

Package ‘phylopath’

August 20, 2026

Type Package

Title Perform Phylogenetic Path Analysis

Version 1.4.0

Maintainer Wouter van der Bijl <wouter@zoology.ubc.ca>

Description A comprehensive and easy to use R implementation of confirmatory phylogenetic path analysis as described by Von Hardenberg and Gonzalez-Voyer (2012) <[doi:10.1111/j.1558-5646.2012.01790.x](https://doi.org/10.1111/j.1558-5646.2012.01790.x)>. Note that the required package 'ggm' depends on 'graph' from Bioconductor, which is not installed automatically; the simplest route is `install.packages("`BiocManager"); BiocManager::install("`phylopath")`.

URL <https://ax3man.github.io/phylopath/>

BugReports <https://github.com/Ax3man/phylopath/issues>

License GPL-3

LazyData TRUE

Depends R (>= 2.10)

Imports ape (>= 4.1), future.apply, ggm (>= 2.3), ggplot2 (>= 3.0.0),
ggraph (>= 1.0.0), igraph (>= 1.0.1), phylolm (>= 2.5), purrr
(>= 0.2.3), rlang (>= 1.0.0), tibble

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Encoding UTF-8

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Wouter van der Bijl [aut, cre]

Repository CRAN

Date/Publication 2026-08-20 09:20:02 UTC

Contents

as.data.frame.fitted_DAG	2
average	3
average_DAGs	4
best	6
choice	7
cichlids	8
cichlids_tree	8
coef.fitted_DAG	9
coef_plot	10
DAG	11
define_model_set	12
est_DAG	13
phylo_path	14
plot.DAG	17
plot.fitted_DAG	18
plot.phylopath_summary	20
plot_model_set	21
red_list	22
red_list_tree	23
rhino	24
rhino_tree	24
show_warnings	25
Index	26

as.data.frame.fitted_DAG	<i>Extract the paths of a fitted causal model.</i>
--------------------------	--

Description

Returns the estimated paths of a fitted_DAG as a long data.frame, one row per path, which is a convenient starting point for custom tables and figures. Paths that are absent from the causal model are not included.

Usage

```
## S3 method for class 'fitted_DAG'  
as.data.frame(x, row.names = NULL, optional = FALSE, ..., order_by = "default")
```

Arguments

- x An object of class fitted_DAG.
- row.names, optional, ... Ignored, present for consistency with the generic.

`order_by` Either "default", to order the paths by their position in the causal model, "causal", to follow the paths downstream, or "strength", to order them by the size of their coefficients. Ordering by strength is not available for models that contain both binary and continuous variables, since their coefficients are not comparable, see `est_DAG()`. Ordering by cause is not available for cyclical models, which `average()` can produce when the direction of a path is not resolved.

Value

A data.frame with columns from, to, coef and se, as well as lower and upper if the model was fitted with boot larger than 0.

Examples

```
d <- DAG(LS ~ BM, NL ~ BM, DD ~ NL + LS)
d_fitted <- est_DAG(d, rhino, rhino_tree, 'lambda')
as.data.frame(d_fitted)
as.data.frame(d_fitted, order_by = 'strength')
```

average	<i>Extract and average the best supported models from a phylogenetic path analysis.</i>
---------	---

Description

Extract and average the best supported models from a phylogenetic path analysis.

Usage

```
average(phylopath, cut_off = 2, avg_method = "conditional", ...)
```

Arguments

<code>phylopath</code>	An object of class phylopath.
<code>cut_off</code>	The CICc cut-off used to select the best models. Use Inf to average over all models. Use the <code>best()</code> function to only use the top model, or <code>choice()</code> to select any single model.
<code>avg_method</code>	Either "full" or "conditional". The methods differ in how they deal with averaging a path coefficient where the path is absent in some of the models. The full method sets the coefficient (and the variance) for the missing paths to zero, meaning paths that are missing in some models will shrink towards zero. The conditional method only averages over models where the path appears, making it more sensitive to small effects. Following von Hardenberg & Gonzalez-Voyer 2013, conditional averaging is set as the default.
<code>...</code>	Arguments to pass to <code>phylolm::phylolm</code> and <code>phylolm::phyloglm</code> . Provide <code>boot = K</code> parameter to enable bootstrapping, where K is the number of bootstrap replicates. If you specified other options in the original <code>phylo_path</code> call you don't need to specify them again.

