Package ‘ouch’
July 19, 2017

Type Package
Title Ornstein-Uhlenbeck Models for Phylogenetic Comparative Hypotheses
Version 2.11-1
Date 2017-07-18
Author Aaron A. King <kingaa@umich.edu> and Marguerite A. Butler <mbutler@hawaii.edu>
Maintainer Aaron A. King <kingaa@umich.edu>
Description Fit and compare Ornstein-Uhlenbeck models for evolution along a phylogenetic tree.
Depends R(>= 3.0.0), methods, stats, graphics, subplex
Suggests ape, geiger
URL http://kingaa.github.io/ouch/
License GPL (>= 2)
LazyLoad true
NeedsCompilation yes
Repository CRAN
Date/Publication 2017-07-18 22:42:21 UTC

R topics documented:

  ouch-package .................................................. 2
  anolis.ssd ..................................................... 3
  ape2ouch ....................................................... 4
  bimac .......................................................... 5
  brown ........................................................... 7
  browntree ....................................................... 8
  hansen .......................................................... 9
  hansentree ..................................................... 12
  hansentree-methods ........................................... 12
  ouchtree ....................................................... 14
  paint .......................................................... 15
Description

The `ouch` package provides facilities for Ornstein-Uhlenbeck based methods of phylogenetic comparative analysis.

Classes

The basic class, `ouchtree`, is provided to encode a phylogenetic tree. Plot and print methods are provided.

The class `brownTree` derives from class `ouchtree` and encodes the results of fitting a Brownian Motion model to data.

The class `hansenTree` derives from class `ouchtree` and encodes the results of fitting a Hansen model to data.

Detailed Documentation

- **ouchtree** `ouchtree`
- **Brownian motion methods** `brown, brownTree-class`
- **Ornstein-Uhlenbeck methods** `hansen, hansenTree-class`
- **simulate methods** `simulate-brownTree, simulate-hansenTree`
- **plot methods** `plot-brownTree, plot-hansenTree`
- `ape2ouch` Convert a tree in `ape` format to `ouch` format. See `ape2ouch`

Author(s)

Aaron A. King (kingaa at umich dot edu)

References

The dataset consists of sexual size-dimorphism data for 38 species of anoles from Cuba, Hispaniola, Jamaica, and Puerto Rico (Butler, Schoener, and Losos 2000). Each of these species belongs to one of six microhabitat types, or “ecomorphs” (sensu Williams, 1972): trunk-ground, grass-bush, trunk, trunk-crown, twig, and crown-giant. The data were used to demonstrate an evolutionary association between habitat type and degree of sexual size dimorphism.

Usage

data(anolis.ssd)

Format

A data frame with 38 observations on the following 6 variables.

- **node**: Labels for the nodes.
- **species**: Names of extant species.
- **log.SSD**: Log sexual size dimorphism of extant species.
- **ancestor**: Ancestor node.
- **time**: Time of node.
- **OU.1**: a factor with levels ns
- **OU.7**: a factor with levels corresponding to ecomorph (tg tc gb cg tw tr anc)

Details

Size dimorphism was calculated as the log-ratio of male snout-to-vent length to female snout-to-vent length (males are larger).

In this example, we tested three models of evolution: Brownian motion, Ornstein-Uhlenbeck with one global optimum, and Ornstein-Uhlenbeck with 7 optima (one for each ecomorph type plus an additional one for an “unknown” type).

For the 7-optima model, we assigned each terminal branch to an optimum according to the ecomorph type of the extant species. Because we had no information to help guide hypotheses about internal branches, we assigned internal branches to the “unknown” selective regime. The phylogeny of these species is consistent with and adaptive radiation, with a burst of speciation events early in the evolutionary history of this clade (see phylogeny in Butler & King (2004) or example below.

Author(s)

Marguerite A. Butler <mbutler at hawaii dot edu> and Aaron A. King <kingaa at umich dot edu>
Source


References


Examples

data(anolis.ssd)
tree <- with(anolis.ssd, ouchtree(node, ancestor, time/max(time), species))
plot(tree, node.names=TRUE)
print(h1 <- brown(anolis.ssd['log.SSD'], tree))
plot(h1)
print(h2 <- hansen(anolis.ssd['log.SSD'], tree, anolis.ssd['0U.1'], sqrt.alpha=1, sigma=1))
plot(h2)
print(h3 <- hansen(anolis.ssd['log.SSD'], tree, anolis.ssd['0U.7'], sqrt.alpha=1, sigma=1))
plot(h3)

ape2ouch

Convert an "ape" tree to an "ouch" tree.

