

Package: semtree (via r-universe)

October 11, 2026

Type Package

Title Recursive Partitioning for Structural Equation Models

Author Andreas M. Brandmaier [aut, cre], John J. Prindle [aut], Manuel Arnold [aut], Caspar J. Van Lissa [aut], Moritz John [ctb]

Maintainer Andreas M. Brandmaier <andy@brandmaier.de>

Depends R (>= 2.10), OpenMx (>= 2.6.9),

Imports rpart, rpart.plot (>= 3.0.6), lavaan, cluster, ggplot2 (>= 3.3.0), tidyr, dplyr, methods, strucchange, sandwich, zoo, crayon, clisymbols, future.apply, data.table, expm, gridBase

Suggests knitr, rmarkdown, viridis, MASS, psych, psychTools, testthat, future, ctsem, ctsemOMX

Description SEM Trees and SEM Forests -- an extension of model-based decision trees and forests to Structural Equation Models (SEM). SEM trees hierarchically split empirical data into homogeneous groups each sharing similar data patterns with respect to a SEM by recursively selecting optimal predictors of these differences. SEM forests are an extension of SEM trees. They are ensembles of SEM trees each built on a random sample of the original data. By aggregating over a forest, we obtain measures of variable importance that are more robust than measures from single trees. A description of the method was published by Brandmaier, von Oertzen, McArdle, & Lindenberger (2013) <doi:10.1037/a0030001> and Arnold, Voelkle, & Brandmaier (2020) <doi:10.3389/fpsyg.2020.564403>.

License GPL-3

Encoding UTF-8

LazyLoad yes

Version 0.10.0

Date 2026-10-07

VignetteBuilder knitr

BugReports <https://github.com/brandmaier/semtree/issues>

URL <https://github.com/brandmaier/semtree>

Language en-US

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake make libicu-dev

Repository <https://lifebrain.r-universe.dev>

Date/Publication 2026-10-11 19:24:41 UTC

RemoteUrl <https://github.com/brandmaier/semtree>

RemoteRef HEAD

RemoteSha 0560d8ab311b6a81c8724045fb104a168894340d

Contents

semtree-package	3
aggregateVarimp	3
biodiversity	4
boruta	5
coef.semtree	6
computePval_maxLR	6
countPredictors	7
diversityMatrix	8
evaluate	8
evaluateDataLikelihood	9
evaluateTree	10
findOtherSplits	11
fitSubmodels	12
getDepth	12
getHeight	13
getLeafs	14
getNodeById	14
getNumNodes	15
getParDiffForest	15
getParDiffTree	16
getTerminalNodes	17
hellinger	17
isLeaf	18
kl	18
klsym	19
lgcm	19
merge.semforest	20
modelEstimates	20
outliers	21
parameters	22
partialDependence	23
partialDependence_data	24
partialDependence_growth	25

plot.semtree	26
plotParDiffForest	27
plotParDiffTree	28
plotTreeStructure	28
predict.semforest	29
proximity	30
prune	31
se	32
semforest	33
semforest.control	34
semtree	35
semtree.constraints	37
semtree_control	38
strip	41
subforest	42
subtree	43
surrogate	44
thinOut	44
toTable	45
varimp	45
varimpConvergencePlot	47
Index	49

semtree-package	<i>SEM Tree Package</i>
-----------------	-------------------------

Description

SEM Tree Package

Usage

.SCALE_METRIC

aggregateVarimp	<i>Aggregate Variable Importance Estimates</i>
-----------------	--

Description

This function aggregates variable importance estimates over trees. It is a helper function used when print() is called on a variable importance estimate from a SEM forest.

Usage

```
aggregateVarimp(
  vimp,
  aggregate = c("mean", "median"),
  scale = c("absolute", "relative.baseline"),
  omit.na = TRUE,
  scale.by = 1
)
```

Arguments

<code>vimp</code>	Variable importance estimate from a SEM forest.
<code>aggregate</code>	Character. Either 'mean' or 'median' as function to aggregate estimates over a forest
<code>scale</code>	Character. Either 'absolute' or 'relative.baseline'.
<code>omit.na</code>	Boolean. By default TRUE, which ignores NA estimates when aggregating. Otherwise they are interpreted as zero.
<code>scale.by</code>	Integer. By default 1. Scaling parameter.

biodiversity

Quantify bio diversity of a SEM Forest

Description

A function to calculate biodiversity of a [semforest](#) object.

Usage

```
biodiversity(x, aggregate.fun = median)
```

Arguments

<code>x</code>	A semforest object
<code>aggregate.fun</code>	Takes a function to apply to the vector of pairwise diversities. By default, this is the median.

Author(s)

Andreas M. Brandmaier

boruta

Run the Boruta algorithm on a sem tree

Description

Grows a series of SEM Forests following the boruta algorithm to determine feature importance as moderators of the underlying model.

Usage

```
boruta(
  model,
  data,
  control = NULL,
  predictors = NULL,
  maxRuns = 30,
  pAdjMethod = "none",
  alpha = 0.05,
  verbose = FALSE,
  quant = 1,
  ...
)
```

Arguments

model	A template SEM. Same as in <code>semtree</code> .
data	A dataframe to boruta on. Same as in <code>semtree</code> .
control	A semforest control object to set forest parameters.
predictors	An optional list of covariates. See <code>semtree</code> code example.
maxRuns	Maximum number of boruta search cycles
pAdjMethod	A value from p.adjust.methods defining a multiple testing correction method
alpha	p-value cutoff for decision making. Default .05
verbose	Verbosity level for boruta processing similar to the same argument in semtree.control and semforest.control
quant	Quantile for selection. Default 1.
...	Optional parameters to undefined subfunctions

Value

A vim object with several elements that need work. Of particular note, ‘\$importance’ carries mean importance; ‘\$decision’ denotes Accepted/Rejected/Tentative; ‘\$impHistory’ has the entire varimp history; and ‘\$details’ has exit values for each parameter.

Author(s)

Priyanka Paul, Timothy R. Brick, Andreas Brandmaier

See Also

[semtree](#) [semforest](#)

<code>coef.semtree</code>	<i>Return the parameter estimates of a given leaf of a SEM tree</i>
---------------------------	---

Description

Return the parameter estimates of a given leaf of a SEM tree

Usage

```
## S3 method for class 'semtree'  
coef(object, ...)
```

Arguments

<code>object</code>	semtree. A SEM tree node.
<code>...</code>	Extra arguments. Currently unused.

