Package ‘l1ou’

June 18, 2017

Version 1.41

Date 2015-12-05

Title Tools for detecting past changes in the expected mean trait values and studying trait evolution from comparative data

Author Mohammad Khabbazian [aut], Cecile Ane [ctb], Qing Yu (Sabrina) [ctb]

Maintainer Mohammad Khabbazian <khabbazian@wisc.edu>

Description Provides functions to study trait evolution from comparative data and detect past changes in the expected mean trait values. It uses the Ornstein-Uhlenbeck process, which can model a changing adaptive landscape over time and over lineages. Detection of evolutionary shifts in trait evolution from extant taxa is motivated by the study of convergent evolution, or to correlate shifts in traits with habitat changes or with changes in other phenotypes.

License GPL (>= 3)

URL https://github.com/khabbazian/l1ou

Encoding UTF-8

NeedsCompilation yes

Depends R (>= 3.1.0),
igraph,
ape,
phylolm (>= 2.0.0),
lars,
parallel,
grplasso,
magic,
genlasso

Imports Rcpp

LinkingTo Rcpp

VignetteBuilder knitr

Suggests knitr,
rmarkdown

RoxygenNote 5.0.1

R topics documented:

`adjust_data` *Adjusts the tree and traits to meet the requirements of* `estimate_shift_configuration`

Description

Returns a new tree and new data matrix, where the tree edges are in postorder, and the data row names match the order of the tree tip labels.

Usage

```
adjust_data(tree, Y, normalize = TRUE, quietly = FALSE)
```

Arguments

`tree`	ultrametric tree of class phylo with branch lengths.
`Y`	trait vector/matrix without missing entries.
`normalize`	logical. If TRUE, normalizes branch lengths to a unit tree height.
`quietly`	logical. If FALSE, changes in tree/trait are printed.

Value

`tree`	tree of class phylo, with the same topology as the input `tree` but adjusted edge order.
`Y`	trait vector/matrix with adjusted row names and row order.

Examples

```
data(lizard.tree, lizard.traits)
# here, lizard.traits is a matrix, so columns retain row names:
names(lizard.traits[,1])
lizard <- adjust_data(lizard.tree, lizard.traits[,1])

# for a data frame, make sure to retain row names if a single column is selected:
lizard.traits <- as.data.frame(lizard.traits)
lizard <- adjust_data(lizard.tree, subset(lizard.traits, select=1))
```

`configuration_ic` *Computes the information criterion score for a given configuration*

Description

Computes the information criterion score for a given configuration

Usage

```
configuration_ic(tree, Y, shift.configuration, criterion = c("pBIC",
  "pBICess", "mBIC", "BIC", "AICc"), root.model = c("OUfixedRoot",
  "OUrandomRoot"), alpha.starting.value = NA,
  alpha.upper = alpha_upper_bound(tree), alpha.lower = NA,
  fit.OU.model = FALSE, l1ou.options = NA)
```

Arguments

`tree`	ultrametric tree of class phylo, with branch lengths, and edges in postorder.
`Y`	trait vector/matrix without missing entries. The row names of the data must be in the same order as the tip labels.
`shift.configuration`	shift positions, i.e. vector of indices of the edges where the shifts occur.
`criterion`	an information criterion (see Details).
`root.model`	an ancestral state model at the root.
`alpha.starting.value`	optional starting value for the optimization of the phylogenetic adaptation rate.
`alpha.upper`	optional upper bound for the phylogenetic adaptation rate. The default value is log(2) over the minimum length of external branches, corresponding to a half life greater or equal to the minimum external branch length.
`alpha.lower`	optional lower bound for the phylogenetic adaptation rate.
`fit.OU.model`	logical. If TRUE, it returns an object of class l1ou with all the parameters estimated.
`l1ou.options`	if provided, all the default values will be ignored.

Details

AICc gives the usual small-sample size modification of AIC. BIC gives the usual Bayesian information criterion, here penalizing each shift as 2 parameters. mBIC is the modified BIC proposed by Ho and Ané (2014). pBIC is the phylogenetic BIC for shifts proposed by Khabbazian et al. pBICess is a version of pBIC where the determinant term is replaced by a sum of the log of effective sample sizes (ESS), similar to the ESS proposed by Ané (2008).

