**
  - vector of parameters, given in the same order as in the 'model' object.
- **v**
  - boolean specifying the verbose mode. Default value : FALSE.

### Value

- **mean**
  - Expectation vector of the tip trait distribution.
- **Sigma**
  - Variance-covariance matrix of the tip trait distribution.
getTipDistribution-methods

**Author(s)**

M Manceau

**References**


**Examples**

```r
#Loading an example tree
tree <- read.tree(text=newick)

#Creating a BM model
modelBM <- createModel(tree, 'BM')

#Tip trait distribution under the model:
getTipDistribution(modelBM, c(0, 0, 0, 1))
```

**Description**

Computes the mean and variance of the tip trait distribution under a specified model of trait evolution.

**Methods**

- `signature(object = "PhenotypicModel")` In the most general case, this function computes the expectation vector and the variance-covariance matrix using a numerical integration procedure that may take time.
- `signature(object = "PhenotypicACDC")` The function has been optimized for this subclass.
- `signature(object = "PhenotypicADiag")` The function has been optimized for this subclass.
- `signature(object = "PhenotypicBM")` The function has been optimized for this subclass.
- `signature(object = "PhenotypicDD")` The function has been optimized for this subclass.
- `signature(object = "PhenotypicGMM")` The function has been optimized for this subclass.
- `signature(object = "PhenotypicOU")` The function has been optimized for this subclass.
- `signature(object = "PhenotypicPM")` The function has been optimized for this subclass.

**References**

**GIC**.fit_pl.rpanda

*Generalized Information Criterion (GIC) to compare models fit by Maximum Likelihood (ML) or Penalized Likelihood (PL).*

**Description**

The GIC allows comparing models fit by Maximum Likelihood (ML) or Penalized Likelihood (PL).

**Usage**

```r
## S3 method for class 'fit_pl.rpanda'
GIC(object, ...)
```

**Arguments**

- `object`: An object of class "fit_pl.rpanda". See `?fit_t_pl`
- `...`: Options to be passed through.

**Details**

GIC allows comparing the fit of various models estimated by Penalized Likelihood (see `?fit_t_pl`). It’s a wrapper to the `gic_criterion` function.

**Value**

A list with the following components:

- `LogLikelihood`: the log-likelihood estimated for the model with estimated parameters
- `GIC`: the GIC criterion
- `bias`: the value of the bias term estimated to compute the GIC

**Author(s)**

J. Clavel

**References**


**See Also**

`gic_criterion, fit_t_pl mvglsl`
Examples

```r
require(mvMORPH)
set.seed(1)
n <- 32 # number of species
p <- 40 # number of traits
tree <- pbtree(n=n) # phylogenetic tree
R <- Posdef(p) # a random symmetric matrix (covariance)
# simulate a dataset
Y <- mvSIM(tree, model="BM1", nsim=1, param=list(sigma=R))

fit1 <- fit_t_pl(Y, tree, model="BM", method="RidgeAlt")
fit2 <- fit_t_pl(Y, tree, model="OU", method="RidgeAlt")

GIC(fit1); GIC(fit2)
```

---

gic_criterion

**Generalized Information Criterion (GIC) to compare models fit by Maximum Likelihood (ML) or Penalized Likelihood (PL).**

---

**Description**

The GIC allows comparing models fit by Maximum Likelihood (ML) or Penalized Likelihood (PL).

**Usage**

```r
gic_criterion(Y, tree, model="BM", method=c("RidgeAlt", "RidgeArch", "LASSO", "ML", "RidgeAltapprox", "LASSOapprox"), targM=c("null", "Variance", "unitVariance"), param=NULL, tuning=0, REML=TRUE, ...)
```

**Arguments**

- **Y**: A matrix of phenotypic traits values (the variables are represented as columns)
- **tree**: An object of class `phylo` (see ape documentation)
- **model**: The evolutionary model, "BM" is Brownian Motion, "OU" is Ornstein-Uhlenbeck, "EB" is Early Burst, and "lambda" is Pagel's lambda transformation.
The target matrix used for the Ridge regularizations. "null" is a null target, "Variance" for a diagonal unequal variance target, "unitVariance" for an equal diagonal target. Only works with "RidgeArch","RidgeAlt" methods.

Parameter for the evolutionary model (see "model" above).

The tuning/regularization parameter.

Use REML (default) or ML for estimating the parameters.

Additional options. Not used yet.

Details
gic_criterion allows comparing the fit of various models estimated by Penalized Likelihood (see ?fit_t_pl). Use the wrapper GIC instead for models fit with fit_t_pl.

Value

a list with the following components

LogLikelihood the log-likelihood estimated for the model with estimated parameters

GIC the GIC criterion

bias the value of the bias term estimated to compute the GIC

Note

The tuning parameter is assumed to be zero when using the "ML" method.

Author(s)

J. Clavel

References


See Also

GIC.fit_pl,rpanda,fit_t_pl

Examples

```
require(mvMORPH)
set.seed(123)
n <- 32 # number of species
p <- 2 # number of traits
```
tree <- pbtree(n=n) # phylogenetic tree
R <- Posdef(p) # a random symmetric matrix (covariance)

# simulate a dataset
Y <- mvSIM(tree, model="BM1", nsim=1, param=list(sigma=R))

# Compute the GIC for ML
gic_criterion(Y, tree, model="BM", method="ML", tuning=0) # ML

# Compare with PL?
test <- fit_t_pl(Y, tree, model="BM", method="RidgeAlt")
#GIC(test)

greenalgae

---

**greenalgae**

*Green algae diversity since the Jurassic*

**Description**

Green algae fossil diversity since the Jurassic

**Usage**

data(greenalgae)

**Details**

Green algae fossil diversity since the Jurassic compiled from the Neptune Database (Lazarus, 1994) and Paleobiology Database (https://paleobiodb.org/). Diversity curves are estimated at the genus level using shareholder quorum subsampling (Alroy, 2010) at two-million-year bins. The format is a dataframe with the two following variables:

- **age** a numeric vector corresponding to the geological age, in Myrs before the present
- **greenalgae** a numeric vector corresponding to the estimated green algae change at that age

**References**


**Examples**

data(greenalgae)
plot(greenalgae)
**InfTemp**

*Paleotemperature data across the Cenozoic*

**Description**

Paleotemperature data across the Cenozoic inferred from delta O18 measurements

**Usage**

```r
data(InfTemp)
```

**Details**

Paleotemperature data inferred from delta O18 measurements using the equation of Epstein et al. (1953). The format is a dataframe with the two following variables:

- **Age** a numeric vector corresponding to the geological age, in Myrs before the present
- **Temperature** a numeric vector corresponding to the inferred temperature at that age

**References**


**Examples**

```r
data(InfTemp)
plot(InfTemp)
```

---

**JSDtree**

*Jensen-Shannon distance between phylogenies*

**Description**

Computes the Jensen-Shannon distance metric between spectral density profiles of phylogenies.

**Usage**

```r
JSDtree(phylo,meth=c("standard"))
```
Arguments

phylo  a list of objects of type 'phylo' (see ape documentation)
meth  the method used to compute the spectral density, which can either be "standard", "normal1", or "normal2". if set to "normal1", computes the spectral density normalized to the degree matrix. if set to "normal2", computes the spectral density normalized to the number of eigenvalues. if set to "standard", computes the unnormalized version of the spectral density (see the associated paper for an explanation)

Value

a matrix providing the Jensen-Shannon distance values between phylogeny pairs

Author(s)

E Lewitus

References

Lewitus, E., Morlon, H., Characterizing and comparing phylogenies from their Laplacian spectrum, bioRxiv doi: http://dx.doi.org/10.1101/026476

See Also

JSDtree_cluster, spectR, BICompare

Examples

trees<-TESS::tess.sim.age(n=20,age=10,0.15,0.05,MRCA=TRUE)
JSDtree(trees)

JSDtree_cluster  Clustering of phylogenies

Description

Clusters phylogenies using hierarchical and k-medoids clustering

Usage

JSDtree_cluster(JSDtree, alpha=0.9, draw=T)

Arguments

JSDtree  a matrix of distances between phylogeny pairs, typically the output of the JSDtree function when the distance is measured as the Jensen-Shannon distance
alpha  the confidence value for demarcating clusters in the hierarchical clustering plot; the default is 0.9
draw  plot heatmap and hierarchical cluster in new windows
Value

plots a heatmap and a hierarchical cluster with bootstrap support, and outputs results of the k-medoids clustering in the form of a list with the following components

clusters the optimal number of clusters around medoids (see pamk documentation)
cluster_assignments assignments of trees to clusters
cluster_support a list with the following components: widths: a table specifying the cluster to which each tree belongs, the neighbor (i.e. most similar) cluster, and the silhouette width of the observation (see silhouette documentation); clus.avg.widths: average silhouette width for each cluster; vg.width: average silhouette width across all clusters

Note

The k-medoids clustering may not work with fewer than 10 trees

Author(s)

E Lewitus

References

Lewitus, E., Morlon, H., Characterizing and comparing phylogenies from their Laplacian spectrum, bioRxiv doi: http://dx.doi.org/10.1101/026476

See Also

JSDtree

Examples

trees<-TESS::tess.sim.age(n=20,age=10,0.15,0.05,MRCA=TRUE)
res<-JSDtree(trees)
#JSDtree_cluster(res,alpha=0.9,draw=T)

Description

Computes the Jensen-Shannon distance metric between spectral density profiles of phylogenetic trait data and clusters on those distances.
JSDt_cluster

Usage

JSDt_cluster(phylo, mat, plot=F)

Arguments

phylo an object of type 'phylo' (see ape documentation)
mat a matrix of trait data with one trait per column and rows aligned to phylo tips
plot plot hierarchical cluster in a new window

Value

plots a heatmap and hierarchical cluster with bootstrap support (>0.9) and outputs results of the k-medoids clustering on the optimal number of clusters in the form of a list with the following components

clusters a list with the following components: size, max_diss, av_diss, diameter, and separation
J-S matrix a matrix providing the Jensen-Shannon distance values between pairs of phylogenetic trait data
cluster assignment a table that lists for each trait its cluster assignment and silhouette width

Author(s)

E Lewitus

References


See Also

spectR_t

Examples

data(Cetacea)
n<-length(Cetacea$tip.label)
mat<-replicate(20, rnorm(n))
colnames(mat)<-1:dim(mat)[2]
JSDt_cluster(Cetacea,mat)
**landplant**

*Land plant diversity since the Jurassic*

**Description**

Land plant fossil diversity since the Jurassic

**Usage**

```r
data(landplant)
```

**Details**

Land plant fossil diversity since the Jurassic compiled from the Neptune Database (Lazarus, 1994) and Paleobiology Database (https://paleobiodb.org/). Diversity curves are estimated at the genus level using shareholder quorum subsampling (Alroy, 2010) at two-million-year bins. The format is a dataframe with the two following variables:

- `age` a numeric vector corresponding to the geological age, in Myrs before the present
- `landplant` a numeric vector corresponding to the estimated land plant change at that age

**References**


**Examples**

```r
data(landplant)
plot(landplant)
```

---

**likelihood_bd**

*Likelihood of a phylogeny under the general birth-death model*

**Description**

Computes the likelihood of a phylogeny under a birth-death model with potentially time-varying rates and potentially missing extant species. Notations follow Morlon et al. PNAS 2011.