Description

ape2ouch translates ape's phylo representation of a phylogenetic tree into ouch's ouchtree representation. The user can change the branch lengths while preserving the topology.

Usage

ape2ouch(tree, scale = TRUE, branch.lengths = tree$edge.length)

Arguments

tree a tree of class phylo created in package ape.

scale if scale=TRUE, the tree’s depth will be scaled to 1. If scale is a number, then the branch lengths will be scaled by this number.

branch.lengths optional vector of branch lengths.

Author(s)

Aaron A. King <kingaa@umich.edu>
Description

This is the Anolis bimaculatus dataset used in Butler & King (2004). It is used to test a hypothesis of character displacement using an interspecific dataset of body sizes and current data on sympatry/allopatry. The data frame consists of the following columns: species which are species names, size which is the phenotypic data, and the variables ancestor and time which specify the topology of the phylogeny and the location of the nodes in time, respectively. The columns OU.1, OU.3, OU.4, and OU.LP specify four hypothetical arrangements of selective regimes. Explanations of the data are given below.

Usage

data(bimac)

Format

A data frame with 45 observations on the following 8 variables.

- **node**: Labels for the nodes.
- **species**: Species names for extant species.
- **size**: Body size (head length in mm) of extant species.
- **ancestor**: Ancestral node.
- **time**: Time of node.
- **OU.1**: a factor with levels ns
- **OU.3**: a factor with levels small medium large
- **OU.4**: a factor with levels small medium large anc
- **OU.LP**: a factor with levels small medium large

Details

**Body size.** We use the phenotypic data and phylogeny of Losos (1990), which employed the head lengths (of males) as a proxy for body size. In this group of lizards, head length correlates very strongly with snout-to-vent length and the cube root of mass, which are standard measures of body size. The data are head lengths in mm, note that we use the log of this value in analyses.

**Tree topology** The tree topology is encoded via two vectors: ancestor and time. Each node of the phylogenetic tree has a corresponding row in the data frame, numbered from 1 to 45. The columns ancestor and time specify the phylogeny. The ancestor variable specifies the topology: it is a list indicating the ancestor of each node. The root node has ancestor 0. The variable time specifies the temporal location of each node, with the root node being at time 0.
Specifications of selective regimes. (Columns OU.1, OU.3, OU.4, OU.LP). These columns are factors, the levels of which correspond to the “paintings” of the respective adaptive regime hypotheses onto the phylogeny. Each selective regime is named (small, medium, large, etc.). Put the corresponding name on each branch segment to indicate which selective regime it belongs to. Each column corresponds to a different painting of the selective regimes, and thus to a different hypothesis. In this example, there are 3 alternative models (see Butler & King 2004): OU.4 is 4-regime model, OU.3 is 3-regime model (all ancestors are medium), OU.LP is linear parsimony model.

Author(s)
Marguerite A. Butler <mbutler at hawaii dot edu> and Aaron A. King <kingaa at umich dot edu>

Source

References


Examples
```r
data(bimac)
tree <- with(bimac, ouchtree(node, ancestor, time/max(time), species))
plot(tree, node.names=TRUE)
print(h1 <- brown(log(bimac['size']), tree))
plot(h1)
print(h2 <- hansen(log(bimac['size']), tree, bimac[ 'OU.1'], sqrt.alpha=1, sigma=1))
plot(h2)
print(h3 <- hansen(log(bimac['size']), tree, bimac[ 'OU.3'], sqrt.alpha=1, sigma=1))
plot(h3)
print(h4 <- hansen(log(bimac['size']), tree, bimac[ 'OU.4'], sqrt.alpha=1, sigma=1))
plot(h4)
h5 <- hansen(log(bimac['size']), tree, bimac[ 'OU.LP'], sqrt.alpha=1, sigma=1, reltol=1e-5)
print(h5 <- update(h5, method='subplex', reltol=1e-11, parscale=c(0.1,0.1), hessian=TRUE))
simdat <- simulate(h5, nsim=10)
hsim <- update(h5, data=simdat[[1]])
print(summary(hsim))
bsim <- update(h1, data=simdat[[1]])
print(summary(bsim))
```
**brown**

*Fit a Brownian-motion model of evolution along a phylogenetic tree*

---

**Description**

*brown* fits a Brownian motion model to the given data.