Value

Information criterion value of the given shift configuration.

References

Cécile Ané, 2008. "Analysis of comparative data with hierarchical autocorrelation". Annals of Applied Statistics 2(3):1078-1102.

Ho, L. S. T. and Ané, C. 2014. "Intrinsic inference difficulties for trait evolution with Ornstein-Uhlenbeck models". Methods in Ecology and Evolution. 5(11):1133-1146.

Mohammad Khabbazian, Ricardo Kriebel, Karl Rohe, and Cécile Ané (2016). "Fast and accurate detection of evolutionary shifts in Ornstein-Uhlenbeck models". Methods in Ecology and Evolution. doi:10.1111/2041-210X.12534

See Also

estimate_shift_configuration adjust_data

Examples

```
data(lizard.tree, lizard.traits)
lizard <- adjust_data(lizard.tree, lizard.traits[,1])
eModel <- estimate_shift_configuration(lizard$tree, lizard$Y)
configuration_ic(lizard$tree, eModel$Y, eModel$shift.configuration, criterion="pBIC")

### building l1ou object out of the second best score
eModel2 = configuration_ic(eModel$tree, eModel$Y, eModel$profile$configurations[[2]],
                           fit.OU.model=TRUE, l1ou.options=eModel$l1ou.options)
plot(eModel2)
```

`convert_shifts2regions` *Converts shift values to optimum values on the edges.*

Description

Converts a model indicated with shift values to a model with optimum values on the edges.

Usage

```
convert_shifts2regions(tree, shift.configuration, shift.values)
```

Arguments

`tree`	ultrametric tree of class phylo with branch lengths.
`shift.configuration`	vector of edge indices with shifts.
`shift.values`	vector of shift values.

Value

vector of size number of edges with optimum value of the trait on the corresponding edge.

Examples

```
data(lizard.tree, lizard.traits)

sc <- c(55, 98, 118, 74, 14, 77,  32, 164)
sv <- c(2 ,  3,   4,  4,  1,  2, 0.5,   1)

root.value <- -2

optimum.values <- convert_shifts2regions(lizard.tree, sc, sv) + root.value
```

---

estimate_convergent_regimes

Detects convergent regimes under an OU model

Description

Takes a model previously estimated by `estimate_shift_configuration`, including one or more traits and a configuration of evolutionary shifts, and detect which of these regime shifts are convergent.

Usage

```
estimate_convergent_regimes(model, criterion = c("AICc", "pBIC", "BIC"),
  method = c("backward", "rr"), fixed.alpha = FALSE, nCores = 1)
```

Arguments

- `model` — fitted object of class l1ou returned by `estimate_shift_configuration`.
- `criterion` — information criterion for model selection (see Details in `configuration_ic`).
- `method` — search method for finding convergent regimes. “rr” is based on genlasso, a regularized linear regression estimation. Currenly, this method can only accept a single trait. The default “backward” method is a heuristic similar to `surface_backward` in the `surface` package, using backward steps to repeatedly merge similar regimes into convergent regimes.
- `fixed.alpha` — indicates if the alpha parameters should be optimized while phylolm optimize the likelihood function.
- `nCores` — number of processes to be created for parallel computing. If nCores=1 then it will run sequentially. Otherwise, it creates nCores processes by using mclapply function. For parallel computing it, requires parallel package.

See Also

`estimate_shift_configuration`

Examples

```
library(l1ou)
data("lizard.traits", "lizard.tree")
Y <- lizard.traits[, 1:1]
## first fit a model to find individual shifts (no convergence assumed):
fit_ind <- estimate_shift_configuration(lizard.tree, Y, criterion="AICc")
fit_ind
## then detect which of these shifts are convergent:
fit_conv <- estimate_convergent_regimes(fit_ind, criterion="AICc")
fit_conv
plot(fit_conv)
```

---

estimate_shift_configuration

Detects evolutionary shifts under an OU model

Description

This function takes in one or multiple traits, and automatically detects the phylogenetic placement and the magnitude of shifts in the evolution of these traits. The model assumes an Ornstein-Uhlenbeck process whose parameters are estimated (adaptation ‘strength’ α and drift variance σ^2). Instantaneous shifts in the optimal trait value affect the traits over time.