**Usage**

```r
likelihood_bd(phylo, tot_time, f.lamb, f.mu, f, cst.lamb = FALSE, cst.mu = FALSE, expo.lamb = FALSE, expo.mu = FALSE, dt=0, cond = "crown")
```
Arguments

- **phylo**: an object of type 'phylo' (see ape documentation)
- **tot_time**: the age of the phylogeny (crown age, or stem age if known). If working with crown ages, tot_time is given by max(node.age(phylo)$ages).
- **f.lamb**: a function specifying the time-variation of the speciation rate $\lambda$. This function as a single argument (time). Any function may be used.
- **f.mu**: a function specifying the time-variation of the speciation rate $\mu$. This function as a single argument (time). Any function may be used.
- **f**: the fraction of extant species included in the phylogeny
- **cst.lamb**: logical: should be set to TRUE only if f.lamb is constant (i.e. does not depend on time) to use analytical instead of numerical computation in order to reduce computation time.
- **cst.mu**: logical: should be set to TRUE only if f.mu is constant (i.e. does not depend on time) to use analytical instead of numerical computation in order to reduce computation time.
- **expo.lamb**: logical: should be set to TRUE only if f.lamb is exponential to use analytical instead of numerical computation in order to reduce computation time.
- **expo.mu**: logical: should be set to TRUE only if f.mu is exponential to use analytical instead of numerical computation in order to reduce computation time.
- **dt**: the default value is 0. In this case, integrals in the likelihood are computed using R "integrate" function, which can be quite slow. If a positive dt is given as argument, integrals are computed using a piece-wise contant approximation, and dt represents the length of the intervals on which functions are assumed to be constant. For an exponential dependency of the speciation rate with time, we found that dt=1e-3 gives a good trade-off between precision and computation time.
- **cond**: conditioning to use to fit the model:
  - "FALSE": no conditioning (not recommended);
  - "stem": conditioning on the survival of the stem lineage (use when the stem age is known, in this case tot_time should be the stem age);
  - "crown" (default): conditioning on a speciation event at the crown age and survival of the 2 daugther lineages (use when the stem age is not known, in this case tot_time should be the crown age).

Details

When specifying f.lamb and f.mu, time runs from the present to the past (hence if the speciation rate decreases with time, f.lamb must be a positive function of time).

Value

the loglikelihood value of the phylogeny, given f.lamb and f.mu

Author(s)

H Morlon
References


Examples

data(Cetacea)
tot_time <- max(node.age(Cetacea)$ages)
# Compute the likelihood for a pure birth model (no extinction) with
# an exponential variation of speciation rate with time
lamb_par <- c(0.1, 0.01)
f.lamb <- function(t){lamb_par[1] * exp(lamb_par[2] * t)}
f.mu <- function(t){0}
f <- 87/89
lh <- likelihood_bd(Cetacea,tot_time,f.lamb,f.mu,f,cst.mu=TRUE,expo.lamb=TRUE, dt=1e-3)

---

**likelihood_coal_cst**  Likelihood of a phylogeny under the equilibrium diversity model

Description

Computes the likelihood of a phylogeny under the equilibrium diversity model with potentially time-varying rates and potentially missing extant species. Notations follow Morlon et al. *PloSB* 2010.

Usage

`likelihood_coal_cst(Vtimes, ntips, tau0, gamma, N0)`

Arguments

- `Vtimes`: a vector of branching times (sorted from present to past)
- `ntips`: the number of tips in the phylogeny
- `tau0`: the turnover rate at present
- `gamma`: the parameter controlling the exponential variation in turnover rate. With gamma=0, the turnover rate is constant over time.
- `N0`: the number of extant species

Details

Time runs from the present to the past. Hence, a positive gamma (for example) means that the turnover rate declines from past to present.
Value

a list containing the following components:

res the loglikelihood value of the phylogeny, given tau0 and gamma
all vector of all the individual loglikelihood values corresponding to each branching event

Author(s)

H Morlon

References


Examples

data(Cetacea)
Vtimes <- sort(branching.times(Cetacea))
tau0 <- 0.1
gamma <- 0.001
ntips <- Ntip(Cetacea)
N0 <- 89
likelihood <- likelihood_coal_cst(Vtimes,ntips,tau0,gamma,N0)

Description

Computes the likelihood of a phylogeny under the expanding diversity model with potentially time-varying rates and potentially missing extant species to a phylogeny. Notations follow Morlon et al. PloSB 2010.

Usage

likelihood_coal_var(Vtimes, ntips, lamb0, alpha, mu0, beta, N0, pos = TRUE)

Arguments

Vtimes a vector of branching times (sorted from present to past)
ntips number of species in the phylogeny
lamb0 the speciation rate at present
alpha the parameter controlling the exponential variation in speciation rate.
mu0 the extinction rate at present
likelihood_sgd

beta the parameter controlling the exponential variation in extinction rate.
N0 the number of extant species
pos logical: should be set to FALSE only to not enforce positive speciation and extinction rates

Details
Time runs from the present to the past. Hence, a positive alpha (for example) means that the speciation rate declines from past to present.

Value
a list containing the following components:

res the loglikelihood value of the phylogeny, given the parameters
all vector of all the individual loglikelihood values corresponding to each branching event

Author(s)
H Morlon

References

Examples
data(Cetacea)
Vtimes <- sort(branching.times(Cetacea))
lamb0 <- 0.1
alpha <- 0.001
mu0<-0
beta<-0
ntips <- Ntip(Cetacea)
N0 <- 89
likelihood <- likelihood_coal_var(Vtimes, ntips, lamb0, alpha, mu0, beta, N0)

likelihood_sgd Likelihood of a phylogeny under the SGD model

Description
Computes the likelihood of a phylogeny under the SGD model with exponential increasing of the metacommunity, and potentially missing extant species. Notations follow Manceau et al. (2015).
Usage

likelihood_sgd(phylo, tot_time, b, d, nu, f)

Arguments

phylo an object of type 'phylo' (see ape documentation)
tot_time the age of the phylogeny (crown age, or stem age if known). If working with
crown ages, tot_time is given by max(node.age(phylo)$ages).
b the (constant) birth rate of individuals in the model.
d the (constant) death rate of individuals in the model.
nu the (constant) mutation rate of individuals in the model.
f the fraction of extant species included in the phylogeny

Value

the likelihood value of the phylogeny, given the model and the parameter values b, d, nu.

Author(s)

M Manceau

References


Examples

data(Cetacea)
tot_time <- max(node.age(Cetacea)$ages)
b <- 1e6
d <- 1e6-0.5
nu <- 0.6
f <- 87/89
#lh <- likelihood_sgd(Cetacea, tot_time, b, d, nu, f)

likelihood_subgroup_model

Likelihood of a dataset under models with biogeography fit to a sub-
group.

Description

Computes the likelihood of a dataset under either the linear or exponential diversity dependent
model with specified sigma2 and slope values and with a geography.object formed using CreateGeoObject.
Usage

likelihood_subgroup_model(data, phylo, geography.object, model=c("MC", "DDexp", "DDlin"), par, return.z0=FALSE, maxN=NULL)

Arguments

phylo  
an object of type 'phylo' (see ape documentation) produced as "map" from CreateGeobyClassObject. NB: the length of this object need not match number of items in data, since map may include tips outside of group with some part of their branch in the group

data  
a named vector of continuous data for a subgroup of interest with names corresponding to phylo$tip.label

geography.object  
a list of sympatry/group membership through time created using CreateGeobyClassObject

model  
model chosen to fit trait data, "DDlin" is the diversity-dependent linear model, and "DDexp" is the diversity-dependent exponential model of Weir & Mursleen 2013.

par  
a vector listing a value for log(sig2) (see Note) and either b (for the linear diversity dependent model) or r (for the exponential diversity dependent model), in that order.

return.z0  
logical indicating whether to return an estimate of the trait value at the root given the parameter values (if TRUE, function returns root value rather than negative log-likelihood)

maxN  
when fitting DDlin model, it is necessary to specify the maximum number of sympatric lineages to ensure that the rate returned does not correspond to negative sig2 values at any point in time (see Details).

Details

When specifying par, log(sig2) (see Note) must be listed before the slope parameter (b or r).

maxN can be calculated using maxN=max(vapply(geo.object$geography.object, function(x)max(rowSums(x)),1)), where geo.object is the output of CreateGeoObject

Value

The negative log-likelihood value of the dataset (accordingly, the negative of the output should be recorded as the likelihood), given the phylogeny, sig2 and slope values, and geography.object.

If return.z0=TRUE, the estimated root value for the par values is returned instead of the negative log-likelihood.

Note

To stabilize optimization, this function exponentiates the input sig2 value, thus the user must input the log(sig2) value to compute the correct log likelihood (see example).
Author(s)

Jonathan Drury jonathan.p.drury@gmail.com
Julien Clavel

References


See Also

fit_t_comp CreateGeoObject likelihood_t_DD

Examples

data(BGB.examples)

Canidae.phylo<-BGB.examples$Canidae.phylo
names(dummy.group)<-Canidae.phylo$tip.label

Canidae.simmap<-phytools::make.simmap(Canidae.phylo, dummy.group)

set.seed(123)
Canidae.data<-rnorm(length(Canidae.phylo$tip.label))
names(Canidae.data)<-Canidae.phylo$tip.label
Canidae.A<-Canidae.data[which(dummy.group=="A")]
Canidae.geobyclass.object<-CreateGeobyClassObject(phylo=Canidae.phylo, simmap=Canidae.simmap, trim.class="A", ana.events=BGB.examples$Canidae.ana.events, clado.events=BGB.examples$Canidae.clado.events,stratified=FALSE, rnd=5)

par <- c(log(0.01),-0.0005)
maxN<-max(vapply(Canidae.geobyclass.object$geography.object$geography.object, function(x)max(rowSums(x)),1))

lh <- -likelihood_subgroup_model(data=Canidae.A, phylo=Canidae.geobyclass.object$map, geo.object=Canidae.geobyclass.object$geography.object, model="DDlin", par=par, return.z0=FALSE, maxN=maxN)
likelihood_t_DD  Likelihood of a dataset under diversity-dependent models.

Description

Computes the likelihood of a dataset under either the linear or exponential diversity dependent model with specified sigma2 and slope values.

Usage

likelihood_t_DD(phylo, data, par, model=c("DDlin","DDexp"))

Arguments

phylo  an object of type 'phylo' (see ape documentation)
data  a named vector of continuous data with names corresponding to phylo$tip.label
par  a vector listing a value for log(sig2) (see Note) and either b (for the linear diversity dependent model) or r (for the exponential diversity dependent model), in that order.
model  model chosen to fit trait data, "DDlin" is the diversity-dependent linear model, and "DDexp" is the diversity-dependent exponential model of Weir & Mursleen 2013.

Details

When specifying par, log(sig2) must be listed before the slope parameter (b or r).

Value

the negative log-likelihood value of the dataset (accordingly, the negative of the output should be recorded as the likelihood), given the phylogeny and sig2 and slope values

Note

To stabilize optimization, this function exponentiates the input sig2 value, thus the user must input the log(sig2) value to compute the correct log likelihood (see example).

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com
Julien Clavel

References

See Also

fit_t_comp likelihood_t_DD_geog

Examples

data(Anolis.data)
phylo <- Anolis.data$phylo
pPC1 <- Anolis.data$data

# Compute the likelihood that the r value is twice the ML estimate for the DDexp model
par <- c(0.08148371, (2*-0.3223835))
lh <- -likelihood_t_DD(phylo,pPC1,par,model="DDexp")

Description

Computes the likelihood of a dataset under either the linear or exponential diversity dependent model with specified \( \sigma^2 \) and slope values and with a geography.object formed using CreateGeoObject.

Usage

likelihood_t_DD_geog(phylo, data, par, geo.object, model=c("DDlin","DDexp"), maxN=NA)

Arguments

- **phylo**: an object of type 'phylo' (see ape documentation)
- **data**: a named vector of continuous data with names corresponding to phylo$tip.label
- **par**: a vector listing a value for \( \log(\sigma^2) \) (see Note) and either \( b \) (for the linear diversity dependent model) or \( r \) (for the exponential diversity dependent model), in that order.
- **geo.object**: a list of sympatry through time created using CreateGeoObject
- **model**: model chosen to fit trait data, "DDlin" is the diversity-dependent linear model, and "DDexp" is the diversity-dependent exponential model of Weir & Mursleen 2013.
- **maxN**: when fitting DDlin model, it is necessary to specify the maximum number of sympatic lineages to ensure that the rate returned does not correspond to negative sig2 values at any point in time (see Details).

Details

When specifying \( \text{par}, \log(\sigma^2) \) (see Note) must be listed before the slope parameter (\( b \) or \( r \)).

maxN can be calculated using \( \text{maxN} = \max(vapply(\text{geo.object$geography.object}, \text{function}(x) \max(\text{rowSums}(x)), 1)) \), where geo.object is the output of CreateGeoObject.
Value

the negative log-likelihood value of the dataset (accordingly, the negative of the output should be recorded as the likelihood), given the phylogeny, sig2 and slope values, and geography.object.