**Usage**

```r
brown(data, tree)
```

**Arguments**

- `data` Phenotypic data for extant species, i.e., at the terminal ends of the phylogenetic tree. This can either be a numeric vector or a list. If it is a numeric vector, there must be one entry for every node. If it is a list, it must consist entirely of numeric vectors, each of which has one entry per node. A data-frame is coerced to a list.
- `tree` A phylogenetic tree, specified as an `ouchtree` object.

**Value**

*brown* returns an object of class `browntree`. See `browntree-class` for information on the methods of this class.

**Author(s)**

Aaron A. King <kingaa at umich dot edu>

**References**


**See Also**

`ouchtree`, `browntree`
browntree  

Fitted phylogenetic Brownian motion model

Description

A fitted phylogenetic Brownian-motion model object.

Details

The function `brown` creates a `browntree` object by fitting a Brownian-motion model to data.

Methods

- `plot()` plots the tree.
- `print()` prints the tree as a table, along with the coefficients of the fitted model and diagnostic information.
- `show()` displays the fitted `browntree` object.
- `summary()` displays information on the call, the fitted coefficients, and model selection statistics.
- `coef(object,...)` extracts the coefficients of the fitted model. This is a list with three elements:
  - `sigma`: the coefficients of the sigma matrix.
  - `theta`: a list of the estimated optima, one per character.
  - `sigma.sq.matrix`: the sigma-squared matrix itself.
- `loglik(object,...)` extracts the log likelihood of the fitted model.
- `update(object,...)` refits the model. `object` is the `browntree` object. Additional arguments (in ...) replace the corresponding arguments in the original call.
- `bootstrap(object, nboot = 200, seed = NULL, ...)` performs a parametric bootstrap for estimation of confidence intervals. `object` is the `browntree` object. `nboot` is the number of bootstraps. `seed` allows one to fix the random seed (see `simulate` below). Additional arguments (in ...) are passed to `update`.
- `simulate(object, nsim = 1, seed = NULL, ...)` generates random deviates from the fitted model. `object` is the `browntree` object, `nsim` is the desired number of replicates, and `seed` is (optionally) the random seed to use. `simulate` returns a list of data-frames, each comparable to the original data.

Author(s)

Aaron A. King kingaa at umich dot edu

See Also

`ouchtree`, `brown`
**hansen**  

**Hansen model of evolution along a phylogenetic tree**

**Description**

Fits the Ornstein-Uhlenbeck-based Hansen model to data. The fitting is done using optim or subplex.

**Usage**

```r
hansen(data, tree, regimes, sqrt.alpha, sigma,  
       fit = TRUE,  
       method = c("Nelder-Mead","subplex","BFGS","L-BFGS-B"),  
       hessian = FALSE, ...)
```

**Arguments**

- **data**  
  Phenotypic data for extant species, i.e., species at the terminal twigs of the phylogenetic tree. This can either be a single named numeric vector, a list of nchar named vectors, or a data-frame containing nchar data variables. There must be an entry per variable for every node in the tree; use NA to represent missing data. If the data are supplied as one or more named vectors, the names attributes are taken to correspond to the node names specified when the ouchtree was constructed (see ouchtree). If the data are supplied as a data-frame, the rownames serve that purpose.

- **tree**  
  A phylogenetic tree, specified as an ouchtree object.

- **regimes**  
  A vector of codes, one for each node in the tree, specifying the selective regimes hypothesized to have been operative. Corresponding to each node, enter the code of the regime hypothesized for the branch segment terminating in that node. For the root node, because it has no branch segment terminating on it, the regime specification is irrelevant. If there are nchar quantitative characters, then one can specify a single set of regimes for all characters or a list of nchar regime specifications, one for each character.

- **sqrt.alpha, sigma**  
  These are used to initialize the optimization algorithm. The selection strength matrix $\alpha$ and the random drift variance-covariance matrix $\sigma^2$ are parameterized by their matrix square roots. Specifically, these initial guesses are each packed into lower-triangular matrices (column by column). The product of this matrix with its transpose is the $\alpha$ or $\sigma^2$ matrix. See Details, below.

- **fit**  
  If fit=TRUE, then the likelihood will be maximized. If fit=FALSE, the likelihood will be evaluated at the specified values of sqrt.alpha and sigma; the optima theta will be returned as well.

- **method**  
  The method to be used by the optimization algorithm, optim. See subplex and optim for information on the available options.