<code>computePval_maxLR</code>	<i>Wrapper function for computing the maxLR corrected p value from strucchange</i>
--------------------------------	--

Description

Wrapper function for computing the maxLR corrected p value from strucchange

Usage

```
computePval_maxLR(maxLR, q, covariate, from, to, nrep)
```

Arguments

maxLR	maximum of the LR test statistics
q	number of free SEM parameters / degrees of freedom
covariate	covariate under evaluation. This is important to get the level of measurement from the covariate and the bin size for ordinal and categorical covariates.
from	numeric from interval (0, 1) specifying start of trimmed sample period. With the default from = 0.15 the first and last 15 percent of observations are trimmed. This is only needed for continuous covariates.
to	numeric from interval (0, 1) specifying end of trimmed sample period. By default, to is 1.
nrep	numeric. Number of replications used for simulating from the asymptotic distribution (passed to efpFunctional). Only needed for ordinal covariates.

Value

Numeric. p value for maximally selected LR statistic

Author(s)

Manuel Arnold

countPredictors	<i>Count predictors</i>
-----------------	-------------------------

Description

This function is a generic to count the number of occurrences of predictors in either a "semtree" or a "semforest". Note that this must not be confused with the importance of those predictors. To estimate importance, rather use permutation-based variable importance with varimp.

Usage

```
countPredictors(x, ...)

## S3 method for class 'semtree'
countPredictors(x, ...)

## S3 method for class 'semforest'
countPredictors(x, ...)
```

Arguments

x	An object representing either a SEM tree or a forest.
...	Additional arguments passed to methods.

Value

A result depending on the method.

diversityMatrix	<i>Diversity Matrix</i>
-----------------	-------------------------

Description

Computes a diversity matrix using a distance function between trees

Usage

```
diversityMatrix(forest, divergence = klsym, showProgressBar = TRUE)
```

Arguments

forest	A SEM forest
divergence	A divergence function such as hellinger or klsym
showProgressBar	Boolean. Show a progress bar.

evaluate	<i>Average Deviance of a Dataset given a Forest</i>
----------	---

Description

Evaluates the average deviance (-2LL) of a dataset given a forest.

Usage

```
evaluate(x, data = NULL, ...)
```

Arguments

x	A fitted semforest object
data	A data.frame
...	No extra parameters yet.

Value

Average deviance

Author(s)

Andreas M. Brandmaier

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[evaluateDataLikelihood](#), [semtree](#), [semforest](#)

evaluateDataLikelihood

Compute the Negative Two-Loglikelihood of some data given a model (either OpenMx or lavaan)

Description

This helper function is used in the [semforest varimp](#) and [proximity](#) aggregate functions.

Usage

```
evaluateDataLikelihood(
  model,
  data,
  data_type = "raw",
  loglik = c("default", "model", "mvn")
)
```

Arguments

model	A model as used in semtree and semforest .
data	Data set to apply to a fitted model.
data_type	Type of data ("raw", "cov", "cor")
loglik	Character. Either 'model' for model-based evaluation or 'mvn' for multivariate normal density.

Value

Returns a -2LL model fit for the model

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[semtree](#), [semforest](#)

evaluateTree

Evaluate Tree -2LL

Description

A helper function to evaluate the negative two log-likelihood (-2LL) of leaf (terminal) nodes for a dataset. When given a [semtree](#) and a unique dataset, the model estimates -2LL for the tree parameters and data subsets that fit the tree branching criteria.

Usage

```
evaluateTree(
  tree,
  test_set,
  data_type = "raw",
  leaf_ids = NULL,
  loglik = c("default", "model", "mvn")
)
```

Arguments

tree	A fitted semtree object
test_set	Dataset to fit to a fitted semtree object
data_type	type of data ("raw", "cov", "cor")
leaf_ids	Identifies which nodes are leaf nodes. Default is NULL, which checks model for leaf nodes and fills this information in automatically.
loglik	Algorithm to compute log likelihood. The default is 'model' and refers to a model-based computation. This is preferable because it is more general. As an alternative, 'mvn' computes the log likelihood based on the multivariate normal density and the model-implied mean and covariance matrix.

Value

A list with two elements:

deviance	Combined -2LL for leaf node models of the tree.
num_models	Number of leaf nodes used for the deviance calculations.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[evaluateDataLikelihood](#), [semtree](#), [semforest](#)

findOtherSplits	<i>Find Other Node Split Values</i>
-----------------	-------------------------------------

Description

Search tool to search nodes for alternative splitting values found during the [semtree](#) process. Given a particular node, competing split values are listed assuming they also meet the criteria for a significant splitting value as set by [semtree.control](#).

Usage

```
findOtherSplits(node, tree)
```

Arguments

node	A node from a semtree object.
tree	A semtree object which the node is part of.

Value

A `data.frame()` with rows corresponding to the variable names and split values for alternative splits found in the node of interest. ...

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

fitSubmodels	<i>Fit multigroup model for evaluating a candidate split. This function has the fundamental logic for fitting two-group models in different variants</i>
--------------	--

Description

Fit multigroup model for evaluating a candidate split. This function has the fundamental logic for fitting two-group models in different variants

Usage

```
fitSubmodels(
  model,
  subset1,
  subset2,
  control,
  invariance = NULL,
  return.models = FALSE
)
```

Arguments

model	A model specification that is used as template for each of the two groups
subset1	Dataset for the first group model
subset2	Dataset for the second group model
control	a <code>semtree.control</code> object
invariance	fit models with invariant parameters if given. NULL otherwise (default).
return.models	boolean. Return the fitted models returns NA if fit fails

getDepth	<i>Get the depth (or, height) a tree.</i>
----------	---

Description

Returns the length of the longest path from a root node to a leaf node.

Usage

```
getDepth(tree)
```

Arguments

tree	A semtree object
------	----------------------------------

Author(s)

Andreas M. Brandmaier

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

getHeight

Determine Height of a Tree

Description

Returns height of a SEM Tree, which equals to the length of the longest path from root to a terminal node.

Usage

getHeight(tree)

Arguments

tree A SEM tree.

Details

Example: A SEM tree with only the root node has depth 0. A SEM tree with one decision node has depth 1.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

getLeafs	<i>Get a list of all leafs in a tree</i>
----------	--

Description

Get a list of all leafs in a tree by recursively searching the tree starting at the given node (if not data object is given. If data is given, the function returns the leafs that are predicted for each row of the given data.

Usage

```
getLeafs(tree, data = NULL)
```

Arguments

tree	A semtree object
data	A <code>data.frame</code>

Author(s)

Andreas M. Brandmaier

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

getNodeById	<i>Get Node By Id</i>
-------------	-----------------------

Description

Return a node matching a given node ID

Usage

```
getNodeById(tree, id)
```

Arguments

tree	A SEM Tree object.
id	Numeric. A Node id.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

getNumNodes	<i>Tree Size</i>
-------------	------------------

Description

Counts the number of nodes in a tree.