Note

To stabilize optimization, this function exponentiates the input sig2 value, thus the user must input the log(sig2) value to compute the correct log likelihood (see example).

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com
Julien Clavel

References


See Also

fit_t_comp CreateGeoObject likelihood_t_DD

Examples

data(Anolis.data)
phylo <- Anolis.data$phylo
pPC1 <- Anolis.data$data
geography.object <- Anolis.data$geography.object

# Compute the likelihood with geography using ML parameters for fit without geography
par <- c(log(0.01153294),-0.0006692378)
maxN<-max(vapply(geography.object$geography.object,function(x)max(rowSums(x)),1))
lh <- -likelihood_t_DD_geog(phylo,pPC1,par,geography.object,model="DDlin",maxN=maxN)

likelihood_t_env  Likelihood of a dataset under environmental models of trait evolution.

Description

Computes the likelihood of a dataset under either the linear or exponential environmental model, or an user defined environmental model. This function is used internally by fit_t_env.

Usage

likelihood_t_env(phylo, data, model=c("EnvExp", "EnvLin"), ...)
Arguments

phylo  an object of class 'phylo' (see ape documentation)
data  a named vector of continuous data with names corresponding to phylo$tip.label
...  "param", "fun", "times", "mtot" and "error" arguments.
- param: a vector with the parameters used in the environmental function. The first value is $\sigma^2$ and the second is $\beta$.
- fun: a time continuous function of an environmental variable (see e.g. ?fit_t_env)
- times: a vector of branching times starting at zero (e.g. max(branching.times(phylo))-branching.times(phylo))
- mtot: root age of the tree (e.g. max(branching.times(phylo)))
- error: a vector of standard error (se) for each species
If the "times" argument is not provided, the "phylo" object is used to compute it as well as "mtot".
Note that the argument "mu" can be used to specify the root state (e.g. when using an mcmc sampler)

model  model chosen to fit trait data, "EnvExp" is the exponential-environmental model, and "EnvLin" is the linear-environmental model. Otherwise, an user specified model can be provided.

Details

the "fun" argument can be filled by an environmental dataframe.

Value

the log-likelihood value of the environmental model

Author(s)

Julien Clavel

References


See Also

fit_t_env

Examples

data(Cetacea)
data(InfTemp)

# Simulate a trait with temperature dependence on the Cetacean tree
set.seed(123)
likelihood_t_MC

trait <- sim_t_env(Cetacea, param=c(0.1,-0.2), env.data=InfTemp, model="EnvExp", root.value=0, step=0.001, plot=TRUE)

# Compute the likelihood
likelihood_t_env(Cetacea, trait, param=c(0.1, 0), fun=InfTemp, model="EnvExp")

# Provide the times
rttime<-branching.times(Cetacea)
mtot<-max(rttime)
times<-mtot-rttime

likelihood_t_env(Cetacea,trait,param=c(0.1, 0), fun=InfTemp, times=times, mtot=mtot, model="EnvExp")

# Provide the environmental function rather than the dataset (faster if used recursively)
spline_result <- sm.spline(InfTemp[,1],InfTemp[,2], df=50)
env_func <- function(t){predict(spline_result,t)}
t<-unique(InfTemp[,1])
# We build the interpolated smoothing spline function
env_data<-splinefun(t,env_func(t))

likelihood_t_env(Cetacea, trait, param=c(0.1, 0), fun=env_data, times=times, mtot=mtot, model="EnvExp")

---

likelihood_t_MC  

Likelihood of a dataset under the matching competition model.

Description

Computes the likelihood of a dataset under the matching competition model with specified \( \sigma^2 \) and \( S \) values.

Usage

likelihood_t_MC(phylo, data, par)

Arguments

phylo  an object of type 'phylo' (see ape documentation)
data  a named vector of continuous data with names corresponding to phylo$tip.label
par  a vector listing a value for \( \log(\sigma^2) \) (see Note) and \( S \) (parameters of the matching competition model), in that order

Details

When specifying par, \( \log(\sigma^2) \) must be listed before \( S \).
Value

the negative log-likelihood value of the dataset (accordingly, the negative of the output should be recorded as the likelihood), given the phylogeny and sig2 and S values.

Note

To stabilize optimization, this function exponentiates the input sig2 value, thus the user must input the log(sig2) value to compute the correct log likelihood (see example).

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com
Julien Clavel

References


See Also

`fit_t_comp`, `likelihood_t_MC_geog`

Examples

```r
data(Anolis.data)
phylo <- Anolis.data$phylo
pPC1 <- Anolis.data$data

# Compute the likelihood that the S value is twice the ML estimate
par <- c(0.0003139751, (2*-0.06387258))
lh <- -likelihood_t_MC(phylo, pPC1, par)
```

Description

Computes the likelihood of a dataset under the matching competition model with specified sigma2 and S values and with a geography.object formed using `CreateGeoObject`.

Usage

`likelihood_t_MC_geog(phylo, data, par, geo.object)`
Arguments

- **phylo**: an object of type 'phylo' (see ape documentation)
- **data**: a named vector of continuous data with names corresponding to phylo$tip.label
- **par**: a vector listing a value for \( \log(\text{sig2}) \) (see Note) and \( S \) (parameters of the matching competition model), in that order
- **geo.object**: a geography object indicating sympatry through time, created using `CreateGeoObject`

Details

When specifying `par`, \( \log(\text{sig2}) \) must be listed before \( S \).

Value

the negative log-likelihood value of the dataset (accordingly, the negative of the output should be recorded as the likelihood), given the phylogeny, sig2 and \( S \) values, and `geo.object`.

Note

\( S \) must be negative (if it is positive, the likelihood function will multiply input by -1).

To stabilize optimization, this function exponentiates the input \( \text{sig2} \) value, thus the user must input the \( \log(\text{sig2}) \) value to compute the correct log likelihood (see example).

Author(s)

Jonathan Drury jonathan.p.drury@gmail.com
-Julien Clavel

References


See Also

`fit_t_comp` `CreateGeoObject` `likelihood_t_MC`

Examples

data(Anolis.data)
phylo <- Anolis.data$phylo
pPC1 <- Anolis.data$data
geo.object <- Anolis.data$geo.object

# Compute the likelihood with geography using ML parameters for fit without geography
par <- c(0.0003139751, -0.06387258)
llh <- -likelihood_t_MC_geog(phylo, pPC1, par, geo.object)
lines.fit_t.env

Add to a plot line segments joining the phenotypic evolutionary rate through time estimated by the fit_t_env function

Description
Plot estimated evolutionary rate as a function of the environmental data and time.

Usage

```r
## S3 method for class 'fit_t.env'
lines(x, steps = 100, ...)
```

Arguments

- `x`: an object of class 'fit_t.env' obtained from a fit_t_env fit.
- `steps`: the number of steps from the root to the present used to compute the evolutionary rate $\sigma^2$ through time.
- `...`: further arguments to be passed to `plot`. See `?plot`.

Value

lines.fit_t.env returns invisibly a list with the following components used to add the line segments to the current plot:

- `time_steps`: the times steps where the climatic function was evaluated to compute the rate. The number of steps is controlled through the argument `steps`.
- `rates`: the estimated evolutionary rate through time estimated at each `time_steps`.

Note

All the graphical parameters (see `par`) can be passed through (e.g. line type: `lty`, line width: `lwd`, color: `col` ...)

Author(s)

J. Clavel

References


See Also

`plot.fit_t.env`, `likelihood_t_env`
Examples

```r
data(Cetacea)
data(InfTemp)

# Plot estimated evolutionary rate as a function of the environmental data and time.
set.seed(123)
trait <- sim_t_env(Cetacea, param=c(0.1,-0.2), env_data=InfTemp, model="EnvExp",
root.value=0, step=0.01, plot=TRUE)

## Fit the Environmental-exponential model with different smoothing parameters
result1=fit_t_env(Cetacea, trait, env_data=InfTemp, scale=TRUE)
result2=fit_t_env(Cetacea, trait, env_data=InfTemp, scale=TRUE, df=10)

# first plot result1
plot(result1, lwd=3)

# add result2 to the current plot
lines(result2, lty=2, lwd=3, col="red")
```

---

`modelSelection`  
*Phenotypic model selection from tip trait data.*

Description

For each model taken as input, fits the model and returns its AIC value in a recap table.

Usage

```r
modelSelection(object, data)
```

Arguments

- **object**: a vector of objects of class 'PhenotypicModel'.
- **data**: vector of tip trait data.

Details

Warning: This function relies on the standard R optimizer "optim". It may not always converge well. Please double check the convergence by trying distinct parameter sets for the initialisation.

Value

A recap table presenting the AIC value of each model.
Author(s)
M Manceau

References

---

modelSelection-methods

~~ Methods for Function modelSelection ~~

---

Description

~~ Methods for function modelSelection ~~

Methods

signature(object = "PhenotypicModel") This is the only method available for this function. Same behaviour for any PhenotypicModel.

---

MPhiFFT

A class used internally to compute ClaDS's likelihood

---

Description

This class represents a matrix \( A = (1/\text{rowSums(Toep)}) \times \text{Toep} \) where Toep is a Toeplitz matrix.

References


See Also

fit_ClaDS
**ostracoda**

---

**Description**

Ostracod fossil diversity since the Jurassic

**Usage**

```r
data(sealevel)
```

**Details**

Ostracod fossil diversity since the Jurassic compiled from the Neptune Database (Lazarus, 1994) and Paleobiology Database (https://paleobiodb.org/). Diversity curves are estimated at the genus level using shareholder quorum subsampling (Alroy, 2010) at two-million-year bins. The format is a dataframe with the two following variables:

- `age` a numeric vector corresponding to the geological age, in Myrs before the present
- `ostracoda` a numeric vector corresponding to the estimated ostracod change at that age

**References**


**Examples**

```r
data(ostracoda)
plot(ostracoda)
```

---

**PhenotypicACDC-class**  
*Class “PhenotypicACDC”*

---

**Description**

Subclass of the PhenotypicModel class intended to represent the model of ACcelerating or DeCel-erating phenotypic evolution.

**Objects from the Class**

Objects can be created by calls of the form `new("PhenotypicACDC", ...).`
Slots

matrixCoalescenceTimes: Object of class "matrix" ~~
name: Object of class "character" ~~
period: Object of class "numeric" ~~
aAGamma: Object of class "function" ~~
numbersCopy: Object of class "numeric" ~~
numbersPaste: Object of class "numeric" ~~
initialCondition: Object of class "function" ~~
paramsNames: Object of class "character" ~~
constraints: Object of class "function" ~~
params0: Object of class "numeric" ~~
tipLabels: Object of class "character" ~~
tipLabelsSimu: Object of class "character" ~~
comment: Object of class "character" ~~

Extends

Class "PhenotypicModel", directly.

Methods

getTipDistribution signature(object = "PhenotypicACDC"): ...

Author(s)

Marc Manceau

References


Examples

showClass("PhenotypicACDC")
**PhenotypicADiag-class**  
Class "PhenotypicADiag"

**Description**
A subclass of the PhenotypicModel class, intended to represent models of phenotypic evolution with a "A" matrix diagonalizable.

**Objects from the Class**
Objects can be created by calls of the form new("PhenotypicADiag", ...).

**Slots**
- name: Object of class "character" ~
- period: Object of class "numeric" ~
- aAGamma: Object of class "function" ~
- numbersCopy: Object of class "numeric" ~
- numbersPaste: Object of class "numeric" ~
- initialCondition: Object of class "function" ~
- paramsNames: Object of class "character" ~
- constraints: Object of class "function" ~
- params0: Object of class "numeric" ~
- tipLabels: Object of class "character" ~
- tipLabelsSimu: Object of class "character" ~
- comment: Object of class "character" ~

**Extends**
Class "PhenotypicModel", directly.

**Methods**
- getTipDistribution signature(object = "PhenotypicADiag"): ...

**Author(s)**
Marc Manceau

**References**
PhenotypicBM-class

Examples

showClass("PhenotypicADiag")

PhenotypicBM-class  Class "PhenotypicBM"

Description

A subclass of the PhenotypicModel class, intended to represent the model of Brownian phenotypic evolution.

Objects from the Class

Objects can be created by calls of the form new("PhenotypicBM",...).