- **hessian**  
  If hessian=TRUE, then the Hessian matrix will be computed by optim.
Additional arguments will be passed as control options to optim or subplex. See optim and subplex for information on the available options.

Details

The Hansen model for the evolution of a multivariate trait $X$ along a lineage can be written as a stochastic differential equation (Ito diffusion)

$$dX = \alpha(\theta(t) - X(t))dt + \sigma dB(t),$$

where $t$ is time along the lineage, $\theta(t)$ is the optimum trait value, $B(t)$ is a standard Wiener process (Brownian motion), and $\alpha$ and $\sigma$ are matrices quantifying, respectively, the strength of selection and random drift. Without loss of generality, one can assume $\sigma$ is lower-triangular. This is because only the infinitesimal variance-covariance matrix $\sigma^2 = \sigma\sigma^T$ is identifiable, and for any admissible variance-covariance matrix, we can choose $\sigma$ to be lower-triangular. Moreover, if we view the basic model as describing evolution on a fitness landscape, then $\alpha$ will be symmetric and if we further restrict ourselves to the case of stabilizing selection, $\alpha$ will be positive definite as well. We make these assumptions and therefore can assume that the matrix $\alpha$ has a lower-triangular square root.

The hansen code uses unconstrained numerical optimization to maximize the likelihood. To do this, it parameterizes the $\alpha$ and $\sigma^2$ matrices in a special way: each matrix is parameterized by $n\text{char} \times (n\text{char}+1)/2$ parameters, where $n\text{char}$ is the number of quantitative characters. Specifically, the parameters initialized by the sqrt.alpha argument of hansen are used to fill the nonzero entries of a lower-triangular matrix (in column-major order), which is then multiplied by its transpose to give the selection-strength matrix. The parameters specified in sigma fill the nonzero entries in the lower triangular $\sigma$ matrix. When hansen is executed, the numerical optimizer maximizes the likelihood over these parameters. The print, show, and summary methods for the resulting hansentree object display (among other things) the estimated $\alpha$ and $\sigma^2$ matrices. The coef method extracts a named list containing not only these matrices (given as the alpha.matrix and sigma.sq.matrix elements) but also the MLEs returned by the optimizer (as sqrt.alpha and sigma, respectively). The latter elements should not be interpreted, but can be used to restart the algorithm, etc.

Value

hansen returns an object of class hansentree. For details on the methods of that class, see hansentree.

Author(s)

Aaron A. King <kingaa at umich dot edu>

References


See Also

ouchtree, hansentree, optim, bimac, anolis.ssd
Examples

```r
## Not run:
if (library(geiger)) {

### an example data set (Darwin's finches)
data(geospiza)
str(geospiza)
sapply(geospiza, class)

### check the correspondence between data and tree tips:
print(nc <- with(geospiza, name.check(geospiza.tree, geospiza.data)))
### looks like one of the terminal twigs has no data associated
### drop that tip:
tree <- with(geospiza, drop.tip(geospiza.tree, nc$tree_not_data))
dat <- as.data.frame(geospiza$dat)

### make an ouchtree out of the phy-format tree
ot <- ape2ouch(tree)

### merge data with tree info
otd <- as(ot, "data.frame")
### in these data, it so happens that the rownames correspond to node names
### we will exploit this correspondence in the 'merge' operation:
dat$labels <- rownames(dat)
otd <- merge(otd, dat, by="labels", all=TRUE)
rownames(otd) <- otd$nodes
print(otd)
### this data-frame now contains the data as well as the tree geometry

### now remake the ouch tree
ot <- with(otd, ouchtree(nodes=nodes, ancestors=ancestors, times=times, labels=labels))

b1 <- brown(tree=ot, data=otd[c("tarsusL","beakD")])
summary(b1)

### evaluate an OU model with a single, global selective regime
otd$regimes <- as.factor("global")
h1 <- hansen(  
  tree=ot,  
  data=otd[c("tarsusL","beakD")],  
  regimes=otd["regimes"],  
  sqrt.alpha=c(1,0,1),  
  sigma=c(1,0,1),  
  maxit=10000
)

summary(h1)
}

## End(Not run)
```
hansentree

hansen tree object

Description

A fitted phylogenetic Hansen-model object.

Details

The function hansen creates a hansentree object by fitting a Hansen model to data.

Methods

See hansentree-methods.

Author(s)

Aaron A. King kingaa at umich dot edu

See Also

ouchtree, hansen, hansentree-methods

hansentree-methods

Methods of the "hansentree" class

Description

Methods of the "hansentree" class.