Usage

```
getNumNodes(tree)
```

Arguments

tree	A SEM tree object.
------	--------------------

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

getParDiffForest	<i>Return list with parameter differences of a forest</i>
------------------	---

Description

Returns a list of tables with some measure of parameter differences between post-split nodes.

Usage

```
getParDiffForest(forest, measure = "wald", normalize = FALSE)
```

Arguments

forest	a semforest object.
measure	a character. "wald" (default) gives the squared parameter differences divided by their pooled standard errors. "test" gives the contributions of the parameters to the test statistic. "raw" gives the absolute values of the parameter differences.
normalize	logical value; if TRUE parameter differences of each split are divided by sum of all differences the corresponding split. Set to FALSE by default.

Value

A list with data.frames containing parameter differences for each tree of the forest. The rows of the data.frames correspond to the non-leaf nodes of the respective trees. The first column contains the name of the predictor variables and the remaining columns contain the parameter differences. The rows of the data.frames are named by the node IDs as given `getNodeById` and the columns are named as in `coef`.

Author(s)

Manuel Arnold

<code>getParDiffTree</code>	<i>Return table with parameter differences of a tree</i>
-----------------------------	--

Description

Returns a table with some measure of parameter differences between post-split nodes.

Usage

```
getParDiffTree(tree, measure = "wald", normalize = FALSE)
```

Arguments

<code>tree</code>	a <code>semtree</code> object.
<code>measure</code>	a character. "wald" (default) gives the squared parameter differences divided by their pooled standard errors. "test" gives the contributions of the parameters to the test statistic. "raw" gives the absolute values of the parameter differences.
<code>normalize</code>	logical value; if TRUE parameter differences of each split are divided by sum of all differences the corresponding split. Set to FALSE by default.

Value

A matrix containing parameter differences. The matrix has n rows and k columns, where n is the number of non-leaf nodes of the tree and k is the number of model parameters. The rows are named by the node IDs as given `getNodeById` and the columns are named as in `coef`.

Author(s)

Manuel Arnold

getTerminalNodes	Returns all leafs of a tree
------------------	-----------------------------

Description

Returns all leafs (=terminal nodes) of a tree.

Usage

```
getTerminalNodes(tree)
```

Arguments

tree	A semtree object.
------	-------------------

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

hellinger	Hellinger distance of two multivariate normal distributions
-----------	---

Description

Compute distance between two multivariate normal distribution of matching dimension. To be used with the diversity function.

Usage

```
hellinger(mu1, cov1, mu2, cov2)
```

Arguments

mu1	Mean vector of the first distribution
cov1	Covariance matrix of the first distribution
mu2	Mean vector of the second distribution
cov2	Covariance matrix of the second distribution

isLeaf	<i>Test whether a semtree object is a leaf.</i>
--------	---

Description

Tests whether a semtree object is a leaf. Returns TRUE or FALSE.

Usage

```
isLeaf(tree)
```

Arguments

tree	A semtree object
------	----------------------------------

Author(s)

Andreas M. Brandmaier

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

kl	<i>Kullback-Leiber Divergence</i>
----	-----------------------------------

Description

Divergence measures for multivariate normal distributions as used in the diversityMatrix function.

Usage

```
kl(mu1, cov1, mu2, cov2)
```

Arguments

mu1	Mean vector
cov1	Covariance matrix
mu2	Mean vector
cov2	Covariance matrix

klsym	<i>Symmetric version of Kullback-Leiber Divergence</i>
-------	--

Description

Distance measures for multivariate normal distributions as used in the diversityMatrix function.

Usage

```
klsym(mu1, cov1, mu2, cov2)
```

Arguments

mu1	Mean vector
cov1	Covariance matrix
mu2	Mean vector
cov2	Covariance matrix

lgcm	<i>Simulated Linear Latent Growth Curve Data</i>
------	--

Description

This data set provides simple data to fit with a LGCM.

Format

lgcm is a matrix containing 400 rows and 8 columns of simulated data. Longitudinal observations are o1-o5. Covariates are agegroup, training, and noise.

Author(s)

Andreas M. Brandmaier <brandmaier@mpib-berlin.mpg.de>

merge.semforest	<i>Merge two SEM forests</i>
-----------------	------------------------------

Description

This overrides generic base::merge() to merge two forests into one.

Usage

```
## S3 method for class 'semforest'
merge(x, y, ...)
```

Arguments

x	A SEM Forest
y	A second SEM Forest
...	Extra arguments. Currently unused.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[semtree](#)

modelEstimates	<i>Returns all estimates of a tree</i>
----------------	--

Description

Return model estimates of the tree.

Usage

```
modelEstimates(tree, ...)
```

Arguments

tree	A semtree object.
...	Optional arguments.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

outliers

Find outliers based on case proximity

Description

Compute outlier score based on proximity matrix.

Usage

```
outliers(prox)
```

Arguments

prox A proximity matrix.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[proximity](#)

parameters	<i>SEMtrees Parameter Estimates Table</i>
------------	---

Description

Returns a table of parameters with columns corresponding to freely estimated parameters and rows corresponding to nodes in the tree.

Usage

```
parameters(tree, leafs.only = TRUE)
```

Arguments

tree	A SEMtree object obtained from semtree
leafs.only	Default = TRUE. Only the terminal nodes (leafs) are printed. If set to FALSE, all node parameters are written to the <code>data.frame</code> .

Details

The row names of the resulting data frame correspond to internal node ids and the column names correspond to parameters in the SEM. Standard errors of the estimates can be obtained from [parameters](#).

Value

Returns a `data.frame` with rows for parameters and columns for terminal nodes.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[semtree](#), [semtree.control](#), [se](#)

partialDependence	<i>Compute partial dependence</i>
-------------------	-----------------------------------

Description

Compute the partial dependence of a predictor, or set of predictors, on a model parameter.

Usage

```
partialDependence(  
  x,  
  data,  
  reference.var,  
  support = 20,  
  points = NULL,  
  mc = NULL,  
  FUN = "median",  
  ...  
)
```

Arguments

x	An object for which a method exists
data	Optional data.frame that was used to train the model.
reference.var	Character vector, referring to the (independent) reference variable or variables for which partial dependence is calculated. Providing two (or more) variables allows for probing interactions, but note that this is computationally expensive.
support	Integer. Number of grid points for interpolating the reference.var. Alternatively, use points for one or more variables named in reference.var.
points	Named list, with elements corresponding to reference.var. Use this argument to provide specific points for which to obtain marginal dependence values; for example, the mean and +/- 1SD of reference.var.
mc	Integer. If mc is not NULL, the function will sample mc number of rows from data with replacement, to estimate marginal dependency using Monte Carlo integration. This is less computationally expensive.
FUN	Character string with function used to integrate predictions across all elements of x.
...	Extra arguments passed to FUN.