Usage

```
estimate_shift_configuration(tree, Y,
  max.nShifts = floor(length(tree$tip.label)/2), criterion = c("pBIC",
  "pBICess", "mBIC", "BIC", "AICc"), root.model = c("OUfixedRoot",
  "OUrandomRoot"), candid.edges = NA, quietly = TRUE,
  alpha.starting.value = NA, alpha.upper = alpha_upper_bound(tree),
  alpha.lower = NA, lars.alg = c("lasso", "stepwise"), nCores = 1,
  rescale = TRUE, edge.length.threshold = .Machine$double.eps,
  grp.delta = 1/16, grp.seq.ub = 5, l1ou.options = NA)
```

Arguments

`tree` ultrametric tree of class phylo with branch lengths, and edges in postorder.

`Y` trait vector/matrix without missing entries. The row names of the data must be in the same order as the tip labels.

`max.nShifts` upper bound for the number of shifts. The default value is half the number of tips.

`criterion` information criterion for model selection (see Details in `configuration_ic`).

`root.model` ancestral state model at the root.

`candid.edges` a vector of indices of candidate edges where the shifts may occur. If provided, shifts will only be allowed on these edges; otherwise all edges will be considered.

`quietly` logical. If FALSE, a basic summary of the progress and results is printed.

`alpha.starting.value` optional starting value for the optimization of the phylogenetic adaptation rate.

`alpha.upper` optional upper bound for the phylogenetic adaptation rate. The default value is log(2) over the minimum branch length connected to tips.

`alpha.lower` optional lower bound for the phylogenetic adaptation rate.

`lars.alg` model selection algorithm for LARS in univariate case.

`nCores` number of processes to be created for parallel computing. If nCores=1 then it will run sequentially. Otherwise, it creates nCores processes by using mclapply function. For parallel computing it, requires parallel package.

`rescale` logical. If TRUE, the columns of the trait matrix are first rescaled so that all have the same l2-norm. If TRUE, the scores will be based on the rescale one.

`edge.length.threshold` minimum edge length that is considered non-zero. Branches with shorter length are considered as soft polytomies, disallowing shifts on such branches.

`grp.delta` internal (used when the data contain multiple traits). The input lambda sequence for the group lasso, in 'grplasso', will be lambda.max*(0.5^seq(0, grp.seq.ub, grp.delta)).

`grp.seq.ub` (used for multiple traits). The input lambda sequence for grplasso will be lambda.max*(0.5^seq(0, grp.seq.ub, grp.delta)).

`l1ou.options` if provided, all the default values will be ignored.

Details

For information criteria: see `configuration_ic`.

Value

`Y`	input trait vector/matrix.
`tree`	input tree.
`shift.configuration`	estimated shift positions, i.e. vector of indices of edges where the estimated shifts occur.
`shift.values`	estimates of the shift values.
`shift.means`	estimates change of the expectation of the shift values
`nShifts`	estimated number of shifts.
`optima`	optimum values of the trait at tips. If the data are multivariate, this is a matrix where each row corresponds to a tip.
`edge.optima`	optimum values of the trait on the edges. If the data are multivariate, this is a matrix where each row corresponds to an edge.
`alpha`	maximum likelihood estimate(s) of the adaptation rate α, one per trait.
`sigma2`	maximum likelihood estimate(s) of the variance rate σ^2, one per trait.
`mu`	fitted values, i.e. estimated trait means.
`residuals`	residuals. These residuals are phylogenetically correlated.
`score`	information criterion value of the estimated shift configuration.
`profile`	list of shift configurations sorted by their ic scores.
`l1ou.options`	list of options that were used.