Slots

- matrixCoalescenceTimes: Object of class "matrix"
- name: Object of class "character"
- period: Object of class "numeric"
- aAGamma: Object of class "function"
- numbersCopy: Object of class "numeric"
- numbersPaste: Object of class "numeric"
- initialCondition: Object of class "function"
- paramsNames: Object of class "character"
- constraints: Object of class "function"
- params0: Object of class "numeric"
- tipLabels: Object of class "character"
- tipLabelsSimu: Object of class "character"
- comment: Object of class "character"

Extends

Class "PhenotypicModel", directly.

Methods

- `getTipDistribution` signature(object = "PhenotypicBM"): ...

Author(s)

Marc Manceau
PhenotypicDD-class

References


Examples

showClass("PhenotypicBM")

PhenotypicDD-class Class "PhenotypicDD"

Description

A subclass of the PhenotypicModel class, intended to represent the model of Density-Dependent phenotypic evolution.

Objects from the Class

Objects can be created by calls of the form new("PhenotypicDD",...).

Slots

matrixCoalescenceJ: Object of class "matrix" ~~
MLivingLineages: Object of class "numeric" ~~
name: Object of class "character" ~~
period: Object of class "numeric" ~~
aAGamma: Object of class "function" ~~
numbersCopy: Object of class "numeric" ~~
numbersPaste: Object of class "numeric" ~~
initialCondition: Object of class "function" ~~
paramsNames: Object of class "character" ~~
constraints: Object of class "function" ~~
params0: Object of class "numeric" ~~
tipLabels: Object of class "character" ~~
tipLabelsSimu: Object of class "character" ~~
comment: Object of class "character" ~~

Extends

Class "PhenotypicModel", directly.
Methods

**getTipDistribution** signature(object = "PhenotypicDD"): ...

Author(s)

Marc Manceau

References


Examples

showClass("PhenotypicDD")

PhenotypicGMM-class  Class "PhenotypicGMM"

Description

A subclass of the PhenotypicModel class, intended to represent the Generalist Matching Mutualism model of phenotypic evolution. This is a model of phenotypic evolution with interactions between two clades, running on two trees.

Objects from the Class

Objects can be created by calls of the form `new("PhenotypicGMM",...)`.

Slots

- `n1`: Object of class "numeric"
- `n2`: Object of class "numeric"
- `name`: Object of class "character"
- `period`: Object of class "numeric"
- `aAGamma`: Object of class "function"
- `numbersCopy`: Object of class "numeric"
- `numbersPaste`: Object of class "numeric"
- `initialCondition`: Object of class "function"
- `paramsNames`: Object of class "character"
- `constraints`: Object of class "function"
- `params0`: Object of class "numeric"
- `tipLabels`: Object of class "character"
- `tipLabelsSimu`: Object of class "character"
- `comment`: Object of class "character"
PhenotypicModel-class

Description

This class describes a model of phenotypic evolution running on a phylogenetic tree, with or without interactions between lineages.

Objects from the Class

Objects can be created by calls of the form `new("PhenotypicModel", ...`). Alternatively, you may just want to use the `createModel` function for predefined models.

Slots

- name: Object of class "character"
- period: Object of class "numeric"
- aAGamma: Object of class "function"
- numbersCopy: Object of class "numeric"
- numbersPaste: Object of class "numeric"
- initialCondition: Object of class "function"
- paramsNames: Object of class "character"
- constraints: Object of class "function"
- params0: Object of class "numeric"
PhenotypicOU-class

Methods

[<- signature(x = "PhenotypicModel", i = "ANY", j = "ANY", value = "ANY"): ...
[ signature(x = "PhenotypicModel", i = "ANY", j = "ANY", drop = "ANY"): ...

fitTipData signature(object = "PhenotypicModel"): ...

dataLikelihood signature(object = "PhenotypicModel"): ...

dataLikelihood signature(object = "PhenotypicModel"): ...

modelSelection signature(object = "PhenotypicModel"): ...

print signature(x = "PhenotypicModel"): ...

show signature(object = "PhenotypicModel"): ...

simulate TipData signature(object = "PhenotypicModel"): ...

Author(s)

Marc Manceau

References


Examples

showClass("PhenotypicModel")

PhenotypicOU-class  Class "PhenotypicOU"

Description

A subclass of the PhenotypicModel class, intended to represent the Ornstein-Uhlenbeck model of phenotypic evolution.

Objects from the Class

Objects can be created by calls of the form new("PhenotypicOU", ...).
Slots

- matrixCoalescenceTimes: Object of class "matrix"
- name: Object of class "character"
- period: Object of class "numeric"
- aAGamma: Object of class "function"
- numbersCopy: Object of class "numeric"
- numbersPaste: Object of class "numeric"
- initialCondition: Object of class "function"
- paramsNames: Object of class "character"
- constraints: Object of class "function"
- params0: Object of class "numeric"
- tipLabels: Object of class "character"
- tipLabelsSimu: Object of class "character"
- comment: Object of class "character"

Extends

Class "PhenotypicModel", directly.

Methods

gTipDistribution signature(object = "PhenotypicOU"): ...

Author(s)

Marc Manceau

References


Examples

showClass("PhenotypicOU")
PhenotypicPM-class

Class "PhenotypicPM"

Description

A subclass of the PhenotypicModel class, intended to represent the Phenotypic Matching model of phenotypic evolution, by Nuismer and Harmon (Eco Lett, 2014).

Objects from the Class

Objects can be created by calls of the form new("PhenotypicPM",...).

Slots

- name: Object of class "character"
- period: Object of class "numeric"
- aAGamma: Object of class "function"
- numbersCopy: Object of class "numeric"
- numbersPaste: Object of class "numeric"
- initialCondition: Object of class "function"
- paramsNames: Object of class "character"
- constraints: Object of class "function"
- params0: Object of class "numeric"
- tipLabels: Object of class "character"
- tipLabelsSimu: Object of class "character"
- comment: Object of class "character"

Extends

Class "PhenotypicModel", directly.

Methods

- getTipDistribution signature(object = "PhenotypicPM"):

Author(s)

Marc Manceau

References

Examples

showClass("PhenotypicPM")

---

Phocoenidae  

Phocoenidae phylogeny

Description

Ultrametric phylogenetic tree of the 6 extant Phocoenidae (porpoise) species

Usage

data(Phocoenidae)

Details

This phylogeny was extracted from Steeman et al. Syst Bio 2009 cetacean phylogeny

References


Examples

data(Phocoenidae)
print(Phocoenidae)
plot(Phocoenidae)

---

phyl.pca_pl  

Regularized Phylogenetic Principal Component Analysis (PCA).

Description

Performs a principal component analysis (PCA) on a regularized evolutionary variance-covariance matrix obtained using the fit_t_pl function.

Usage

phyl.pca_pl(object, plot=TRUE, ...)

Arguments

object A penalized likelihood model fit obtained by the \texttt{fit_t_pl} function.

plot Plot of the PC’s axes. Default is TRUE (see details).

... Options to be passed through. (e.g., axes=c(1,2), col, pch, cex, mode="cov" or "corr", etc.)

Details

\texttt{phyl.pca_pl} allows computing a phylogenetic principal component analysis (following Revell 2009) using a regularized evolutionary variance-covariance matrix from penalized likelihood models fit to high-dimensional datasets (where the number of variables \( p \) is potentially larger than \( n \); see details for the models options in \texttt{fit_t_pl}). Models estimates are more accurate than maximum likelihood methods, particularly in the high-dimensional case. Plotting options, the number of axes to display (axes=c(1,2) is the default), and whether the covariance (mode="cov") or correlation (mode="corr") should be used can be specified through the ellipsis "..." argument.

Value

a list with the following components

values the eigenvalues of the evolutionary variance-covariance matrix

scores the PC scores

loadings the component loadings

nodes_scores the scores for the ancestral states at the nodes (projected on the space of the tips)

mean the mean/ancestral value used to center the data

vectors the eigenvectors of the evolutionary variance-covariance matrix

Note

Contrary to conventional PCA, the principal axes of the phylogenetic PCA are not orthogonal, they represent the main axes of (independent) evolutionary changes.

Author(s)

J. Clavel

References


See Also

\texttt{fit_t_pl}, \texttt{ancestral}, \texttt{GIC.fit_pl.rpanda}, \texttt{gic_criterion}
Examples

```r
require(mvMORPH)
set.seed(1)
n <- 32 # number of species
p <- 31 # number of traits
tree <- pbtree(n=n) # phylogenetic tree
R <- Posdef(p) # a random symmetric matrix (covariance)

# simulate a dataset
Y <- mvSIM(tree, model="BM1", nsim=1, param=list(sigma=R))

# fit a multivariate Pagel lambda model with Penalized likelihood
fit <- fit_t_pl(Y, tree, model="lambda", method="RidgeAlt")

# Perform a phylogenetic PCA using the model fit (Pagel lambda model)
pca_results <- phyl.pca_pl(fit, plot=T)

# retrieve the scores
head(pca_results$scores)
```

---

**Phyllostomidae**  
*Phyllostomidae phylogeny*

**Description**

Ultrametric phylogenetic tree of 150 of the 165 extant known Phyllostomidae species

**Usage**

data(Phyllostomidae)

**Details**

This phylogeny is the maximum clade credibility tree used in Rolland et al. (2014), which originally comes from the Bininda-Emonds tree (Bininda-Emonds et al. 2007)

**References**


See Also

Phyllostomidae_genera

Examples

data(Phyllostomidae)
print(Phyllostomidae)
#plot(Phyllostomidae)

Phyllostomidae_genera  Phylogenies of Phyllostomidae genera

Description

List of 25 ultrametric phylogenetic trees corresponding to 25 Phyllostomidae genera

Usage

data(Phyllostomidae_genera)

See Also

Phyllostomidae

Examples

data(Phyllostomidae_genera)
print(Phyllostomidae_genera)

plot.fit_t.env  Plot the phenotypic evolutionary rate through time estimated by the fit_t_env function

Description

Plot estimated evolutionary rate as a function of the environmental data and time.

Usage

## S3 method for class 'fit_t.env'
plot(x, steps = 100, ...)
Arguments

- **x**: an object of class 'fit_t.env' obtained from a `fit_t.env` fit.
- **steps**: the number of steps from the root to the present used to compute the evolutionary rate $\sigma^2$ through time.
- **...**: further arguments to be passed to `plot`. See ?plot.

Value

`plot.fit_t.env` returns invisibly a list with the following components used in the current plot:

- **time_steps**: the times steps where the climatic function was evaluated to compute the rate. The number of steps is controlled through the argument steps.
- **rates**: the estimated evolutionary rate through time estimated at each time_steps

Note

All the graphical parameters (see `par`) can be passed through (e.g. line type: `lty`, line width: `lwd`, color: `col` ...)

Author(s)

J. Clavel

References


See Also

`lines.fit_t.env`, `likelihood_t_env`

Examples

data(Cetacea)
data(InfTemp)

# Simulate a trait with temperature dependence on the Cetacean tree
set.seed(123)

trait <- sim_t_env(Cetacea, param=c(0.1,0.2), env_data=InfTemp, model="EnvExp", root.value=0, step=0.01, plot=TRUE)

## Fit the Environmental-exponential model

result1=fit_t_env(Cetacea, trait, env_data=InfTemp, scale=TRUE)

plot(result1)
# further options
plot(result1, lty=2, lwd=2, col="red")

---

**plot_BICompare**

*Display modalities on a phylogeny.*

**Description**

Plot a phylogeny with branches colored according to modalities

**Usage**

`plot_BICompare(phylo, BICompare)`

**Arguments**

- `phylo`: an object of type 'phylo' (see ape documentation)
- `BICompare`: an object of class 'BICompare', output of the 'BICompare' function

**Value**

a plot of the phylogeny with branches colored according to which modalities they belong to.

**Author(s)**

E Lewitus

**References**

Lewitus, E., Morlon, H., Characterizing and comparing phylogenies from their Laplacian spectrum, bioRxiv doi: http://dx.doi.org/10.1101/026476

**See Also**

`BICompare`

**Examples**

```r
data(Cetacea)
#result <- BICompare(Cetacea,5)
#plot_BICompare(Cetacea,result)
```
**plot_ClaDS0_chains**

Plot the MCMC chains obtained when infering ClaDS0 parameters.