Usage

## S4 method for signature 'hansentree'
logLik(object)
## S4 method for signature 'hansentree'
coef(object, ...)
## S4 method for signature 'hansentree'
summary(object, ...)
## S4 method for signature 'hansentree'
show(object)
## S4 method for signature 'hansentree'
print(x, ...)
## S4 method for signature 'hansentree'
plot(x, y, regimes, ...)
## S4 method for signature 'hansentree'
simulate(object, nsim = 1, seed = NULL, ...)
## sT method for signature `hansentree`

```r
update(object, data, regimes, sqrt.alpha, sigma, ...)
```

## sT method for signature `hansentree`

```r
bootstrap(object, nboot = 200, seed = NULL, ...)
```

## sT method for signature `hansentree`

```r
as(object, class)
```

## sT method for signature `hansentree, data.frame`

```r
coerce(from, to = "data.frame", strict = TRUE)
```

### Arguments

- **object**: The hansentree object.
- **x**: the hansentree object.
- **class**: character; name of the class to which object should be coerced.
- **from, to**: the classes between which coercion should be performed.
- **nsim**: The number of simulations to perform.
- **nboot**: The number of bootstraps to perform.
- **seed**: The random seed to use in simulations.
- **regimes**, **sqrt.alpha**, **sigma**: See `hansen`.  
- **data**: see `hansen`.
- **y, strict**: Ignored.
- **...**: Further arguments (either ignored or passed to underlying functions). In the case of `update`, these replace the corresponding arguments in the original call.

### Methods

- **plot()**: plots the tree, with branches colored according to the selective regimes. See `plot-ouchtree` for more details.
- **print()**: prints the tree as a table, along with the coefficients of the fitted model and diagnostic information.
- **show()**: displays the fitted hansentree object.
- **summary()**: displays information on the call, the fitted coefficients, and model selection statistics.
- **coerce**: A hansentree object can be coerced to a data-frame via `as(object,"data.frame")`.
- **coef()**: extracts the coefficients of the fitted model. This is a list with five elements:
  - **sqrt.alpha**: the coefficients that parameterize the alpha matrix.
  - **sigma**: the coefficients that parameterize the sigma matrix.
  - **theta**: a list of the estimated optima, one per character. Each element of the list is a vector containing one optimal value per regime.
  - **alpha.matrix**: the alpha matrix itself.
  - **sigma.sq.matrix**: the sigma-squared matrix itself.
- **logLik()**: extracts the log likelihood of the fitted model.
- **update()**: refines the model fit.
bootstrap() performs a parametric bootstrap for confidence intervals.
simulate() generates random deviates from the fitted model. object is the hansentree object, nsim is the desired number of replicates, and seed is (optionally) the random seed to use. simulate returns a list of data-frames, each comparable to the original data.

**Author(s)**

Aaron A. King kingaa at umich dot edu

**See Also**

ouchtree, hansen

---

**ouchtree**

*Phylogenetic tree object in 'ouch' format.*

**Description**

An object containing a phylogenetic tree in a form suitable for using ouch methods.

**Usage**

ouchtree(nodes, ancestors, times, labels = as.character(nodes))

**Arguments**

- **nodes**: A character vector giving the name of each node. These are used internally and must be unique.
- **ancestors**: Specification of the topology of the phylogenetic tree. This is in the form of a character vector specifying the name (as given in the nodes argument) of the immediate ancestor of each node. In particular, the i-th name is that of the ancestor of the i-th node. The root node is distinguished by having no ancestor (i.e., NA).
- **times**: A vector of nonnegative numbers, one per node in the tree, specifying the time at which each node is located. Time should be increasing from the root node to the terminal twigs.
- **labels**: Optional vector of node labels. These will be used in plots to label nodes. It is not necessary that these be unique.

**Details**

ouchtree creates an ouchtree object. This contains the topology, branch times, and epochs. It also holds (optionally) names of taxa for display purposes.
Methods

plot(tree, regimes=NULL, node.names=FALSE, legend=TRUE, ...) displays the phylogenetic tree graphically.

print() displays the tree in table form.
show() displays the tree.
coerce An ouchtree object can be coerced to a data-frame via as(object, "data.frame").

Author(s)

Aaron A. King kingaa at umich dot edu

See Also

ouchtree, ape2ouch, brown, hansen

Examples

data(bimac)
tree <- with(bimac, ouchtree(nodes=node, ancestors=ancestor, times=time, labels=species))
plot(tree)
plot(tree, node.names=TRUE)
print(tree)

Painting selective regimes on a phylogenetic tree.