References

Mohammad Khabbazian, Ricardo Kriebel, Karl Rohe, and Cécile Ané (2016). "Fast and accurate detection of evolutionary shifts in Ornstein-Uhlenbeck models". Methods in Ecology and Evolution. doi:10.1111/2041-210X.12534

Examples

```
data(lizard.tree, lizard.traits)
# here lizard.traits already has row names:
rownames(lizard.traits)
# also, it is a matrix (not data frame) so columns retain row names:
names(lizard.traits[,1])
# If your trait data "dat" does not have row names but instead has
# species names in a column called "species", then you can
# create row names containing the species names like this:
# rownames(dat) <- dat$species
lizard <- adjust_data(lizard.tree, lizard.traits[,1])
eModel <- estimate_shift_configuration(lizard$tree, lizard$Y)
```

```
eModel

## use parallel computing to accelerate the computation
eModel.par <- estimate_shift_configuration(lizard$tree, lizard$Y, nCores=8)

stopifnot( identical( sort(eModel.par$shift.configuration), sort(eModel$shift.configuration) ) ) ## TRUE

nEdges <- Nedge(lizard.tree) # total number of edges
ew <- rep(1,nEdges)  # to set default edge width of 1
ew[eModel$shift.configuration] <- 3   # to widen edges with a shift
plot(eModel, cex=0.5, label.offset=0.02, edge.width=ew)

# example to constrain the set of candidate branches with a shift
eModel <- estimate_shift_configuration(lizard$tree, lizard$Y, criterion="AICc")
ce <- eModel$shift.configuration # set of candidate edges
eModel <- estimate_shift_configuration(lizard$tree, lizard$Y, candid.edges = ce)
plot(eModel, edge.ann.cex=0.7, cex=0.5, label.offset=0.02)
```

`fit_OU` *Fits an OU model based on a given configuration*

Description

Fits an OU model based on a given configuration

Usage

```
fit_OU(tree, Y, shift.configuration, criterion = c("pBIC", "pBICess", "mBIC",
  "BIC", "AICc"), root.model = c("OUfixedRoot", "OUrandomRoot"),
  cr.regimes = NULL, alpha.starting.value = NA,
  alpha.upper = alpha_upper_bound(tree), alpha.lower = NA,
  l1ou.options = NA)
```

Arguments

`tree` ultrametric tree of class phylo, with branch lengths, and edges in postorder.

`Y` trait vector/matrix without missing entries. The row names of the data must be in the same order as the tip labels.

`shift.configuration` shift positions, i.e. vector of indices of the edges where the shifts occur.

`criterion` an information criterion (see Details).

`root.model` model for the ancestral state at the root.

`alpha.starting.value` optional starting value for the optimization of the phylogenetic adaptation rate.

`alpha.upper` optional upper bound for the phylogenetic adaptation rate. The default value is log(2) over the minimum length of external branches, corresponding to a half life greater or equal to the minimum external branch length.

alpha.lower optional lower bound for the phylogenetic adaptation rate.

l1ou.options if provided, all the default values will be ignored.

Details

AICc gives the usual small-sample size modification of AIC. BIC gives the usual Bayesian information criterion, here penalizing each shift as 2 parameters. mBIC is the modified BIC proposed by Ho and Ané (2014). pBIC is the phylogenetic BIC for shifts proposed by Khabbazian et al. pBICess is a version of pBIC where the determinant term is replaced by a sum of the log of effective sample sizes (ESS), similar to the ESS proposed by Ané (2008).

Value

an object of class l1ou similar to estimate_shift_configuration.

References

Cécile Ané, 2008. "Analysis of comparative data with hierarchical autocorrelation". Annals of Applied Statistics 2(3):1078-1102.

Ho, L. S. T. and Ané, C. 2014. "Intrinsic inference difficulties for trait evolution with Ornstein-Uhlenbeck models". Methods in Ecology and Evolution. 5(11):1133-1146.