Value

An object of class fitted_DAG.

See Also

[est_DAG\(\)](#) for what the coefficients mean.

Examples

```
candidates <- define_model_set(
  A = NL ~ RS,
  B = RS ~ NL + BM,
  .common = c(LS ~ BM, DD ~ NL, NL ~ BM)
)
p <- phylo_path(candidates, rhino, rhino_tree)
summary(p)

# Models A and B have similar support, so we may decide to take
# their average.

avg_model <- average(p)
# Print the average model to see coefficients, se and ci:
avg_model

## Not run:
# Plot to show the weighted graph:
plot(avg_model)

# One can see that an averaged model is not necessarily a DAG itself.
# This model actually has a path in two directions.

# Note that coefficients that only occur in one of the models become much
# smaller when we use full averaging:

coef_plot(avg_model)
coef_plot(average(p, method = 'full'))

## End(Not run)
```

average_DAGs

Perform model averaging on a list of DAGs.

Description

Perform model averaging on a list of DAGs.

Usage

```
average_DAGs(
  fitted_DAGs,
  weights = rep(1, length(coef)),
  avg_method = "conditional",
  ...
)
```

Arguments

<code>fitted_DAGs</code>	A list of <code>fitted_DAG</code> objects containing coefficients and standard errors, usually obtained by using <code>est_DAG()</code> on several DAGs.
<code>weights</code>	A vector of associated model weights.
<code>avg_method</code>	Either "full" or "conditional". The methods differ in how they deal with averaging a path coefficient where the path is absent in some of the models. The full method sets the coefficient (and the variance) for the missing paths to zero, meaning paths that are missing in some models will shrink towards zero. The conditional method only averages over models where the path appears, making it more sensitive to small effects. Following von Hardenberg & Gonzalez-Voyer 2013, conditional averaging is set as the default.
<code>...</code>	Use of the ellipses is deprecated.

Value

An object of class `fitted_DAG`, including standard errors and confidence intervals.

See Also

`est_DAG()` for what the coefficients mean.

Examples

```
# Normally, I would advocate the use of the phylo_path and average
# functions, but this code shows how to average any set of models. Note
# that not many checks are implemented, so you may want to be careful and
# make sure the DAGs make sense and contain the same variables!
candidates <- define_model_set(
  A = NL ~ BM,
  B = NL ~ LS,
  .common = c(LS ~ BM, DD ~ NL)
)
fit_cand <- lapply(candidates, est_DAG, rhino, rhino_tree,
  model = 'lambda', method = 'logistic_MPLE')
ave_cand <- average_DAGs(fit_cand)
coef_plot(ave_cand)
```

best	<i>Extract and estimate the best supported model from a phylogenetic path analysis.</i>
------	---

Description

Extract and estimate the best supported model from a phylogenetic path analysis.

Usage

```
best(phylopath, ...)
```

Arguments

phylopath	An object of class phylopath.
...	Arguments to pass to phylolm::phylolm and phylolm::phyloglm . Provide boot = K parameter to enable bootstrapping, where K is the number of bootstrap replicates. If you specified other options in the original phylo_path call you don't need to specify them again.

Value

An object of class fitted_DAG.

See Also

[est_DAG\(\)](#) for what the coefficients mean.

Examples

```
candidates <- define_model_set(
  A = NL ~ BM,
  B = NL ~ LS,
  .common = c(LS ~ BM, DD ~ NL)
)
p <- phylo_path(candidates, rhino, rhino_tree)
best_model <- best(p)
# Print the best model to see coefficients, se and ci:
best_model
# Plot to show the weighted graph:
plot(best_model)
```

choice	<i>Extract and estimate an arbitrary model from a phylogenetic path analysis.</i>
--------	---

Description

Extract and estimate an arbitrary model from a phylogenetic path analysis.

Usage

```
choice(phylopath, choice, ...)
```

Arguments

phylopath	An object of class phylopath.
choice	A character string of the name of the model to be chosen, or the index in model_set.
...	Arguments to pass to phylolm::phylolm and phylolm::phyloglm . Provide boot = K parameter to enable bootstrapping, where K is the number of bootstrap replicates. If you specified other options in the original phylo_path call you don't need to specify them again.