Description

Function to paint selective regimes on a phylogenetic tree.

Usage

paint(tree, subtree, branch, which = 1)

Arguments

tree An object of class ouchtree.
subtree An optional named vector specifying the root nodes of subtrees. Each branch that descends from this node will be painted with the specified regime.
branch An optional named vector specifying the end nodes of branches. The unique branch that terminates at the named node will be painted with the specified regime.
which integer; if tree is a hansentree, start not with a blank canvas but with the regime specifications tree contains for the character indicated by which.
Details

The names of subtree and branch must be the names of nodes of tree. The painting proceeds in a particular order: one can overpaint a branch. The subtrees indicated by the elements of subtree are painted first, in order. Then the branches indicated by branch are painted. If tree is a simple ouchtree object, then paint begins with a blank canvas, i.e., a tree painted with the single regime "nonspec". If tree inherits class hansentree, then paint begins with the regimes specified in the regimes slot of tree. Note that, if tree is a multivariate hansentree, then there are multiple regime specifications contained in tree. In this case, the argument which lets you pick which one you wish to begin with; by default, the first is used.

Value

A vector of class 'factor' with names corresponding to the nodes in tree, specifying selective regimes.

Author(s)

Aaron A. King kingaa at umich dot edu

See Also

ouchtree, hansen

Examples

data(bimac)
x <- with(bimac, ouchtree(nodes=node, times=time/max(time), ancestors=ancestor, labels=species))
r <- paint(x, subtree=c("1"="medium", "9"="large", "2"="small"), branch=c("38"="large", "2"="medium"))
plot(x, regimes=r, node.names=TRUE)
# compare to bimac
h5 <- hansen(data=log(bimac["size"]), tree=x, regimes=bimac["OU.LP"], sqrt.alpha=1, sigma=1, reltol=1e-5)
r <- paint(h5, branch=c("18"="large"), subtree=c("9"="small"))
plot(x, regimes=r, node.names=TRUE)
Index

+Topic models
   anolis.ssd, 3
   apeouch, 4
   bimac, 5
   brown, 7
   browntree, 8
   hansen, 9
   hansentree, 12
   hansentree-class (hansentree), 12
   hansentree-methods, 12
   ouch-package, 2
   ouchtree, 14
   paint, 15
   coerce, ouchtree, data.frame-method
      (ouchtree), 14
   hansen, 2, 9, 12–14
   hansentree, 10, 12
   hansentree-class (hansentree), 12
   hansentree-methods, 12, 12
   logLik, browntree-method (browntree), 8
   logLik, hansentree-method
      (hansentree-methods), 12
   logLik-browntree (browntree), 8
   logLik-hansentree (hansentree-methods), 12
   optim, 9, 10
   ouch-package, 2
   ouchtree, 2, 7–10, 12, 14, 14
   ouchtree-class (ouchtree), 14
   paint, 15
   plot, browntree-method (browntree), 8
   plot, hansentree-method
      (hansentree-methods), 12
   plot, ouchtree-method (ouchtree), 14
   plot-browntree (browntree), 8
   plot-hansentree (hansentree-methods), 12
   plot-ouchtree, 13
   plot-ouchtree (ouchtree), 14
   print, browntree-method (browntree), 8
   print, hansentree-method
      (hansentree-methods), 12
   print, ouchtree-method (ouchtree), 14
   print-browntree (browntree), 8
   print-hansentree (hansentree-methods), 12
   print-ouchtree (ouchtree), 14
   show, browntree-method (browntree), 8
   show, hansentree-method
      (hansentree-methods), 12
show, ouchtree-method (ouchtree), 14
show-browntree (browntree), 8
show-hansentree (hansentree-methods), 12
show-ouchtree (ouchtree), 14
simulate, browntree-method (browntree), 8
simulate, hansentree-method
    (hansentree-methods), 12
simulate-browntree (browntree), 8
simulate-hansentree
    (hansentree-methods), 12
subplex, 9, 10
summary, browntree-method (browntree), 8
summary, hansentree-method
    (hansentree-methods), 12
summary-browntree (browntree), 8
summary-hansentree
    (hansentree-methods), 12
update, browntree-method (browntree), 8
update, hansentree-method
    (hansentree-methods), 12
update-browntree (browntree), 8
update-hansentree (hansentree-methods), 12