**Description**

Plot the MCMC chains obtained with run_ClaDS0.

**Usage**

```r
plot_ClaDS0_chains(sampler, burn = 1/2, thin = 1,
                  param = c("sigma", "alpha", "l_0", "LP"))
```

**Arguments**

- `sampler`: The output of a run_ClaDS0 run.
- `burn`: Number of iterations to drop in the beginning of the chains.
- `thin`: Thinning parameter, one iteration out of "thin" is plotted.
- `param`: Either a vector of "character" elements with the name of the parameter to plot, or a vector of integers indicating what parameters to plot.

**Author(s)**

O. Maliet

**References**


**See Also**

`fit_ClaDS0`, `getMAPS_ClaDS0`, `plot_ClaDS_chains`

**Examples**

```r
data("ClaDS0_example")

plot_ClaDS0_chains(ClaDS0_example$C10_chains)
plot_ClaDS0_chains(ClaDS0_example$C10_chains, param = paste0("lambda_", c(1,10,5)))
```
plot_ClaDS_chains

Plot the MCMC chains obtained when inferring ClaDS parameters

Description
Plot the MCMC chains obtained with fit_ClaDS.

Usage
plot_ClaDS_chains(sampler, burn = 1/2, thin = 1,
                   param = c("sigma", "alpha", "mu", "LP"))

Arguments
- sampler: The output of a fit_ClaDS run.
- burn: Number of iterations to drop in the beginning of the chains.
- thin: Thinning parameter, one iteration out of "thin" is plotted.
- param: Either a vector of "character" elements with the name of the parameter to plot,
or a vector of integers indicating what parameters to plot.

Author(s)
O. Maliet

References

See Also
fit_ClaDS, getMAPS_ClaDS, plot_ClaDS0_chains

Examples
data("Caprimulgidae_ClaDS2")

plot_ClaDS_chains(Caprimulgidae_ClaDS2$sampler)

plot_ClaDS_chains(Caprimulgidae_ClaDS2$sampler, burn = 1/4,
                   param = c("sigma", "alpha", "l_0", "LP"))

plot_ClaDS_chains(Caprimulgidae_ClaDS2$sampler, burn = 1/5, thin = 5, param = c(1,5,6,15))
plot_ClaDS_phylo

Plot a phylogeny with branch-specific values

Description
Plot a phylogeny with branches colored according to branch-specific rate values

Usage
plot_ClaDS_phylo(phylo, rates, rates2 = NULL,
                  same.scale = T, main = NULL, lwd = 2, log = T, show.tip.label = F, ...)

Arguments
phylo An object of class 'phylo'.
rates A vector containing the branch-specific rates, in the same order as phylo$edges.
rates2 An optional second vector containing the branch-specific rates, in the same order as phylo$edges. If NULL (the default), the tree is only plotted once with the rate values from rates. If not, the tree is plotted twice, with the rate values from rates in the left panel and those from rates2 in the right panel.
same.scale A boolean specifying whether the values from rates and rates2 are plotted with the same colorscale. Default to TRUE.
main A title for the plot.
lwd Width of the tree branch lengths. Default to 2.
log A boolean specifying whether the rates values are plotted on a log scale. Default to TRUE.
show.tip.label A boolean specifying whether the labels of the phylogeny should be displayed. Default to FALSE.
... Optional arguments for plot.phylo.

Author(s)
O. Maliet

References

See Also
sim_ClaDS
Examples

```r
set.seed(1)

obj = sim_ClaDS(lamb_0=0.1,
                 mu_0=0.5,
                 sigma_lamb=0.7,
                 alpha_lamb=0.90,
                 condition="taxa",
                 taxa_stop = 20,
                 prune_extinct = TRUE)

tree = obj$tree
speciation_rates = obj$lamb[obj$rates]
extinction_rates = obj$mu[obj$rates]

plot_ClaDS_phylo(tree,speciation_rates)
plot_ClaDS_phylo(tree,speciation_rates, lwd = 4, log = FALSE)
```

---

### plot_dtt

*Plot diversity through time*

**Description**

Plot the estimated number of species through time

**Usage**

```r
plot_dtt(fit.bd, tot_time, N0)
```

**Arguments**

- **fit.bd**: an object of class 'fit.bd', output of the 'fit_bd' function
- **tot_time**: the age of the underlying phylogeny (crown age, or stem age if known). If working with crown ages, tot_time is given by max(node.age(phylo)$ages).
- **N0**: number of extant species. If all extant species are represented in the phylogeny, N0 is given by length(phylo$tip.label)

**Value**

Plot representing how the estimated number of species vary through time

**Author(s)**

H Morlon
References

See Also
fit_bd

Examples

data(Balaenopteridae)
tot_time=max(node.age(Balaenopteridae)$ages)

# Fit the pure birth model (no extinction) with exponential variation of the speciation rate
# with time
f.lamb <-function(t,y){y[1] * exp(y[2] * t)}
f.mu<-function(t,y){0}
lamb_par<-c(0.08, 0.01)
mu_par<-c()
result <- fit_bd(Balaenopteridae,tot_time,f.lamb,f.mu,lamb_par,mu_par,f=1,
expo.lamb = TRUE, fix.mu=TRUE)

# plot estimated number of species through time
# plot_dtt(result, tot_time, N0=9)

---

plot_fit_bd

Plot speciation, extinction & net diversification rate functions of a fitted model

Description
Plot estimated speciation, extinction & net diversification rates through time

Usage
plot_fit_bd(fit.bd, tot_time)

Arguments

| fit.bd   | an object of class 'fit.bd', output of the 'fit_bd' function |
| tot_time | the age of the phylogeny (crown age, or stem age if known). If working with crown ages, tot_time is given by max(node.age(phylo)$ages). |

Value
Plots representing how the estimated speciation, extinction & net diversification rate functions vary through time
Author(s)
H Morlon

See Also
fit_bd

Examples

data(Balaenopteridae)
tot_time<-max(node.age(Balaenopteridae)$ages)

# Fit the pure birth model (no extinction) with exponential variation of the speciation rate
# with time
f.lamb <-function(t,y){y[1] * exp(y[2] * t)}
f.mu<-function(t,y){0}
lamb_par<-c(0.08, 0.01)
mu_par<-c()
result <- fit_bd(Balaenopteridae,tot_time,f.lamb,f.mu,lamb_par,mu_par,
               expo.lamb = TRUE, fix.mu=TRUE)

# plot fitted rates
#plot_fit_bd(result, tot_time)

plot_fit_env

Plot speciation, extinction & net diversification rate functions of a fitted environmental model

Description
Plot estimated speciation, extinction & net diversification rates as a function of the environmental data and time

Usage
plot_fit_env(fit.env, env_data, tot_time)

Arguments
fit.env an object of class 'fit.env', output of the 'fit_env' function
env_data environmental data, given as a data frame with two columns. The first column is time, the second column is the environmental data (temperature for instance).
tot_time the age of the phylogeny (crown age, or stem age if known). If working with crown ages, tot_time is given by max(node.age(phylo)$ages).

Value
Plots representing how the estimated speciation, extinction & net diversification rate functions vary as a function of the environmental data & time
plot_prob_dtt

**Plot diversity through time with confidence intervals.**

**Description**

Plots confidence intervals of the estimated number of species through time using a matrix of probabilities given by the function `prob_dtt`.

**Usage**

```r
plot_prob_dtt(mat, grain = 0.1, plot.prob = TRUE, plot.mean = TRUE, int = TRUE, plot.bound = FALSE, conf = 0.95, add = FALSE, col.mean = "red", col.bound = "blue", lty = "solid", lwd = 1)
```

**Arguments**

- **mat**
  - matrix of probabilities, with species numbers as rows and times as columns with rownames and colnames set to the values of each.

- **grain**
  - the upper limit of a range of probabilities plotted in a gray scale (lower limit is zero). Higher probabilities are plotted in black. Default value is 0.1.
The function assumes that the matrix of probabilities `mat` has species numbers as rows and times as columns with rownames and colnames set to the values of each.

'Grain' must be between 0 and 1. If the plot is too pale 'grain' should be diminished (and inversely if the plot is too dark)

Value

Plot representing how the estimated number of species vary through time with confidence intervals. The darker is the plot, the higher is the probability.

Author(s)

O.Billaud, T.L.Parsons, D.S.Moen, H.Morlon

References


Billaud, O., Moen, D. S., Parsons, T. L., Morlon, H. (under review) Estimating Diversity Through Time using Molecular Phylogenies: Old and Species-Poor Frog Families are the Remnants of a Diverse Past.

See Also

`fit_bd, plot_dtt, prob_dtt`

Examples

data(Balaenopteridae)
tot_time<-max(node.age(Balaenopteridae)$ages)

# Fit the pure birth model (no extinction) with exponential variation of the speciation rate
# with time
f.lamb <- function(t, y) {y[1] * exp(y[2] * t)}
f.mu <- function(t, y) {0}
lamb_par <- c(0.08, 0.01)
mu_par <- c()
result <- fit_bd(Balaenopteridae, tot_time, f.lamb, f.mu, lamb_par, mu_par, f=1,
                 expo.lamb = TRUE, fix.mu=TRUE)

# Compute the matrix of probabilities
prob <- prob_dtt(result, tot_time, 1:tot_time, N0=9, type="crown")

# Check that the sums of probabilities are equal to 1
colSums(prob)

# Plot Diversity through time
plot_prob_dtt(prob)

---

plot_spectR

**Spectral density plot of a phylogeny.**

**Description**

Plot the spectral density of a phylogeny and all eigenvalues ranked in descending order.

**Usage**

`plot_spectR(spectR)`

**Arguments**

- `spectR`: an object of class 'spectR', output of the 'spectR' function

**Value**

A 2-panel plot with the spectral density profile on the first panel and the eigenvalues ranked in descending order on the second panel

**Author(s)**

E Lewitus

**References**

Lewitus, E., Morlon, H., Characterizing and comparing phylogenies from their Laplacian spectrum, bioRxiv doi: http://dx.doi.org/10.1101/026476

**See Also**

- `spectR`
Examples

```r
data(Cetacea)
result <- spectR(Cetacea)
#plot_spectR(result)
```

---

**Posdef**

*Positive definite symmetric matrices*

**Description**

Generates a positive definite and symmetric matrix with specified eigen-values

**Usage**

```r
Posdef(p, ev = rexp(p, 1/100))
```

**Arguments**

- **p**: The dimension of the matrix
- **ev**: The eigenvalues. If not specified, eigenvalues are taken from an exponential distribution.

**Details**

`Posdef` generates random positive definite covariance matrices with specified eigen-values that can be used to simulate multivariate datasets (see Uyeda et al. 2015 - and supplied R codes).

**Value**

Returns a symmetric positive-definite matrix with eigen-values = ev.

**Author(s)**

J. Clavel

**References**


prob_dtt

See Also

GIC.fit_pl.rpanda, fit_t_pl, phyl.pca_pl

Examples

```r
require(mvMORPH)
set.seed(123)
n <- 32  # number of species
p <- 40  # number of traits
tree <- pbtree(n=n)  # phylogenetic tree
R <- Posdef(p)  # a random symmetric matrix (covariance)
# simulate a dataset
Y <- mvSIM(tree, model="BM1", nsim=1, param=list(sigma=R))

test <- fit_t_pl(Y, tree, model="BM", method="RidgeAlt")
GIC(test)
```

Description

Returns a matrix of probabilities to have 'm' species at a given time 't' with 'n' observed extant species (complete sampling or not) and 's' species at the root of the phylogeny (s=1 if the tree has a stem, otherwise s=2)

Usage

```r
prob_dtt(fit.bd, tot_time, time, N0, l=N0, f = l/N0, m = seq(N0), method="simple", lin = FALSE, prec = 1000, type = "stem", logged = TRUE)
```

Arguments

- **fit.bd**: an object of class 'fit.bd', output of the 'fit_bd' function.
- **tot_time**: the age of the underlying phylogeny (crown age, or stem age if known). If working with crown ages, tot_time is given by max(node.age(phylo)$ages).
- **time**: vector of times on which the function calculates the probabilities of 'm' species. The function goes forward in time, so that t = 0 is the time of the most recent common ancestor.
- **N0**: number of extant species. If all extant species are represented in the phylogeny, N0 is given by length(phylo$tip.label).
- **l**: number of extant species sampled. Default value is N0 (complete sampling).
the fraction of extant species included in the phylogeny, given by \( l/N_0 \).