Author(s)

Caspar J. Van Lissa, , Andreas M. Brandmaier

partialDependence_data

Create dataset to compute partial dependence

Description

Create a dataset with fixed values for `reference.var` for all other values of data, or using `mc` random samples from data (Monte Carlo integration).

Usage

```
partialDependence_data(
  data,
  reference.var,
  support = 20,
  points = NULL,
  mc = NULL,
  keep_id = FALSE
)
```

Arguments

<code>data</code>	The <code>data.frame</code> that was used to train the model.
<code>reference.var</code>	Character vector, referring to the (independent) reference variable or variables for which partial dependence is calculated. Providing two (or more) variables allows for probing interactions, but note that this is computationally expensive.
<code>support</code>	Integer. Number of grid points for interpolating the <code>reference.var</code> . Alternatively, use <code>points</code> for one or more variables named in <code>reference.var</code> .
<code>points</code>	Named list, with elements corresponding to <code>reference.var</code> . Use this argument to provide specific points for which to obtain marginal dependence values; for example, the mean and +/- 1SD of <code>reference.var</code> .
<code>mc</code>	Integer. If <code>mc</code> is not <code>NULL</code> , the function will sample <code>mc</code> number of rows from <code>data</code> without replacement, to estimate
<code>keep_id</code>	Boolean. Default is <code>false</code> . Should output contain a row id column? marginal dependency using Monte Carlo integration. This is less computationally expensive.

Author(s)

Caspar J. Van Lissa

partialDependence_growth

Compute partial dependence for latent growth models

Description

Compute the partial dependence of a predictor, or set of predictors, on the predicted trajectory of a latent growth model.

Usage

```
partialDependence_growth(
  x,
  data,
  reference.var,
  support = 20,
  points = NULL,
  mc = NULL,
  FUN = "median",
  times = NULL,
  parameters = NULL,
  ...
)
```

Arguments

x	An object for which a method exists
data	Optional data.frame that was used to train the model.
reference.var	Character vector, referring to the (independent) reference variable or variables for which partial dependence is calculated. Providing two (or more) variables allows for probing interactions, but note that this is computationally expensive.
support	Integer. Number of grid points for interpolating the reference.var. Alternatively, use points for one or more variables named in reference.var.
points	Named list, with elements corresponding to reference.var. Use this argument to provide specific points for which to obtain marginal dependence values; for example, the mean and +/- 1SD of reference.var.
mc	Integer. If mc is not NULL, the function will sample mc number of rows from data with replacement, to estimate marginal dependency using Monte Carlo integration. This is less computationally expensive.
FUN	Character string with function used to integrate predictions across all elements of x.
times	Numeric matrix, representing the factor loadings of a latent growth model, with columns equal to the number of growth parameters, and rows equal to the number of measurement occasions.

parameters	Character vector of the names of the growth parameters; defaults to NULL, which assumes that the growth parameters are the only parameters and are in the correct order.
...	Extra arguments passed to FUN.

Author(s)

Caspar J. Van Lissa

plot.semtree

*Plot a SEM Tree***Description**

Plot the structure of a SEM tree and parameter estimates in its terminal nodes.

Usage

```
## S3 method for class 'semtree'
plot(
  x,
  type = 2,
  no.plot = FALSE,
  show.nonconvergence = FALSE,
  filter.parameters = NULL,
  ...
)
```

Arguments

x	A semtree object.
type	Tree layout type passed to prp . Default is 2.
no.plot	If TRUE, return the converted <code>rpart</code> object without drawing a plot.
show.nonconvergence	If TRUE, mark terminal nodes whose models did not converge.
filter.parameters	Character vector of parameter names to display in terminal nodes. If NULL, display all parameters. An error is raised if no names match a node's model parameters.
...	Additional arguments passed to prp .

Value

If `no.plot = TRUE`, an `rpart` object representing the tree. Otherwise, the result of [prp](#) is returned invisibly and a plot is drawn.

See Also

[semtree](#), [plotTreeStructure](#), [prp](#)

Examples

```
data(lgcm)
model <- lavaan::lavaan("o1 ~ mu*1; o1 ~~ v*o1", data = lgcm)
tree <- semtree(model, data = lgcm, predictors = "training",
               control = semtree_control(max.depth = 1))
plot(tree)
plot(tree, filter.parameters = "mu")
```

plotParDiffForest	<i>Plot parameter differences</i>
-------------------	-----------------------------------

Description

Visualizes parameter differences between post-split nodes in a forest with boxplots.

Usage

```
plotParDiffForest(
  forest,
  plot = "boxplot",
  measure = "wald",
  normalize = FALSE,
  predictors = NULL,
  title = TRUE
)
```

Arguments

forest	a semforest object.
plot	a character that specifies the plot typ. Available plot types are "boxplot" (default) and "jitter" for a jittered strip plot with mean and standard deviation.
measure	a character. "wald" (default) gives the squared parameter differences divided by their pooled standard errors. "test" gives the contributions of the parameters to the test statistic. "raw" gives the absolute values of the parameter differences.
normalize	logical value; if TRUE parameter differences of each split are divided by sum of all differences the corresponding split. Set to FALSE by default.
predictors	a character. Select predictors that are to be plotted.
title	logical value; if TRUE a title is added to the plot.

Author(s)

Manuel Arnold

plotParDiffTree	<i>Plot parameter differences</i>
-----------------	-----------------------------------

Description

Visualizes parameter differences between post-split nodes with different plot types.

Usage

```
plotParDiffTree(
  tree,
  plot = "ballon",
  measure = "wald",
  normalize = FALSE,
  title = TRUE,
  structure = FALSE
)
```

Arguments

tree	a semtree object.
plot	a character that specifies the plot typ. Available plot types are "ballon" (default), "heatmap", and "bar".
measure	a character. "wald" (default) gives the squared parameter differences divided by their pooled standard errors. "test" gives the contributions of the parameters to the test statistic. "raw" gives the absolute values of the parameter differences.
normalize	logical value; if TRUE parameter differences of each split are divided by sum of all differences the corresponding split. Set to FALSE by default.
title	logical value; if TRUE a title is added to the plot.
structure	logical value; if TRUE the structure of the tree is plotted on the right side.

Author(s)

Manuel Arnold

plotTreeStructure	<i>Plot tree structure</i>
-------------------	----------------------------

Description

Plots the structure of a semtree object. This function is similar to `plot.semtree`, but it does not print the parameter values in the leaf nodes and labels the leaf nodes instead.

Usage

```
plotTreeStructure(tree, type = 2, no.plot = FALSE, ...)
```

Arguments

tree	a semtree object.
type	Type of plot. See prp from rpart.plot .
no.plot	logical value; if TRUE structure of the tree is printed to the console.
...	additional arguments passed to prp from rpart.plot .