Mohammad Khabbazian, Ricardo Kriebel, Karl Rohe, and Cécile Ané (2016). "Fast and accurate detection of evolutionary shifts in Ornstein-Uhlenbeck models". Methods in Ecology and Evolution. doi:10.1111/2041-210X.12534

See Also

estimate_shift_configuration adjust_data

Examples

```
data(lizard.tree, lizard.traits)
lizard <- adjust_data(lizard.tree, lizard.traits[,1])
eModel <- estimate_shift_configuration(lizard$tree, lizard$Y)

### building l1ou object out of the second best score
eModel2 = fit_OU(eModel$tree, eModel$Y, eModel$profile$configurations[[2]],
                          l1ou.options=eModel$l1ou.options)
plot(eModel2)

### hypothesis testing

data("lizard.traits", "lizard.tree")
Y <- lizard.traits[,1:1]
tr  <- lizard.tree

tr <- multi2di(tr)
tr <- reorder(tr, "postorder")
```

```
### visualizing the tree with the edge indeces
plot(tr)
edgelabels()

## place the shift position based on the hypothesis
shift.config <- c(116, 77)

hModel <- fit_OU(tr, Y, shift.config, criterion="AICc")
plot(hModel)
print(hModel)
```

---

get_shift_configuration

Returns the best shift configuration with a given number of shifts among the shift configurations that have been evaluated.

Description

Returns the best shift configuration with a given number of shifts among the shift configurations that have been evaluated.

Usage

```
get_shift_configuration(model, nShifts)
```

Arguments

`model`	object of class l1ou returned by `estimate_shift_configuration`.
`nShifts`	number of shifts.

Value

indices of the edges with shifts

---

l1ou_bootstrap_support

Computes bootstrap support for shift positions

Description

Takes a given shift configuration previously detected from data along with shift magnitudes and OU parameters, to calculate bootstrap support for shift positions. The non-parametric bootstrap procedure calculates phylogenetically-uncorrelated standardized residuals, one at each node. These residuals are sampled with replacement, then mapped back onto the tree to create bootstrap replicates. Each replicate is analyzed with the l1ou method and user-specified options.

Usage

```
l1ou_bootstrap_support(model, nItrs = 100, multicore = FALSE, nCores = 2,
  quietly = TRUE)
```

Arguments

model	an object output by estimate_shift_configuration.
nItrs	number of independent iterations (bootstrap replicates).
multicore	logical. If TRUE, nCores processes are used in parallel.
nCores	desired number of parallel processes.
quietly	logical. If FALSE, a summary of each iteration will be printed out.

Details

The results of sequential and parallel runs are not necessarily equal, because different seeds might be used for different bootstrap replicates. For multiple cores to be used, the `parallel` library needs to be installed. To change options for the analysis of each bootstrap replicate, like the information criterion or the maximum allowed number of shifts, modify model$opt.

Value

vector of size the number of edges in the tree. Each entry is the proportion of bootstrap replicates for which a shift is detected on the corresponding edge.

See Also

estimate_shift_configuration

Examples

```
data(lizard.traits, lizard.tree)
Y <- lizard.traits[,1]
eModel <- estimate_shift_configuration(lizard.tree, Y)
result <- l1ou_bootstrap_support(eModel, nItrs=2)
# using only 2 replicates in vastly insufficient in general,
# but used here to make the illustrative example run faster.
nEdges <- Nedge(lizard.tree)
e.w <- rep(1,nEdges)
e.w[eModel$shift.configuration] <- 3
e.l <- round(result$detection.rate*100, digits=1)
# to avoid annotating edges with support at or below 10%
e.l <- ifelse(e.l>10, paste0(e.l,"%"), NA)
plot(eModel, edge.label=e.l, edge.ann.cex=0.7, edge.label.ann=TRUE, cex=0.5, label.offset=0.02, edge.width=e.w)


Y <- lizard.traits[,1:2]
eModel <- estimate_shift_configuration(lizard.tree, Y)
result <- l1ou_bootstrap_support(eModel, nItrs=2, multicore=TRUE, nCores=4)
```

```
result$detection.rate
```

`lizard.traits` *Morphospace of 100 Anolis lizards on Caribbean islands*

Description

Morphospace of 100 Anolis lizards on Caribbean islands

Usage

```
data(lizard.traits)
```

Format

A matrix with 100 rows and 4 columns

References

Mahler, D. Luke, et al. "Exceptional convergence on the macroevolutionary landscape in island lizard radiations." Science 341.6143 (2013): 292-295.