`m` a vector of integers for which we want to know the probability of each value.

`method` reflects which way of computing is chosen. A 'simple' one (quicker) is used when the number of extant species (\( N_0 \)) is known exactly or when the whole phylogeny is sampled (\( f=1 \)). A 'hard one', much longer, is used when \( N_0 \) is not known without doubt and \( f<1 \). The default value is "simple" (the other possibility is "hard")

`lin` logical: set to TRUE if \( \lambda \) & \( \mu \) are fitted with a linear model.

`prec` precision (number of bits used) of the computation. The default value is 1000.

`type` reflects whether the clade has a stem or not. Options are the default "stem" and the alternative "crown", which means the tree starts with two species at time 0.

`logged` logical: set to TRUE to log probabilities and factorials as much as possible (required, except perhaps for very small, young clades).

Details

If the sampling fraction is not equal to 1, the function computes with very high numbers. To be sufficiently accurate, the package 'Rmpfr' is used and "prec" is the precision of the computation. Hence, the calculation may take a lot of time. In case of wrong probabilities (negatives or higher than 1 for instance) you should increase the precision.

If the sampling fraction is equal to 1, the function doesn’t need the package 'Rmpfr' and simply uses the log of probabilities and factorials (argument "logged"). Thus, computation is faster.

The matrix columns names go backward in time.

Value

Matrix of probabilities to have 'm' species at a given time 't' with 'n' observed extant species (complete sampling or not).

Author(s)

O.Billaud, T.L.Parsons, D.S.Moen, H.Morlon

References


Billaud, O., Moen, D. S., Parsons, T. L., Morlon, H. (under review) Estimating Diversity Through Time using Molecular Phylogenies: Old and Species-Poor Frog Families are the Remnants of a Diverse Past.

See Also

`fit_bd`, `plot_dtt`, `plot_prob_dtt`
Examples

data(Balaenopteridae)

```r
tot_time<-max(node.age(Balaenopteridae)$ages)

# Fit the pure birth model (no extinction) with exponential variation of the speciation rate
# with time
f.lamb <-function(t,y){y[1] * exp(y[2] * t)}
f.mu<-function(t,y){0}
lamb_par<-c(0.08, 0.01)
mu_par<-c()
result <- fit_bd(Balaenopteridae,tot_time,f.lamb,f.mu,lamb_par,mu_par,f=1,
expo.lamb = TRUE, fix.mu=TRUE)

# Compute the matrix of probabilities
prob <- prob_dtt(result, tot_time, 1:tot_time, N0=9, type="crown")

# Check that the sums of probabilities are equal to 1
colSums(prob)
```

---

radiolaria Radiolaria diversity since the Jurassic

Description

Radiolaria fossil diversity since the Jurassic

Usage

data(sealevel)

Details

Radiolaria fossil diversity since the Jurassic compiled from the Neptune Database (Lazarus, 1994) and Paleobiology Database (https://paleobiodb.org/). Diversity curves are estimated at the genus level using shareholder quorum subsampling (Alroy, 2010) at two-million-year bins. The format is a dataframe with the two following variables:

- `age` a numeric vector corresponding to the geological age, in Myrs before the present
- `radiolaria` a numeric vector corresponding to the estimated ostracod change at that age

References


Examples

data(radiolaria)
plot(radiolaria)
Red algae diversity since the Jurassic

Red algae fossil diversity since the Jurassic compiled from the Neptune Database (Lazarus, 1994) and Paleobiology Database (https://paleobiodb.org/). Diversity curves are estimated at the genus level using shareholder quorum subsampling (Alroy, 2010) at two-million-year bins. The format is a dataframe with the two following variables:

age  a numeric vector corresponding to the geological age, in Myrs before the present
redalgae  a numeric vector corresponding to the estimated Red algae change at that age

References


Examples

data(redalgae)
plot(redalgae)

Sea level data since the Jurassic

Global sea level change since the Jurassic

Usage

data(sealevel)
Details

Eustatic sea level change since the Jurassic calculated by Miller et al. (2005) from satellite measurements, tide gauges, shoreline markers, reefs, atolls, oxygen isotopes, the flooding history of continental margins, cratons. The format is a dataframe with the two following variables:

- **age** a numeric vector corresponding to the geological age, in Myrs before the present
- **sea level** a numeric vector corresponding to the estimated sea level change at that age

References


Examples

data(sealevel)
plot(sealevel)

---

silica

Silica data across the Cenozoic

Description

Silica weathering ratio across the Cenozoic

Usage

data(silica)

Details

Silica weathering ratio across the Cenozoic calculated by Cermeno et al. (2015) using the lithium isotope record of seawater from Misra and Froelich (2012). The format is a dataframe with the two following variables:

- **age** a numeric vector corresponding to the geological age, in Myrs before the present
- **silica weathering ratio** a numeric vector corresponding to the estimated CO2 at that age

References


**Examples**

```r
data(silica)
plot(silica)
```

---

**sim.convergence.geo**  
*Simulation of trait data under the model of convergent character displacement described in Drury et al. 2017*

---

**Description**

Simulates the evolution of a continuous character that evolves depending on pairwise similarity in another, OU-evolving trait (e.g., a trait that covaries with resource use). `sig2` and `z0` are shared between two traits, `max` and `alpha` are for focal trait, OU parameters for non-focal trait.

**Usage**

```r
sim.convergence.geo(phylo, pars, Nsegments=2500, plot=FALSE, geo.object)
```

**Arguments**

- `phylo` : an object of type 'phylo' (see ape documentation)
- `pars` : A matrix with a number of rows corresponding to the desired number of simulations, columns containing values for `sig2` in [,1], `m` in [,2], `alpha` in [,3], `root.value` in [,4], `psi` of the OU model for the non-focal, resource use trait in [,5], and `theta` in the OU model for the non-focal resource use trait in [,6]
- `Nsegments` : the minimum number of time steps to simulate
- `plot` : if TRUE, returns two plots: the top plot is focal trait undergoing convergence, the bottom plot is non-focal trait evolving under BM or OU
- `geo.object` : geography object created using `CreateGeoObject`

**Details**

Adjusting `Nsegments` will impact the length of time the simulations take. The length of each segment `(max(nodeHeights(phylo))/Nsegments)` should be much smaller than the smallest branch `(min(phylo$edge.length))`.

**Value**

A list of two matrices with the simulated values for each lineage (one simulation per row; columns correspond to species) for trait1 (focal trait undergoing convergence) and non.focal (resource-use trait that determines strength of convergence in trait1)

**Author(s)**

J.P. Drury jonathan.p.drury@gmail.com
References


See Also

CreateGeoObject

Examples

data(Anolis.data)
phylo<-Anolis.data$phylo
geo.object<-Anolis.data$geography.object

#simulate with the OU process present and absent
pars<-expand.grid(0.05,-0.1,1,0,c(2,0),0)
sim.convergence.geo(phylo,pars,Nsegments=2500, plot=FALSE, geo.object)

sim.divergence.geo

Simulation of trait data under the model of divergent character displacement described in Drury et al. 2017

Description

Simulates the evolution of a continuous character under a model of evolution where trait values are repelled according to between-species similarity in trait values, taking into account biogeography using a biogeo.object formatted from RPANDA (see CreateGeoObject function in RPANDA package)

Usage

sim.divergence.geo(phylo,pars, Nsegments=2500, plot=FALSE, geo.object)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phylo</td>
<td>a phylogenetic tree</td>
</tr>
<tr>
<td>pars</td>
<td>A matrix with a number of rows corresponding to the desired number of simulations, columns containing values for ( \sigma^2 ) in ( [,1] ), ( m ) in ( [,2] ), ( \alpha ) in ( [,3] ), root.value in ( [,4] ), ( \psi ) of the OU model in ( [,5] ), and ( \theta ) in the OU model in ( [,6] )</td>
</tr>
<tr>
<td>Nsegments</td>
<td>the minimum number of time steps to simulate</td>
</tr>
<tr>
<td>plot</td>
<td>logical indicating whether to plot the simulated trait values at each time step</td>
</tr>
<tr>
<td>geo.object</td>
<td>geography object created using CreateGeoObject</td>
</tr>
</tbody>
</table>
Details

Adjusting Nsegments will impact the length of time the simulations take. The length of each segment \( \frac{\text{max}(\text{nodeHeights}(\text{phylo}))/\text{Nsegments}}{\text{min}(\text{phylo}$\text{edge.length})} \) should be much smaller than the smallest branch.

Value

A matrix with the simulated values for each lineage (one simulation per row; columns correspond to species)

Author(s)

J.P. Drury jonathan.p.drury@gmail.com F. Hartig

References


See Also

CreateGeoObject

Examples

data(Anolis.data)
phylo<-Anolis.data$phylo
geo.object<-Anolis.data$geography.object

#simulate with the OU process present and absent
pars<-expand.grid(0.05,2,1,0,c(2,0),0)
sim.divergence.geo(phylo,pars,Nsegments=2500, plot=FALSE, geo.object)
simulateTipData-methods

Arguments

object an object of class 'PhenotypicModel'.
params vector of parameters, given in the same order as in the 'model' object.
method an integer specifying the behaviour of the function. If method = 1 (default value), the tip distribution is first computed, before returning a simulated dataset drawn in this distribution. If method = 2, the whole trajectory is simulated step by step, plotted, and returned. Otherwise, the whole trajectory is simulated step by step, and then returned without being plotted.
v boolean specifying the verbose mode. Default value : FALSE.

Value

a vector of trait values at the tips of the tree.

Author(s)

M Manceau

References


Examples

#Loading an example tree
tree <- read.tree(text=newick)

#Creating the models
modelBM <- createModel(tree, 'BM')
modelOU <- createModel(tree, 'OU')

#Simulating tip traits under both models with distinct behaviours of the functions :
dataBM <- simulateTipData(modelBM, c(0,0,0,1))
dataOU <- simulateTipData(modelOU, c(0,0,1,5,1), method=1)
dataBM2 <- simulateTipData(modelBM, c(0,0,0,1), method=2)

 simulateTipData-methods

~~ Methods for Function simulateTipData ~~

Description

~~ Methods for function simulateTipData ~~
Methods

signature(object = "PhenotypicModel") This is the only method available for this function. Same behaviour for any PhenotypicModel.

---

**sim_ClaDS**

*Simulation of the ClaDS model*

---

**Description**

Simulate a birth-death phylo-geny with rate shifts happening at speciation events.