Author(s)

Manuel Arnold

predict.semforest	<i>Predict method for semtree and semforest</i>
-------------------	---

Description

Predict method for semtree and semforest

Usage

```
## S3 method for class 'semforest'
predict(object, data, type = "node_id", ...)
```

Arguments

object	Object of class semtree or 'semforest'.
data	New test data of class data.frame. If no data is provided, attempts to extract the data from the object.
type	Type of prediction. One of 'c('node_id')'. See Details.
...	further arguments passed to or from other methods.

Value

Object of class matrix.

Author(s)

Caspar J. van Lissa, Andreas Brandmaier

proximity	<i>Compute proximity matrix</i>
-----------	---------------------------------

Description

Compute a n by n matrix across all trees in a forest, where n is the number of rows in the data, reflecting the proportion of times two cases ended up in the same terminal node of a tree.

Usage

```
proximity(x, data, ...)
```

Arguments

x	An object for which a method exists.
data	A data.frame on which proximity is computed
...	Parameters passed to other functions.

Details

SEM Forest Case Proximity

Value

A matrix with dimensions [i, j] whose elements reflect the proportion of times case i and j were in the same terminal node of a tree.

Author(s)

Caspar J. Van Lissa, Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[semforest](#), [semtree](#)

Examples

```
nodeids <- structure(c(9, 3, 5, 7, 10, 4, 6, 8, 9, 3, 5, 7, 10, 4, 6, 8),
dim = c(4L, 4L))
class(nodeids) <- "semforest_node_id"
sims <- proximity(nodeids)
dd <- as.dist(1-sims)
hc <- hclust(dd)
groups <- cutree(hc, 2)
```

`prune`*Prune a SEM Tree or SEM Forest*

Description

Returns a new tree with a maximum depth selected by the user. can be used in conjunction with plot commands to view various pruning levels.

Usage

```
prune(object, ...)
```

Arguments

<code>object</code>	A semtree or semforest object.
<code>...</code>	Optional parameters, such as <code>max.depth</code> the maximum depth of each tree, <code>converged</code> indicating whether non-converged nodes should be pruned, or also <code>num.trees</code> when pruning a forest.

Details

The returned tree is only modified by the number of levels for the tree. This function does not reevaluate the data, but provides alternatives to reduce tree complexity. If the user would like to alter the tree by increasing depth, then `max.depth` option must be adjusted in the [semtree.control](#) object (provided further splits are able to be computed).

Value

Returns a [semtree](#) object.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[semtree](#), [semtree.control](#)

Examples

```
## Not run:
# prunes a tree to a maximum depth of five
prune(tree, max.depth = 5)

# prunes a tree that removes all nodes that have at least one
# non-converged daughter node
prune(tree, converged = TRUE)

## End(Not run)
```

se

SEMtrees Parameter Estimates Standard Error Table

Description

Returns a table of standard errors with columns corresponding to freely estimated standard errors and rows corresponding to nodes in the tree.

Usage

```
se(tree, leafs.only = TRUE)
```

Arguments

tree	A SEMtree object obtained from semtree
leafs.only	Default = TRUE. Only the terminal nodes (leafs) are printed. If set to FALSE, all node standard errors are written to the data.frame.

Details

The row names of the resulting data frame correspond to internal node ids and the column names correspond to standard errors in the SEM. Parameter estimates can be obtained from [parameters](#).

Value

Returns a data.frame with rows for parameters and columns for terminal nodes.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[semtree](#), [semtree.control](#), [parameters](#)

semforest

Create a SEM Forest

Description

Grows a SEM Forest from a template model and a dataset. This may take some time.

Usage

```
semforest(
  model,
  data,
  control = NULL,
  predictors = NULL,
  constraints = NULL,
  ...
)
```

Arguments

model	A template SEM. Same as in <code>semtree</code> .
data	A dataframe to create a forest from. Same as in <code>semtree</code> .
control	A <code>semforest</code> control object to set forest parameters.
predictors	An optional list of covariates. See <code>semtree</code> code example.
constraints	An optional list of covariates. See <code>semtree</code> code example.
...	Optional parameters.

Value

A `semforest` object.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Prindle, J. J., McArdle, J. J., & Lindenberger, U. (2016). Theory-guided exploration with structural equation model forests. *Psychological Methods*, 21(4), 566–582.

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71–86.

See Also

[semtree](#)

semforest.control	<i>SEM Forest Control Object</i>
-------------------	----------------------------------

Description

A SEM Forest control object to tune parameters of the forest learning algorithm.

Usage

```
semforest.control(
  num.trees = 5,
  sampling = "subsample",
  control = NA,
  mtry = 2,
  remove_dead_trees = TRUE,
  logfile = FALSE
)

semforest_score_control(...)

semforest_control(...)
```

Arguments

num.trees	Number of trees.
sampling	Sampling procedure. Can be subsample or bootstrap.
control	A SEM Tree control object. Will be generated by default.
mtry	Number of subsampled covariates at each node.
remove_dead_trees	Remove trees from forest that had runtime errors
logfile	Boolean/Character. If FALSE, no log file is written. Otherwise a logfile is written to the path given in this argument.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

Description

Structural equation model (SEM) trees combine SEM with decision-trees (a paradigm also known as recursive partitioning).

Usage

```
semtree(
  model,
  data = NULL,
  control = NULL,
  constraints = NULL,
  predictors = NULL,
  ...
)
```

Arguments

model	A template model specification from OpenMx using the <code>mxModel</code> function or a lavaan model using the <code>lavaan</code> function with option <code>do.fit=FALSE</code>). Model must be syntactically correct within the framework chosen, and converge to a solution.
data	A data.frame used for growing the tree. The order of manifest variables and predictors is not important when providing a dataset to <code>semtree</code> . Use the argument 'predictors' to explicitly specify which variables should be considered as split candidates.
control	A <code>semtree_control</code> object contains all hyper-parameters that govern the splitting procedure.
constraints	A <code>semtree.constraints</code> object setting model parameters as constrained from the beginning of the <code>semtree</code> computation. This includes the specification of focus parameters (i.e., parameter subsets that exclusively go into the function evaluating splits). Also, options for measurement invariance testing in trees are included.
predictors	A vector of variable names matching variable names in the data set. If NULL (default) all variables that are in data set and not part of the model are potential predictors.
...	Optional arguments.

Details

Core idea: Instead of assuming that one SEM fits all individuals equally well, SEM Trees recursively split the sample into subgroups based on covariates (e.g., age, gender, SES) such that model

parameters differ between subgroups. This results in a tree where each node contains an SEM, revealing heterogeneity in model structure or parameters across groups.