Value

An object of class fitted_DAG.

See Also

[est_DAG\(\)](#) for what the coefficients mean.

Examples

```
candidates <- define_model_set(
  A = NL ~ BM,
  B = NL ~ LS,
  .common = c(LS ~ BM, DD ~ NL)
)
p <- phylo_path(candidates, rhino, rhino_tree)
my_model <- choice(p, "B")
# Print the best model to see coefficients, se and ci:
my_model
# Plot to show the weighted graph:
plot(my_model)
```

cichlids

Cichlid traits and the evolution of cooperative breeding.

Description

A data set with binary traits, used in an analysis on the evolution of cooperative breeding by Dey et al 2017. Variable names are shortened for easy of use and consist of cooperative breeding (C), mating system (M), parental care (P), social grouping (G) and diet (D). All traits are coded as two level factors.

Usage

cichlids

Format

An object of class `data.frame` with 69 rows and 5 columns.

Source

Dey, C.J., O'Connor, C.M., Wilkinson, H., Shultz, S., Balshine, S. & Fitzpatrick, J.L. 2017. Direct benefits and evolutionary transitions to complex societies. *Nat Ecol Evol* 1: 137.

cichlids_tree

Cichlid phylogeny.

Description

The phylogenetic tree of cichlid species that accompanies the cichlids dataset. The phylogeny is based on five nuclear genes and three mitochondrial genes.

Usage

cichlids_tree

Format

An object of class `phylo` of length 4.

Source

Dey, C.J., O'Connor, C.M., Wilkinson, H., Shultz, S., Balshine, S. & Fitzpatrick, J.L. 2017. Direct benefits and evolutionary transitions to complex societies. *Nat Ecol Evol* 1: 137.

coef.fitted_DAG	<i>Extract the coefficients and confidence intervals of a fitted causal model.</i>
-----------------	--

Description

coef() returns the path coefficients as a named vector, and confint() the bounds of their confidence intervals. Both name each path as "from -> to" and return them in the same order, so they can be combined with cbind(). Paths that are absent from the causal model are not included. For a data frame with the standard errors alongside the coefficients, see [as.data.frame.fitted_DAG\(\)](#).

Usage

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

## S3 method for class 'fitted_DAG'
confint(object, parm, level = 0.95, ...)
```

Arguments

object	An object of class fitted_DAG.
...	Ignored, present for consistency with the generic.
parm	The paths for which to return intervals, given as names or indices. Defaults to all of them.
level	The confidence level, which has to be 0.95 for now.

Details

Note that these are not the parametric intervals that confint() returns for most model objects, but the ones that were stored when the model was made: bootstrap intervals after fitting with boot, and normal intervals around the averaged estimate for a model from [average\(\)](#).

Value

For coef(), a named numeric vector with one element per path. For confint(), a matrix with a row per path and the lower and upper bound in its two columns.

Examples

```
d <- DAG(LS ~ BM, NL ~ BM, DD ~ NL + LS)
d_fitted <- est_DAG(d, rhino, rhino_tree, 'lambda')
coef(d_fitted)

# Confidence intervals require the model to be fitted with bootstrapping.

d_boot <- est_DAG(d, rhino, rhino_tree, 'lambda', boot = 100)
```

```
confint(d_boot)
cbind(coef = coef(d_boot), confint(d_boot))
```

coef_plot

Plot path coefficients and their confidence intervals or standard errors.

Description

Plot path coefficients and their confidence intervals or standard errors.

Usage

```
coef_plot(
  fitted_DAG,
  error_bar = "ci",
  order_by = "default",
  from = NULL,
  to = NULL,
  reverse_order = FALSE
)
```

Arguments

fitted_DAG	A fitted DAG, usually obtained by best() , average() or est_DAG() .
error_bar	Whether to use confidence intervals ("ci") or standard errors ("se") as error bars. Will force standard errors with a message if confidence intervals are not available.
order_by	By "default", the paths are ordered as in the the model that is supplied. Usually this is in the order that was established by [phylo_path()] for all combined graphs. This can be change to "causal" to do a reordering based on the model at hand, or to "strength" to order them by the standardized regression coefficient.
from	Only show path coefficients from these nodes. Supply as a character vector.
to	Only show path coefficients to these nodes. Supply as a character vector.
reverse_order	If TRUE, the paths are plotted in reverse order. Particularly useful in combination with ggplot2::coord_flip() to create horizontal versions of the plot.

Value

A ggplot object.

Examples