`lizard.tree` *Phylogenetic tree of 100 Anolis lizards on Caribbean islands*

Description

Phylogenetic tree of 100 Anolis lizards on Caribbean islands

Usage

```
data(lizard.tree)
```

Format

a phylogenetic tree of class phylo with 100 species.

References

Mahler, D. Luke, et al. "Exceptional convergence on the macroevolutionary landscape in island lizard radiations." Science 341.6143 (2013): 292-295.'

normalize_tree *Normalizes branch lengths to a unit tree height*

Description

Normalizes all branch lengths including the root.edge if presents by the same factor, so that the distance from the root to all tips is equal to one.

Usage

```
normalize_tree(tree, check.ultrametric = TRUE)
```

Arguments

`tree`	ultrametric tree of class phylo with branch lengths, and edges in postorder.
`check.ultrametric`	logical. If TRUE, it checks if the input tree is ultrametric.

Value

normalized phylogenetic tree, of class phylo.

plot.l1ou *Visualizes a shift configuration: tree and trait(s)*

Description

plots the tree annotated to show the edges with a shift, and the associated trait data side by side.

Usage

```
## S3 method for class 'l1ou'
plot(model, palette = NA, edge.shift.ann = TRUE,
  edge.shift.adj = c(0.5, -0.025), edge.label = c(), asterisk = TRUE,
  edge.label.ann = FALSE, edge.label.adj = c(0.5, 1), edge.label.pos = NA,
  edge.ann.cex = 1, plot.bar = TRUE, bar.axis = TRUE, ...)
```

Arguments

`model`	object of class l1ou returned by `estimate_shift_configuration`.
`palette`	vector of colors, of size the number of shifts plus one. The last element is the color for the background regime (regime at the root).
`edge.shift.ann`	logical. If TRUE, annotates edges by shift values.
`edge.shift.adj`	adjustment argument to give to edgelabel() for labeling edges by shift values.

edge.label	vector of size number of edges.
asterisk	logical. If TRUE, the shift positions will be annotated by "*". It is useful for gray scale plots.
edge.label.ann	logical. If TRUE, annotates edges by labels in tree$edge.label, if non-empty, or edge.label.
edge.label.adj	adjustment argument to give to edgelabel() for labeling edges.
edge.label.pos	relative position of the edge.label on the edge. 0 for the beginning of the edge and 1 for the end of the edge.
edge.ann.cex	amount by which the annotation text should be magnified relative to the default.
plot.bar	logical. If TRUE, the bars corresponding to the trait values will be plotted.
bar.axis	logical. If TRUE, the axis of of trait(s) range will be plotted.
...	further arguments to be passed on to plot.phylo.

Value

none.

Examples

```
data(lizard.traits, lizard.tree)
Y <- lizard.traits[,1]
eModel <- estimate_shift_configuration(lizard.tree, Y)
nEdges <- Nedge(lizard.tree)
ew <- rep(1,nEdges)
ew[eModel$shift.configuration] <- 3
plot(eModel, cex=0.5, label.offset=0.02, edge.width=ew)
```

---

profile.l1ou — *Prints out a summary of the shift configurations investigated by* estimate_shift_configuration

---

Description

prints the list of the shift configurations sorted by number of shifts and corresponding ic scores.

Usage

```
## S3 method for class 'l1ou'
profile(model, ...)
```

Arguments

model	object of class l1ou returned by estimate_shift_configuration.
...	further arguments.

Value

shift.configurations
: list of shift configurations sorted by number of shifts.

scores
: list of scores corresponding to shift.configurations.

nShifts
: number of shifts corresponding to the shift configurations.

Examples

```
data(lizard.traits, lizard.tree)
Y <- lizard.traits[,1]
eModel <- estimate_shift_configuration(lizard.tree, Y)
model.profile  <- profile(eModel)
plot(model.profile$nShifts, model.profile$scores)
```

---

sqrt_OU_covariance *(inverse) square root of the phylogenetic covariance*

Description

Computes an inverse square root and square root of the phylogenetic covariance matrix, under the Brownian motion (BM) or the Ornstein-Uhlenbeck (OU) model. The algorithm traverses the tree only once, hence the algorithm is very fast and can be applied to very big trees.