The package supports model specification in lavaan and OpenMx.

Calling `semtree` with an `mxModel` or `lavaan` model fits the template model to the entire data set and then recurses over the following steps until no further meaningful partition into sub groups is found:

1. Fit the model on the current node's data and compute model fit.
2. For each predictor, find the split point that best improves model fit – this is either done by likelihood ratio tests or score-based tests as determined in the `semtree.control`.
3. Select the best-performing predictor/split combination unless a stopping rule applies, e.g., split is not significant given a significance criterion (`alpha`), there are too few observations in a node (`min.N`, `min.bucket`, a maximum depth is reached, `max.depth`, or a custom stopping rule applies.
4. Continue the procedure independently on each resulting sub group

Predictors can be categorical (ordered or unordered) or continuous. When using unordered categorical predictors with many levels, the number of candidate partitions grows quickly, so limiting the predictor set can reduce computation and the number of multiple comparisons.

Splitting quality can be evaluated with three built-in strategies:

1. "naive" selection compares all possible split values across all predictors and chooses the best overall improvement.
2. "fair" selection uses a two-step procedure at each node: a first phase on half the sample identifies the best split value per predictor, and a second phase on the remaining data picks the most promising predictor among those candidates.
3. "score" relies on score-based statistics that provide faster evaluations while retaining favorable statistical properties for detecting parameter instabilities.

All other parameters controlling the tree growing process are adjusted in the `semtree.control` object.

Use `plot.semtree` to plot a SEM tree. Use `summary.semtree` to get a summary.

In order to get robust estimates of the importance of predictors, consider growing a `semforest`

Value

A `semtree` object. This can be further examined with `summary`, `plot`, and `print`.

Author(s)

Andreas M. Brandmaier, John J. Prindle, Manuel Arnold

References

- Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.
- Arnold, M., Voelkle, M. C., & Brandmaier, A. M. (2021). Score-guided structural equation model trees. *Frontiers in Psychology*, 11, Article 564403. <https://doi.org/10.3389/fpsyg.2020.564403>

See Also

[semtree.control](#), [summary.semtree](#), [parameters](#), [se](#), [prune.semtree](#), [subtree](#), [OpenMx](#), [lavaan](#)

Examples

```
{
  model <- lavaan::lavaan("bwt~~bwt; bwt~1")
  ctrl <- semtree_control(method="score",
    max.depth = 2, alpha = 0.01)
  bw <- with(MASS::birthwt, {
    ui <- factor(ui, labels=c("no", "yes"))
    ht <- factor(ht, labels=c("no", "yes"))
    smoke <- factor(smoke, labels=c("no", "yes"))
    data.frame( bwt, uterine_irritability=ui,
      hypertension=ht, smoke, num_premature_labours=ptl,
      physician_visits=ftv )
  })

  tree <- semtree(model = model, data = bw, control = ctrl)
  plot(tree)
}
```

semtree.constraints *SEM Tree Constraints Object*

Description

A SEM Tree constraints object holds information regarding specifics on how the tree is grown (similar to the control object). The SEM tree control object holds all information that is independent of a specific model whereas the constraints object holds information that is specific to a certain model (e.g., specifies differential treatment of certain parameters, e.g., by holding them constant across the forest).

Usage

```
semtree.constraints(local.invariance = NULL, focus.parameters = NULL)
```

Arguments

local.invariance

Vector of parameter names that are locally equal, that is, they are assumed to be equal when assessing a local split but allowed to differ subsequently.

focus.parameters

Vector of parameter names that exclusively are evaluated for between-group differences when assessing split candidates. If NULL all parameters add to the difference.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[semtree](#)

semtree_control	<i>SEM Tree Control Object</i>
-----------------	--------------------------------

Description

A `semtree_control` object contains parameters that determine the tree growing process. These parameters include choices of different split candidate selection procedures and hyperparameters of those. Calling the constructor without parameters creates a default control object. A number of tree growing methods are included in with this package: 1. 'naive' splitting takes the best split value of all possible splits on each covariate. 2. 'fair' selection is so called because it tests all splits on half of the data, then tests the best split value for each covariate on the other half of the data. The equal footing of each covariate in this two phase test removes bias from testing variables with many possible splits compared to those with few. 3. "fair3" does the phases described above, with an additional step of retesting all of the split values on the best covariate found in the second phase. Variations in the sample from subsetting are removed and bias in split selection further reduced. 4. 'score' implements modern score-based statistics.

Usage

```
semtree_control(
  method = c("naive", "score", "fair", "fair3"),
  min.N = NULL,
  max.depth = NA,
  alpha = 0.05,
  alpha.invariance = NA,
  exclude.heywood = TRUE,
  progress.bar = TRUE,
  verbose = FALSE,
  bonferroni = FALSE,
  use.all = FALSE,
  seed = NA,
  custom.stopping.rule = NA,
  mtry = NA,
  report.level = 0,
  exclude.code = NA,
```

```

linear = TRUE,
min.bucket = NULL,
missing = "ignore",
use.maxlr = FALSE,
strucchange.from = 0.15,
strucchange.to = NULL,
strucchange.nrep = 50000,
refit = TRUE,
ctsem_sd = FALSE,
loglik = c("default", "model", "mvn"),
check.convergence = TRUE,
chunk.random.samples = 0,
num_surrogates = 0
)

```

Arguments

method	Default: 'naive'. One of c("score", "fair", "naive") for a score-based testing scheme, an unbiased multi-step selection algorithm, or a naive take-the-best strategy.
min.N	Integer. Default: 'NULL' heuristically selects this number based on the number of parameters in the model. Minimum sample size per node used to determine whether splitting can continue. It is recommended to set 'min.N' explicitly.
max.depth	Integer. Default: NA. Maximum levels per a branch. Parameter for limiting tree growth.
alpha	Numeric. Default: 0.05. Significance level for splitting at a given node.
alpha.invariance	Numeric. alpha.invariance Default: NA. Significance level for invariance tests. If NA, the value of alpha is used.
exclude.heywood	Default: TRUE. Reports whether there is an identification problem in the covariance structure of an SEM tested.
progress.bar	Boolean. Default: TRUE. Option to enable or disable the progress bar for tree growth.
verbose	Boolean. Default: FALSE. Option to turn on or off <i>all</i> model messages during tree growth.
bonferroni	Boolean. Default: FALSE. Correct for multiple tests with Bonferroni type correction. p-values are adjusted for the number of variables tested.
use.all	Boolean. Treatment of missing variables. By default, missing values stay in a decision node. If TRUE, cases are distributed according to a maximum likelihood principle to the child nodes.
seed	Default: 'NA'. Set a random-number seed to make randomized parts of tree analysis reproducible (for example, in fair splitting or subsampling procedures).
custom.stopping.rule	Default: NA. Otherwise, this can be a boolean function with a custom stopping rule for tree growing.