```
d <- DAG(LS ~ BM, NL ~ BM, DD ~ NL + LS)
plot(d)
d_fitted <- est_DAG(d, rhino, rhino_tree, 'lambda')
plot(d_fitted)
coef_plot(d_fitted, error_bar = "se")
# to create a horizontal version, use this:
coef_plot(d_fitted, error_bar = "se", reverse_order = TRUE) + ggplot2::coord_flip()
```

DAG

Directed acyclic graphs (DAGs)

Description

This function is a simple wrapper around the function from the `ggm` package with the same name. The only differences are that the `order` argument defaults to `TRUE` and that it adds a `DAG` class for easy plotting. Typically, one would use `define_model_set()` to create models for use with the `phylopath` package.

Usage

```
DAG(..., order = TRUE)
```

Arguments

<code>...</code>	a sequence of model formulae
<code>order</code>	logical, defaulting to <code>TRUE</code> . If <code>TRUE</code> the nodes of the DAG are permuted according to the topological order. If <code>FALSE</code> the nodes are in the order they first appear in the model formulae (from left to right). For use in the <code>phylopath</code> package, this should always be kept to <code>TRUE</code> , but the argument is available to avoid potential problems with masking the function from other packages.

Details

Supply a formulas for the model as arguments. Formulas should be of the form `child ~ parent``` and describe each path `~ parent1 + parent2`. Finally, an isolate (unconnected variable) can be included as being connected to itself `late ~ late``.

Value

An object of classes `matrix` and `DAG`

Examples

```
# Use formula notation to create DAGs:
plot(DAG(A~B, B~C))
# Use + to easily add multiple parents to a node:
plot(DAG(A~B+C))
# Add a node as it's own parent to create an isolate:
plot(DAG(A~B+C, D~D))
```

define_model_set	<i>Define a model set.</i>
------------------	----------------------------

Description

This is a convenience function to quickly and clearly define a set of causal models. Supply a list of formulas for each model, using either `c()`. Formulas should be of the form `child ~ parent` and describe each path in your model. Multiple children of a single parent can be combined into a single formula: `child ~ parent1 + parent2`.

Usage

```
define_model_set(..., .common = NULL)
```

Arguments

<code>...</code>	Named arguments, which each are a lists of formulas defining the paths of a causal model.
<code>.common</code>	A list of formulas that contain causal paths that are common to each model.

Details

This function uses [ggm::DAG\(\)](#).

Value

A list of models, each of class `matrix` and `DAG`.

Examples

```
(m <- define_model_set(
  A = c(a~b, b~c),
  B = c(b~a, c~b),
  .common = c(d~a)))
plot_model_set(m)
```

est_DAG	<i>Estimate path coefficients for a DAG.</i>
---------	--

Description

This function will estimate the path coefficients for a DAG, and return them with uncertainty.

Usage