Usage

```
sqrt_OU_covariance(tree, alpha = 0, root.model = c("OUfixedRoot",
  "OUrandomRoot"), check.order = TRUE, check.ultrametric = TRUE)
```

Arguments

tree
: tree of class phylo with branch lengths. If alpha>0, i.e. under the OU model, the tree has to be ultrametric.

alpha
: adaptation rate for the OU model. The default is 0, which corresponds to the BM mode with a fixed ancestral state at the root.

root.model
: ancestral state model at the root.

check.order
: logical. If TRUE, the order will be checked to be in postorder traversal.

check.ultrametric
: logical. If TRUE, the tree will be checked to ultrametric.

Value

sqrtInvSigma
: inverse square root of the phylogenetic covariance matrix.

sqrtSigma
: square root of the phylogenetic covariance matrix.

References

Mohammad Khabbazian, Ricardo Kriebel, Karl Rohe, and Cécile Ané (2016). "Fast and accurate detection of evolutionary shifts in Ornstein-Uhlenbeck models". Methods in Ecology and Evolution. doi:10.1111/2041-210X.12534

Eric A. Stone. 2011. "Why the phylogenetic regression appears robust to tree misspecification". Systematic Biology, 60(3):245-260.

Examples

```
data(lizard.tree)
res <- sqrt_OU_covariance(lizard.tree) # alpha not provided: so BM model.
Sigma <- vcv(lizard.tree)
dimnames(Sigma) <- NULL
all.equal(res$sqrtSigma %*% t(res$sqrtSigma), Sigma) # TRUE
all.equal(res$sqrtInvSigma %*% t(res$sqrtInvSigma), solve(Sigma)) # TRUE


##Here's the example from "Eric A. Stone. 2011." (See references)

tr <-  read.tree(text="((((Homo:.21,Pongo:.21):.28,Macaca:.49):.13,Ateles:.62):.38,Galago:1);")
RE <- sqrt_OU_covariance(tr)
B <- round( RE$sqrtSigma, digits=3)
D <- round( RE$sqrtInvSigma, digits=3)
print(B)
print(D)


##Here is the examples on how to get the contrasts using sqrt_OU_covariance
data(lizard.tree, lizard.traits)
lizard <- adjust_data(lizard.tree, lizard.traits[,1])
eModel <- estimate_shift_configuration(lizard$tree, lizard$Y)
theta <- eModel$intercept + l1ou:::convert_shifts2regions(eModel$tree,
                             eModel$shift.configuration, eModel$shift.values)
REf <- sqrt_OU_covariance(eModel$tree, alpha=eModel$alpha,
                                         root.model = "OUfixedRoot",
                                         check.order=FALSE, check.ultrametric=FALSE)
 covInverseSqrtf  <- t(REf$sqrtInvSigma)
 covSqrtf   <- REf$sqrtSigma
# `covInverseSqrtf` represents the transpose of square root of  the inverse matrix of covariance for FixedRoot mod
# `covSqrtf` represents the square root of the covariance matrix for FixedRoot model.
 Y  <- rTraitCont(eModel$tree, "OU", theta=theta,
                                     alpha=eModel$alpha,
                                     sigma=eModel$sigma, root.value=eModel$intercept)
 contrast    <-  covInverseSqrtf%*%(Y - eModel$mu)
```

summary.l1ou *Prints out a summary of the model*

Description

prints out a summary of the model

Usage

```
## S3 method for class 'l1ou'
summary(model, nTop.scores = 5, ...)
```

Arguments

`model`	object of class l1ou returned by `estimate_shift_configuration`.
`nTop.scores`	number of top scores and shift configuration to print out.
`...`	further arguments.

Value

none.

Examples

```
data(lizard.traits, lizard.tree)
Y <- lizard.traits[,1]
eModel <- estimate_shift_configuration(lizard.tree, Y)
summary(eModel)
```

Index