**Usage**

```r
sim_ClaDS(lambda_0, mu_0,
   new_lamb_law="lognormal*shift", new_mu_law="turnover",
   condition="time", time_stop = 0, taxa_stop = Inf,
   sigma_lamb=0.1, alpha_lamb=1, lamb_max=1, lamb_min=0,
   sigma_mu=0, alpha_mu=1, mu_min=mu_0, mu_max=mu_0,
   theta=1, nShiftMax=Inf,
   return_all_extinct=FALSE, prune_extinct=TRUE,
   maxRate=Inf)
```

**Arguments**

- **lambda_0**: Initial speciation rate.
- **mu_0**: Initial extinction rate, or turnover rate if new_mu_law == "turnover".
- **new_lamb_law**: Distribution in which the new speciation rates are drawn at a speciation event. See details.
- **new_mu_law**: Distribution in which the new extinction rates are drawn at a speciation event. See details.
- **condition**: Stoping condition. Can be "time" (the default) or "taxa".
- **time_stop**: Stoping time if condition == "time".
- **taxa_stop**: Final number of species if condition == "taxa".
  If condition == "time", the process is stoped if the number of species exceeds taxa_stop. This can be usefull for some parametrizations of the model for which the number of species can reach very large number very quickly, leading to computation time and memory issues. To disable this option, use taxa_stop = Inf (the default).
- **sigma_lamb**: Parameter of the new speciation rates distribution, see details.
- **alpha_lamb**: Parameter of the new speciation rates distribution, see details.
- **lamb_max**: Parameter of the new speciation rates distribution, see details.
- **lamb_min**: Parameter of the new speciation rates distribution, see details.
sim_ClaDS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sigma_mu</td>
<td>Parameter of the new extinction rates distribution, see details.</td>
</tr>
<tr>
<td>alpha_mu</td>
<td>Parameter of the new extinction rates distribution, see details.</td>
</tr>
<tr>
<td>mu_min</td>
<td>Parameter of the new extinction rates distribution, see details.</td>
</tr>
<tr>
<td>mu_max</td>
<td>Parameter of the new extinction rates distribution, see details.</td>
</tr>
<tr>
<td>theta</td>
<td>Probability to have a rate shift at speciation. Default to 1.</td>
</tr>
<tr>
<td>nShiftMax</td>
<td>Maximum number of rate shifts. If nShiftMax &lt; Inf, theta is set to 0 as soon as there has been nShiftMax rate shifts. Set nShiftMax = Inf (the default) to disable this option.</td>
</tr>
<tr>
<td>return_all_extinct</td>
<td>Boolean specifying whether the function should return extinct phylogenies. Default to FALSE.</td>
</tr>
<tr>
<td>prune_extinct</td>
<td>Boolean specifying whether extinct species should be removed from the resulting phylogeny. Default to TRUE.</td>
</tr>
<tr>
<td>maxRate</td>
<td>The process is stopped if one of the lineage has a speciation rate that exceeds maxRate. This can be useful for some parameterizations of the model for which the rates can reach very large values, leading to numerical overflows. To disable this option, use maxRate = Inf (the default).</td>
</tr>
</tbody>
</table>

**Details**

Available options for new_lamb_law are:

- "uniform", the new speciation rates are drawn uniformly in \([\text{lamb}\_\text{min}, \text{lamb}\_\text{max}]\).
- "normal", the new speciation rates are drawn in a normal distribution with parameters \((\text{sigma}\_\text{lamb}\^2, \text{parent}\_\text{lambda})\), truncated in 0.
- "lognormal", the new speciation rates are drawn in a lognormal distribution with parameters \((\text{sigma}\_\text{lamb}\^2, \text{parent}\_\text{lambda})\).
- "lognormal*shift", the new speciation rates are drawn in a lognormal distribution with parameters \((\text{sigma}\_\text{lamb}\^2, \text{parent}\_\text{lambda} * \text{alpha}\_\text{lamb})\). This is the default option as it corresponds to the ClaDS model.
- "lognormal*t", the new speciation rates are drawn in a lognormal distribution with parameters \((\text{sigma}\_\text{lamb}\^2 * \text{t}^2, \text{parent}\_\text{lambda})\), where \(t\) is the age of the mother species.
- "logbrownian", the new speciation rates are drawn in a lognormal distribution with parameters \((\text{sigma}\_\text{lamb}\^2 * \text{t}, \text{parent}\_\text{lambda})\), where \(t\) is the age of the mother species. This is used to approximate the case where speciation rates are evolving as the log of a brownian motion, as is done in Beaulieu, J. M. and B. C. O’Meara. (2015).
- "normal+shift", the new speciation rates are drawn in a normal distribution with parameters \((\text{sigma}\_\text{lamb}\^2, \text{parent}\_\text{lambda} + \text{alpha}\_\text{lamb})\), truncated in 0.
- "normal*shift", the new speciation rates are drawn in a normal distribution with parameters \((\text{sigma}\_\text{lamb}\^2, \text{parent}\_\text{lambda} * \text{alpha}\_\text{lamb})\), truncated in 0.

Available options for new_mu_law are:

- "uniform", the new extinction rates are drawn uniformly in \([\text{mu}\_\text{min}, \text{mu}\_\text{max}]\).
• "normal", the new extinction rates are drawn in a normal distribution with parameters (sigma_mu^2, parent_mu), truncated in 0.
• "lognormal", the new extinction rates are drawn in a lognormal distribution with parameters (sigma_mu^2, parent_mu).
• "lognormal*shift", the new extinction rates are drawn in a lognormal distribution with parameters (sigma_mu^2, parent_mu * alpha_mu).
• "normal*t", the new speciation rates are drawn in a normal distribution with parameters (sigma_lamb^2 * t^2, parent_lambda), where t is the age of the mother species.
• "turnover", the turnover rate is constant (in that case mu_0 is the turnover rate), so the new extinction rates are mu_0 times the new speciation rates. This is the default option, corresponding to ClaDS2.

Value

A list with:

- tree: The resulting phylogeny.
- times: A vector with the times of all speciation and extinction events.
- nblineages: A vector in which nblineages[i] is the number of species in the clade after the event happening at time times[i].
- lamb: A vector with all the different speciation rates resulting from the simulation.
- mu: A vector with all the different extinction rates resulting from the simulation.
- rates: A vector of integer mapping the elements of .$lamb and .$mu to the branches of .$tree.
- maxRate: A boolean indicating whether the process was ended before reaching the specified stopping criterion because one of the speciation rates exceeded maxRate (see the "arguments" section).
- root_length: The time before the first speciation event.

Author(s)

O. Maliet

References


See Also

plot_ClaDS_phylo
Examples

# Simulation of a ClaDS2 phylogeny
set.seed(1)
obj = sim_ClaDS(lambda_0=0.1,
    mu_0=0.5,
    sigma_lamb=0.7,
    alpha_lamb=0.90,
    condition="taxa",
    taxa_stop = 20,
    prune_extinct = TRUE)
tree = obj$tree
speciation_rates = obj$lamb[obj$rates]
extinction_rates = obj$mu[obj$rates]
plot_ClaDS_phylo(tree,speciation_rates)

# Simulation of a phylogeny with constant extinction rate and speciation
# rates evolving as a logbrownian
set.seed(4321)
obj = sim_ClaDS(lambda_0=0.1,
    mu_0=0.2,
    new_mu_law = "uniform",
    new_lamb_law = "logbrownian",
    sigma_lamb=0.4,
    condition="taxa",
    taxa_stop = 20,
    prune_extinct = FALSE)
tree = obj$tree
speciation_rates = obj$lamb[obj$rates]
extinction_rates = obj$mu[obj$rates]
plot_ClaDS_phylo(tree,speciation_rates)

# Simulation of a phylogeny with constant extinction rate and at most one shift
# in speciation rates
set.seed(1221)
obj = sim_ClaDS(lambda_0=0.1,
    mu_0=0.05,
    new_mu_law = "uniform",
    new_lamb_law = "uniform",
    lamb_max = 0.5, lamb_min = 0,
    theta = 0.1, nShiftMax = 1,
    condition="taxa",
    taxa_stop = 100,
tree = obj$tree
speciation_rates = obj$lamb[1:obj$rates]
extinction_rates = obj$mu[1:obj$rates]

plot_ClaDS_phylo(tree, speciation_rates)

### sim_env_bd

*Simulate birth-death tree dependent on an environmental curve*

**Description**

Simulates a birth-death tree (starting with one lineage) with speciation and/or extinction rate that varies as a function of an input environmental curve. Notations follow Morlon et al. PNAS 2011 and Condamine et al. ELE 2013.

**Usage**

```r
sim_env_bd(env_data, f.lamb, f.mu, lamb_par, mu_par, df=NULL, time.stop=0,
return.all.extinct=TRUE, prune.extinct=TRUE)
```

**Arguments**

- `env_data`: environmental data, given as a data frame with two columns. The first column is time, the second column is the environmental data (temperature for instance).
- `time.stop`: the age of the phylogeny.
- `f.lamb`: a function specifying the hypothesized functional form of the variation of the speciation rate $\lambda$ with time and the environmental variable. Any functional form may be used. This function has three arguments: the first argument is time; the second argument is the environmental variable; the third argument is a numeric vector of the parameters controlling the time and environmental variation (to be estimated).
- `f.mu`: a function specifying the hypothesized functional form of the variation of the extinction rate $\mu$ with time and the environmental variable. Any functional form may be used. This function has three arguments: the first argument is time; the second argument is the environmental variable; the third argument is a numeric vector of the parameters controlling the time and environmental variation (to be estimated).
- `lamb_par`: a numeric vector of initial values for the parameters of f.lamb to be estimated (these values are used by the optimization algorithm). The length of this vector is used to compute the total number of parameters in the model, so to fit a model with constant speciation rate (for example), lamb_par should be a vector of length 1. Otherwise aic values will be wrong.
mu_par: a numeric vector of initial values for the parameters of f.mu to be estimated (these values are used by the optimization algorithm). The length of this vector is used to compute the total number of parameters in the model, so to fit a model without extinction (for example), mu_par should be empty (vector of length 0). Otherwise aic values will be wrong.

df: the degree of freedom to use to define the spline. As a default, smooth.spline(env_data[,1], env_data[,2])$df is used. See sm.spline for details.

return.all.extinct: return all extinction lineages in simulated tree.

prune.extinct: prune extinct lineages in simulated tree.

Details

In the f.lamb and f.mu functions, time runs from the present to the past.

Value

a list with the following components

tree: the simulated tree with number tips

times: the times of speciation events starting from the past

nblineages: the labels of surviving lineages and total number of surviving lineages

Note

The speed of convergence of the fit might depend on the degree of freedom chosen to define the spline.

Author(s)

E Lewitus and H Morlon

References


See Also

fit_env, fit_bd
Examples

data(InfTemp)
dof<-smooth.spline(InfTemp[,1], InfTemp[,2])$df
# Simulates a tree with lambda varying as an exponential function of temperature
# and mu fixed to 0 (no extinction). Here t stands for time and x for temperature.
f.lamb <-function(t,x,y){y[1] * exp(y[2] * x)}
f.mu<-function(t,x,y){0}
lamb_par<-c(0.10, 0.01)
mu_par<-c()
#result_exp <- sim_env_bd(InfTemp,f.lamb,f.mu,lamb_par,mu_par,time.stop=10)

sim_sgd

Algorithm for simulating a phylogenetic tree under the SGD model

Description

Simulates a phylogeny arising from the SGD model with exponentially increasing metapopulation size. Notations follow Manceau et al. (2015).

Usage

sim_sgd(tau, b, d, nu)

Arguments

tau the simulation time, which corresponds to the length of the phylogeny
b the (constant) per-individual birth rate
d the (constant) per-individual death rate
nu the (constant) per-individual mutation rate

Value

a phylogenetic tree of class "phylo" (see ape documentation)

Author(s)

M Manceau

References

Examples

```r
tau <- 10
b <- 1e6
d <- b-.5
nu <- 0.6
tree <- sim_sgd(tau,b,d,nu)
plot(tree)
```

Description

Simulates datasets for a given phylogeny under matching competition (MC), diversity dependent linear (DDlin), or diversity dependent exponential (DDexp) models of trait evolution. Simulations are carried out from the root to the tip of the tree.

Usage

```r
sim_t_comp(phylo,pars,root.value,Nsegments=1000,model="MC,DDexp,DDlin")
```

Arguments

- **phylo**: an object of type ‘phylo’ (see ape documentation)
- **pars**: a vector containing the two parameters for the chosen model; all models require `sig2`, and additionally, the MC model requires `S`, specifying the level of competition (larger negative values correspond to higher levels of competition), the DDlin model requires `b` and DDexp require `r`, the slope parameters (negative in cases of decline in evolutionary rates with increasing diversity). `sig2` must be listed first.
- **root.value**: a number specifying the trait value for the ancestor
- **Nsegments**: a value specifying the total number of time segments to simulate across for the phylogeny (see Details)
- **model**: model chosen to fit trait data, "MC" is the matching competition model of Nuismer & Harmon 2014, "DDlin" is the diversity-dependent linear model, and "DDexp" is the diversity-dependent exponential model of Weir & Mursleen 2013.

Details

Adjusting `Nsegments` will impact the length of time the simulations take. The length of each segment (`max(nodeHeights(phylo))/Nsegments`) should be much smaller than the smallest branch (`min(phylo$edge.length)`).
**Value**

A named vector with simulated trait values for *n* species in the phylogeny.