mtry	Default: NA. Number of sample columns to use in SEMforest analysis.
report.level	Integer. Default: 0. Values up to 99 increase console reporting detail during tree growth and can help diagnose fitting or split-selection issues. This is rather appropriate for debugging than for end users.
exclude.code	Default: NA. NPSOL error code for exclusion from model fit evaluations when finding best split. Default: Models with errors during fitting are retained.
linear	Boolean. If TRUE (default), the structural equation model is assumed to not contain any nonlinear parameter constraints and scores are computed analytically, resulting in a shorter runtime. Only relevant for models fitted with OpenMx.
min.bucket	Integer. Minimum bucket size. This is the minimum size any node must have, such that a given split is considered valid. Minimum bucket size is a lower bound to the sample size in the terminal nodes of a tree.
missing	Missing value treatment. Default is ignore
use.maxlr	Boolean. Use MaxLR statistic for split point selection (as proposed by Arnold et al., 2021). This corrects the bias in the LR statistics incurred by testing multiple split points within one variable.
strucchange.from	Strucchange argument. See their package documentation.
strucchange.to	Strucchange argument. See their package documentation.
strucchange.nrep	Strucchange argument. See their package documentation.
refit	If TRUE (default) the initial model is fitted on the data provided to semtree .
ctsem_sd	If FALSE (default) no standard errors of CT model parameters are computed. Requesting standard errors increases runtime.
loglik	Character.Character. Algorithm to compute log likelihood. "default" depends on the chosen SEM package: "mvn" for lavaan and "model" for all other packages. "model" refers to model-based computation and is more general. "mvn" computes likelihood from the multivariate normal density using model-implied means and covariance matrices. multivariate normal density and the model-implied mean and covariance matrix.
check.convergence	Boolean. Should convergence be checked when growing a tree. Default: TRUE
chunk.random.samples	Integer. Experimental feature. Not for productive use. Controlling split-point subsampling for 'method = "naive"'. '0' (default) evaluates all eligible split points. Values '> 0' evaluate random chunks of split points, which can speed up tree growth on very large datasets at the cost of a less exhaustive search.
num_surrogates	Integer. Experimental feature. Not for productive use. Number of surrogates for surrogate splitting

Value

A control object containing a list of the above parameters.

Author(s)

Andreas M. Brandmaier, John J. Prindle, Manuel Arnold

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

Arnold, M., Voelkle, M. C., & Brandmaier, A. M. (2021). Score-guided structural equation model trees. *Frontiers in Psychology*, 11, Article 564403. <https://doi.org/10.3389/fpsyg.2020.564403>

See Also

[semtree](#)

Examples

```
# create a control object with an alpha level of 1%
my_control <- semtree_control(alpha=0.01)

# set the minimum number of cases per node to ten
my_control$min.N <- 10

# print contents of the control object
print(my_control)
```

strip	<i>Retain only basic tree structure</i>
-------	---

Description

Removes all elements of a semforest or semtree except for the tree structure and terminal node parameters. This is to reduce the heavy memory footprint of sem trees and forests.

Usage

```
strip(x, parameters = NULL)
```

Arguments

x	An object for which a method exists.
parameters	Character vector, referencing parameters in the SEM model. Defaults to NULL, in which case all free model parameters are returned.

Details

Objects of class `semforest` and `semtree` are very large, which complicates downstream operations such as making partial dependence plots, or using the model in interactive contexts (like Shiny apps). Running `strip` removes all elements of the model except for the tree structure and terminal node parameters. Note that some methods are no longer available for the resulting object - e.g., `varimp` requires the terminal node SEM models to compute the likelihood ratio.

Value

List

Examples

```
## Not run:
if(interactive()){
  #EXAMPLE1
}

## End(Not run)
```

subforest

Creates subsets of trees from forests

Description

Creates subsets of a forest. This can be used to subset a number of trees, e.g. `from:(from+num)`, or to remove all null (`type="nonnull"`) trees that were due to errors, or to randomly select a sub forest (`type=random`).

Usage

```
subforest(forest, num = NULL, type = "nonnull", from = 1)
```

Arguments

<code>forest</code>	A SEM Forest object.
<code>num</code>	Number of trees to select.
<code>type</code>	Either 'random' or 'nonnull' or NULL. First selects a random subset, second selects all non-null trees, third allows subsetting trees.
<code>from</code>	Starting index if <code>type=NULL</code> .

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

subtree	<i>SEMtree Partitioning Tool</i>
---------	----------------------------------

Description

The subtree function returns a tree from a selected node of the [semtree](#) returned tree.

Usage

```
subtree(tree, startNode = NULL, level = 0, foundNode = FALSE)
```

Arguments

tree	A SEMtree object obtained from semtree
startNode	Node id, which will be future root node (0 to max node number of tree)
level	Ignore. Only used internally.
foundNode	Ignore. Only used internally.

Details

The row names of the resulting data frame correspond to internal node ids and the column names correspond to standard errors in the SEM. Standard errors of the estimates can be obtained from [se](#).

Value

Returns a [semtree](#) object which is a partitioned tree from the input semtree.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

See Also

[semtree](#), [semtree.control](#)

surrogate	<i>use the current data 'mydata' and find a surrogate split</i>
-----------	---

Description

use the current data 'mydata' and find a surrogate split

Usage

```
surrogate(rule, mydata, agreement_threshold = 0.7)
```

Arguments

rule	a given split rule
mydata	a dataset
agreement_threshold	Numeric. Threshold for the agreement to consider a variable as surrogate

thinOut	<i>Thin Out a Forest</i>
---------	--------------------------

Description

This is a function that removes all "empty" trees, that is, those that contain only the root node and no splits

Usage

```
thinOut(x)
```

Arguments

x	A SEM forest
---	--------------

toTable	<i>Tabular Representation of a SEM Tree</i>
---------	---

Description

Converts a tree into a tabular representation. This may be useful as a textual representation for use in manuscripts.

Usage

```
toTable(tree, added.param.cols = NULL, round.param = 3)
```

Arguments

- tree A SEM Tree object.
- added.param.cols String. Add extra columns with parameter estimates. Pass a vector with the names of the parameters that should be rendered in the table.
- round.param Integer. Number of digits to round parameter estimates. Default is rounding to three digits after the decimal point.

Author(s)

Andreas M. Brandmaier

References

Brandmaier, A. M., Ram, N., Wagner, G. G., & Gerstorf, D. (in press). Terminal decline in well-being: The role of multi-indicator constellations of physical health and psychosocial correlates. *Developmental Psychology*.

varimp	<i>SEM Forest Variable Importance</i>
--------	---------------------------------------

Description

A function to calculate relative variable importance for selecting node splits over a [semforest](#) object.