**Author(s)**

J Drury jonathan.p.drury@gmail.com

**References**


**See Also**

*fit_t_comp*

**Examples**

```r
data(Cetacea)

# Simulate data under the matching competition model
MC.data<-sim_t_env(Cetacea,pars=c(sig2=0.01,S=-0.1),root.value=0,Nsegments=1000,model="MC")

# Simulate data under the diversity dependent linear model
DDlin.data<-sim_t_env(Cetacea,pars=c(sig2=0.01,b=-0.0001),root.value=0,Nsegments=1000,
model="DDlin")

# Simulate data under the diversity dependent linear model
DDexp.data<-sim_t_env(Cetacea,pars=c(sig2=0.01,r=-0.01),root.value=0,Nsegments=1000,model="DDexp")
```

---

**sim_t_env**  
Recursive simulation (root-to-tip) of the environmental model

**Description**

Simulates datasets for a given phylogeny under the environmental model (see ?fit_t_env)
Usage

\[ \text{sim}_t\_\text{env}(\text{phylo}, \text{param}, \text{env}\_\text{data}, \text{model}, \text{root}.\text{value}=0, \text{step}=0.001, \text{plot}=\text{FALSE}, \ldots) \]

Arguments

- **phylo**: An object of class 'phylo' (see ape documentation)
- **param**: A numeric vector of parameters for the user-defined climatic model. For the EnvExp and EnvLin, there is only two parameters. The first is sigma and the second beta.
- **env_data**: Environmental data, given as a time continuous function (see, e.g. splinefun) or a data frame with two columns. The first column is time, the second column is the environmental data (temperature for instance).
- **model**: The model describing the functional form of variation of the evolutionary rate \( \sigma^2 \) with time and the environmental variable. Default models are "EnvExp" and "EnvLin" (see details). An user defined function of any functional form can be used (forward in time). This function has three arguments: the first argument is time; the second argument is the environmental variable; the third argument is a numeric vector of the parameters controlling the time and environmental variation (to be estimated). See the example below.
- **root.value**: A number specifying the trait value for the ancestor
- **step**: This argument describe the length of the segments to simulate across for the phylogeny. The smaller is the segment, the greater is the accuracy of the simulation at the expense of the computation time.
- **plot**: If TRUE, the simulated process is plotted.
- **...**: Arguments to be passed through. For instance, "col" for plot=TRUE.

Details

The users defined function is simulated forward in time i.e.: from the root to the tips. The speed of the simulations might depend on the value used for the "step" argument. It’s possible to estimate the traits with the MLE from another fitted object (see the example below).

Value

A named vector with simulated trait values for \( n \) species in the phylogeny

Author(s)

J. Clavel

References

See Also

plot.fit_t.env, likelihood_t_env

Examples

data(Cetacea)
data(InfTemp)

set.seed(123)
# define the parameters
param <- c(0.1, -0.5)
# define the environmental function
my_fun <- function(t, env, param){ param[1]*exp(param[2]*env(t))}

# simulate the trait
trait <- sim_t_env(Cetacea, param=param, env_data=InfTemp, model=my_fun, root.value=0, step=0.001, plot=TRUE)

# fit the model to the simulated trait.
fit <- fit_t_env(Cetacea, trait, env_data=InfTemp, model=my_fun, param=c(0.1,0))
fit

# Then use the results from the previous fit to simulate a new dataset
trait2 <- sim_t_env(Cetacea, param=fit, step=0.001, plot=TRUE)
fit2 <- fit_t_env(Cetacea, trait2, env_data=InfTemp, model=my_fun, param=c(0.1,0))
fit2

# When providing the environmental function:
require(pspline)
spline_result <- sm.spline(x=InfTemp[,1],y=InfTemp[,2], df=50)
env_func <- function(t){predict(spline_result,t)}
t<-unique(InfTemp[,1])

# We build the interpolated smoothing spline function
env_data<-splinefun(t,env_func(t))

# provide the environmental function to simulate the traits
trait3 <- sim_t_env(Cetacea, param=param, env_data=env_data, model=my_fun, root.value=0, step=0.001, plot=TRUE)
fit3 <- fit_t_env(Cetacea, trait3, env_data=InfTemp, model=my_fun, param=c(0.1,0))
fit3
Description

Computes the spectra of eigenvalues for the modified graph Laplacian of a phylogenetic tree, identifies the spectral gap, then convolves the eigenvalues with a Gaussian kernel, and plots them alongside all eigenvalues ranked in descending order.

Usage

spectR(phylo, meth=c("standard"), zero_bound=F)

Arguments

phylo  an object of type 'phylo' (see ape documentation)
meth  the method used to compute the spectral density, which can either be "standard" or "normal". If set to "standard", computes the unnormalized version of the spectral density. If set to "normal", computes the spectral density normalized to the degree matrix (see the associated paper for an explanation)
zero_bound  if false, eigenvalues less than one are discarded

Details

Note that the eigengap should in principle be computed with the "standard" option

Value

a list with the following components:
eigenvalues  the vector of eigenvalues
principal_eigenvalue  the largest (or principal) eigenvalue of the spectral density profile
asymmetry  the skewness of the spectral density profile
peak_height  the largest y-axis value of the spectral density profile
eigengap  the position of the largest difference between eigenvalues, giving the number of modalities in the tree

Author(s)

E Lewitus

References

Lewitus, E., Morlon, H., Characterizing and comparing phylogenies from their Laplacian spectrum, bioRxiv doi: http://dx.doi.org/10.1101/026476

See Also

plot_spectR, JSDtree, BICompare
**Examples**

data(Cetacea)
spectR(Cetacea, meth="standard", zero_bound=FALSE)

---

**spectR_t**

*Spectral density plot of phylogenetic trait data*

---

**Description**

Computes the spectra of eigenvalues for the modified graph Laplacian of a phylogenetic tree with associated tip data, convolves the eigenvalues with a Gaussian kernel and plots the density profile of eigenvalues, and estimates the summary statistics of the profile.

**Usage**

```r
spectR_t(phylo, dat, draw=F)
```

**Arguments**

- **phylo**: an object of type 'phylo' (see ape documentation)
- **dat**: a vector of trait data associated with the tips of the phylo object; tips and trait data should be aligned
- **draw**: if true, the spectral density profile of the phylogenetic trait data is plotted

**Value**

a list with the following components:

- **eigenvalues**: the vector of eigenvalues
- **splitter**: the largest (or principal) eigenvalue of the spectral density profile
- **fragmenter**: the skewness of the spectral density profile
- **tracer**: the largest y-axis value of the spectral density profile

**Author(s)**

E Lewitus

**References**


**Examples**

```r
tr<-rtree(10)
dat<-runif(10,1,2)
spectR_t(tr, dat, draw=TRUE)
```
Index

+Topic classes
  PhenotypicACDC-class, 87
  PhenotypicADIag-class, 89
  PhenotypicBM-class, 90
  PhenotypicDD-class, 91
  PhenotypicGMM-class, 92
  PhenotypicModel-class, 93
  PhenotypicOU-class, 94
  PhenotypicPM-class, 96

+Topic datasets
  Balaenopteridae, 8
  Calomys, 10
  Cetacea, 12
  co2, 14
  co2_res, 15
  coccolithophore, 16
  d13c, 25
  foraminifera, 54
  greenalgea, 64
  InfTemp, 65
  landplant, 69
  ostracoda, 87
  Phocoenidae, 97
  Phyllostomidae, 99
  Phyllostomidae_genera, 100
  radiolaria, 115
  redalga, 116
  sealevel, 116
  silica, 117

+Topic methods
  fitTipData-methods, 27
  fitTipData,PhenotypicModel-method
  [PhenotypicModel-class], 93
  [PhenotypicModel,ANY,ANY,ANY-method]
  getDataLikelihood-methods, 56
  getTipDistribution-methods, 60
  modelSelection-methods, 86
  simulateTipData-methods, 121

ancestral, 5, 54, 98
Anolis.data, 7
Balaenopteridae, 8
BGB.examples, 8
BICompare, 9, 66, 102, 133
Calomys, 10
Caprimulgidae, 11
Caprimulgidae_ClaDS2, 11
Cetacea, 12
ClaDS0_example, 13
co2, 14
co2_res, 15
coccolithophore, 16
CreateClassObject, 16, 19
CreateGeoByClassObject, 16, 17, 18, 47, 75
CreateGeoObject, 7, 19, 22, 43, 45, 74–76,
  78, 79, 82, 83, 118–120
CreateGeoObject_BioGeoBEARS, 9, 19, 21
createModel, 23
createModelCoevolution, 24
d13c, 25
fit_bd, 27, 41, 107, 108, 110, 114, 127
fit_ClaDS, 12, 31, 31, 34, 57, 86, 104
fit_ClaDS0, 13, 33, 33, 58, 103
fit_coal_cst, 35, 39
fit_coal_var, 37, 37
fit_env, 39, 109, 127
fit_sgd, 42
fit_t_comp, 20–22, 43, 76, 78, 79, 82, 83, 130
fit_t_comp_subgroup, 17–19, 45
fit_t_env, 6, 48, 80
fit_t_pl, 6, 52, 61, 63, 98, 113
fitTipData, 26
fitTipData,PhenotypicModel-method
  (fitTipData-methods), 27
fitTipData-methods, 27
**INDEX**

foraminifera, 54
getDataLikelihood, 55
getDataLikelihood, PhenotypicModel-method
getDataLikelihood-methods, 56
getTipDistribution, 59
getTipDistribution, PhenotypicACDC-method
getTipDistribution, PhenotypicADIag-method
getTipDistribution, PhenotypicBM-method
getTipDistribution, PhenotypicDD-method
getTipDistribution, PhenotypicGMM-method
getTipDistribution, PhenotypicModel-method
getTipDistribution, PhenotypicOU-method
getTipDistribution, PhenotypicPM-method
GIC, 6
GIC.fit_pl.rpanda, 54, 61, 63, 98, 113
gic_criterion, 6, 54, 61, 62, 98
greenalgae, 64
InfTemp, 65
JSData_cluster, 67
JSData_tree, 10, 65, 67, 133
JSData_cluster, 66, 66
landplant, 69
likelihood_bd, 30, 41, 69
likelihood_coal_cst, 37, 71
likelihood_coal_var, 39, 72
likelihood_sgd, 43, 73
likelihood_subgroup_model, 47, 74
likelihood_t_DD, 45, 76, 77, 79
likelihood_t_DD_geog, 45, 78, 78
likelihood_t_env, 50, 79, 84, 101, 132
likelihood_t_MC, 45, 81, 83
likelihood_t_MC_geog, 45, 82, 82
lines.fit_t.env, 84, 101
make.simmap, 17, 18, 20, 46
modelSelection, 85
modelSelection, PhenotypicModel-method
modelSelection-methods, 86
mPhyFFT, 86
mvglr, 54, 61
optim, 44, 46
ostracoda, 87
par, 84, 101
PhenotypicACDC-class, 87
PhenotypicADIag-class, 89
PhenotypicBM-class, 90
PhenotypicDD-class, 91
PhenotypicGMM-class, 92
PhenotypicModel, 88–91, 93, 95, 96
PhenotypicModel-class, 93
PhenotypicOU-class, 94
PhenotypicPM-class, 96
Phocoenidae, 97
phyl.pca_pl, 6, 54, 97, 113
Phyllostomidae, 99, 100
Phyllostomidae_genera, 100, 100
plot.fit_t.env, 50, 84, 100, 132
plot_BIC_compare, 10, 102
plot_ClaDS0_chains, 34, 58, 103, 104
plot_ClaDS_chains, 12, 33, 57, 103, 104
plot_ClaDS_phylo, 105, 124
plot_dtt, 30, 106, 110, 114
plot_fit_bd, 30, 107
plot_fit_env, 41, 108
plot_prob_dtt, 109, 114
plot_spectR, 111, 133
Posdef, 112
print, PhenotypicModel-method
(PhenotypicModel-class), 93
prob_dtt, 110, 113
radiolaria, 115
redalgae, 116
round, 17
RPANDA (RPANDA-package), 4
RPANDA-package, 4
sealevel, 116
show, PhenotypicModel-method
   (PhenotypicModel-class), 93
silica, 117
sim.convergence.geo, 118
sim.divergence.geo, 119
sim_ClaDS, 105, 122
sim_env_bd, 126
sim_sgd, 128
sim_t_comp, 45, 129
sim_t_env, 130
simulateTipData, 120
simulateTipData, PhenotypicModel-method
   (simulateTipData-methods), 121
simulateTipData-methods, 121
spectR, 10, 66, 111, 132
spectR_t, 68, 134