Usage

```
varimp(
  forest,
  var.names = NULL,
  verbose = FALSE,
  eval.fun = evaluateTree,
  method = NULL,
  conditional = FALSE,
  strict = TRUE,
  ...
)
```

Arguments

forest	A semforest object
var.names	Covariates used in the forest creation process. NULL value will be automatically filled in by the function.
verbose	Boolean to print messages while function is running.
eval.fun	Default is evaluateTree function. The value of the -2LL of the leaf nodes is compared to baseline overall model.
method	Character. Define the method, with which importance is computed. The default is NULL and picks the appropriate permutation-based estimation method depending on whether no focus parameters are given ("permutation") or focus parameters are given ("permutationFocus")
conditional	Conditional variable importance if TRUE, otherwise marginal variable importance.
strict	Boolean. Default is TRUE. Only consider estimates from models if there were no model convergence problems. Otherwise, partial results are used, which may incur some downward bias.
...	Optional arguments.

Author(s)

Andreas M. Brandmaier, John J. Prindle

References

Brandmaier, A.M., Oertzen, T. v., McArdle, J.J., & Lindenberger, U. (2013). Structural equation model trees. *Psychological Methods*, 18(1), 71-86.

varimpConvergencePlot *Variable Importance Convergence Plot*

Description

Plot cumulative variable importance estimates as trees are added to a forest.

Usage

```
varimpConvergencePlot(
  x,
  lty = NULL,
  idx = NULL,
  legend.x = "topright",
  clw = 4,
  legend.cex = 1.5,
  ylim = NULL,
  cex.lab = 1,
  xlim = NULL,
  xlab = NULL,
  ylab = NULL,
  legend.bg = NULL,
  legend.bty = "n",
  na.omit = TRUE,
  extra.legend = NULL,
  aggregate = "mean",
  ...
)
```

Arguments

x	A variable importance object returned by varimp .
lty	Line types for the covariates. If NULL, selected automatically.
idx	Tree indices to display. If NULL, all trees are displayed.
legend.x	Legend position, or NULL to omit the legend.
clw	Numeric. Line width.
legend.cex	Character expansion factor for the legend.
ylim	Vertical axis limits. If NULL, selected automatically.
cex.lab	Character expansion factor for axis labels.
xlim	Horizontal axis limits. If NULL, selected automatically.
xlab	Horizontal axis label. If NULL, uses "Number of Trees".
ylab	Vertical axis label. If NULL, uses "Absolute Increase in Misfit".
legend.bg	Legend background color.

<code>legend.bty</code>	Legend box type.
<code>na.omit</code>	Boolean. If TRUE, missing estimates are excluded from cumulative means. If FALSE, they are replaced by zero.
<code>extra.legend</code>	Additional right margin for a legend outside the plotting region, or NULL.
<code>aggregate</code>	Aggregation method: "mean" or "median". Missing estimates are replaced by zero for cumulative medians.
<code>...</code>	Additional arguments passed to <code>graphics::plot()</code> .

Value

Draws a plot as a side effect.

See Also

[varimp](#)

Index

- * **biodiversity**
 - biodiversity, [4](#)
- * **datasets**
 - lgcm, [19](#)
- * **models**
 - boruta, [5](#)
 - semforest, [33](#)
 - semtree, [35](#)
- * **multivariate**
 - boruta, [5](#)
 - semforest, [33](#)
 - semtree, [35](#)
- * **semforest**
 - biodiversity, [4](#)
- * **tree**
 - boruta, [5](#)
 - semforest, [33](#)
 - semtree, [35](#)
- .SCALE_METRIC (semtree-package), [3](#)
- aggregateVarimp, [3](#)
- biodiversity, [4](#)
- boruta, [5](#)
- coef.semtree, [6](#)
- computePval_maxLR, [6](#)
- countPredictors, [7](#)
- diversityMatrix, [8](#)
- evaluate, [8](#)
- evaluateDataLikelihood, [9](#), [9](#), [11](#)
- evaluateTree, [10](#), [46](#)
- findOtherSplits, [11](#)
- fitSubmodels, [12](#)
- getDepth, [12](#)
- getHeight, [13](#)
- getLeafs, [14](#)
- getNodeById, [14](#)
- getNumNodes, [15](#)
- getParDiffForest, [15](#)
- getParDiffTree, [16](#)
- getTerminalNodes, [17](#)
- hellinger, [17](#)
- isLeaf, [18](#)
- kl, [18](#)
- klsym, [19](#)
- lavaan, [35–37](#)
- lgcm, [19](#)
- merge.semforest, [20](#)
- modelEstimates, [20](#)
- mxModel, [35](#), [36](#)
- nodeFunSmtree (semtree), [35](#)
- OpenMx, [37](#)
- outliers, [21](#)
- p.adjust.methods, [5](#)
- parameters, [22](#), [22](#), [32](#), [33](#), [37](#)
- partialDependence, [23](#)
- partialDependence_data, [24](#)
- partialDependence_growth, [25](#)
- plot.boruta (boruta), [5](#)
- plot.semforest (semforest), [33](#)
- plot.semforest.varimp (varimp), [45](#)
- plot.semtree, [26](#), [36](#)
- plotParDiffForest, [27](#)
- plotParDiffTree, [28](#)
- plotTreeStructure, [27](#), [28](#)
- predict.semforest, [29](#)
- print.boruta (boruta), [5](#)
- print.semforest (semforest), [33](#)

print.semforest.control
 (semforest.control), 34
print.semforest.varimp (varimp), 45
print.semtree (semtree), 35
print.semtree.control
 (semtree_control), 38
proximity, 9, 21, 30
prp, 26, 27
prune, 31
prune.semtree, 37

se, 22, 32, 37, 43
semforest, 4, 6, 8–11, 30, 33, 36, 45, 46
semforest.control, 5, 34
semforest_control (semforest.control),
 34
semforest_score_control
 (semforest.control), 34
semtree, 6, 9–12, 14, 18, 20, 22, 26, 27,
 30–33, 35, 38, 40, 41, 43
semtree-package, 3
semtree.constraints, 35, 37
semtree.control, 5, 11, 22, 31, 33, 36, 37, 43
semtree.control (semtree_control), 38
semtree_control, 35, 38
strip, 41
subforest, 42
subtree, 37, 43
summary.semtree, 36, 37
summary.semtree (semtree), 35
surrogate, 44

thinOut, 44
toLatex.semtree (semtree), 35
toTable, 45

varimp, 9, 42, 45, 47, 48
varimpConvergencePlot, 47
varimpConvergencePlot (varimp), 45