```
est_DAG(
  DAG,
  data,
  tree,
  model = "lambda",
  method = "logistic_MPLE",
  boot = 0,
  ...
)
```

Arguments

DAG	A directed acyclic graph, typically created with DAG.
data	A <code>data.frame</code> with data. If you have binary variables, make sure they are either character values or factors!
tree	A phylogenetic tree of class <code>phylo</code> .
model	The evolutionary model used for the regressions on continuous variables. See phylolm::phylolm for options and details. Defaults to Pagel's lambda model
method	The estimation method for the binary models. See phylolm::phyloglm for options and details. Defaults to logistic MPLE.
boot	The number of bootstrap replicates used to estimate confidence intervals.
...	Arguments passed on to <code>phylolm</code> : lower.bound: optional lower bound for the optimization of the phylogenetic model parameter. upper.bound: optional upper bound for the optimization of the phylogenetic model parameter. starting.value: optional starting value for the optimization of the phylogenetic model parameter. measurement_error: a logical value indicating whether there is measurement error <code>sigma2_error</code> (see Details). Arguments passed on to <code>phyloglm</code> : btol: bound on the linear predictor to bound the searching space. log.alpha.bound: bound for the log of the parameter alpha. start.beta: starting values for beta coefficients. start.alpha: starting values for alpha (phylogenetic correlation).

Details

Continuous variables are standardized before fitting, that is, centered and divided by their standard deviation, while binary variables are left as they are. In the two usual cases this means the coefficients can be compared with each other directly:

- When every variable is continuous, each coefficient is the change in the outcome, in standard deviations, for an increase of one standard deviation in the predictor. These are standardized regression coefficients, as reported by von Hardenberg & Gonzalez-Voyer.
- When every variable is binary, each coefficient is the log odds ratio for the outcome between the two levels of the predictor. A level to level contrast is a natural unit that every binary variable has, so these are comparable too.

When a model contains binary *and* continuous variables, its coefficients can no longer be compared with one another, for two separate reasons. Paths into a binary variable are log odds ratios while paths into a continuous variable are standardized regression coefficients, which are different units entirely. And a binary predictor contributes a contrast between its two levels, which for a variable where a proportion p of species has one of the levels amounts to $1 / \sqrt{p * (1 - p)}$ standard deviations. That is always more than two, so the coefficients of binary predictors are inflated relative to those of continuous predictors in the same model.

Value

An object of class `fitted_DAG`, which is a list with the following components:

coef a matrix of path coefficients, with predictors in the rows and responses in the columns. Absent paths are 0.

se a matrix of standard errors of the coefficients.

lower, upper matrices with the bounds of the bootstrapped confidence intervals, only present if `boot` was larger than 0.

binary a named logical vector marking which variables were modelled as binary.

Examples

```
d <- DAG(LS ~ BM, NL ~ BM, DD ~ NL + LS)
plot(d)
d_fitted <- est_DAG(d, rhino, rhino_tree, 'lambda')
plot(d_fitted)
```

phylo_path

Compare causal models in a phylogenetic context.

Description

Continuous variables are modeled using `phylolm::phylolm`, while binary traits are modeled using `phylolm::phylglm`.

Usage

```

phylo_path(
  model_set,
  data,
  tree,
  model = "lambda",
  method = "logistic_MPLE",
  order = NULL,
  parallel = NULL,
  na.rm = TRUE,
  ...
)

```

Arguments

<code>model_set</code>	A list of directed acyclic graphs. These are matrices, typically created with <code>define_model_set</code> .
<code>data</code>	A <code>data.frame</code> with data. If you have binary variables, make sure they are either character values or factors!
<code>tree</code>	A phylogenetic tree of class <code>phylo</code> .
<code>model</code>	The evolutionary model used for the regressions on continuous variables. See phylolm::phylolm for options and details. Defaults to Pagel's lambda model
<code>method</code>	The estimation method for the binary models. See phylolm::phyloglm for options and details. Defaults to logistic MPLE.
<code>order</code>	Causal order of the included variable, given as a character vector. This is used to determine which variable should be the dependent in the dsep regression equations. If left unspecified, the order will be automatically determined. If the combination of all included models is itself a DAG, then the ordering of that full model is used. Otherwise, the most common ordering between each pair of variables is used to create a general ordering.
<code>parallel</code>	<i>Superseded</i> From v1.2 <code>phylopath</code> uses the <code>future</code> package for all parallel processing, see details.
<code>na.rm</code>	Should rows that contain missing values be dropped from the data as necessary (with a message)?
<code>...</code>	Arguments passed on to <code>phylolm</code> : <code>lower.bound</code> : optional lower bound for the optimization of the phylogenetic model parameter. <code>upper.bound</code> : optional upper bound for the optimization of the phylogenetic model parameter. <code>starting.value</code> : optional starting value for the optimization of the phylogenetic model parameter. <code>measurement_error</code> : a logical value indicating whether there is measurement error <code>sigma2_error</code> (see Details). Arguments passed on to <code>phyloglm</code> : <code>btol</code> : bound on the linear predictor to bound the searching space.

log.alpha.bound: bound for the log of the parameter alpha.
 start.beta: starting values for beta coefficients.
 start.alpha: starting values for alpha (phylogenetic correlation).

Details

Parallel processing: From v1.2, phylopath uses the future framework for parallel processing. This is compatible with the parallel computation within the underlying phylolm, making it easy to enable parallel processing of multiple models, and of bootstrap replicates. To enable, simply set a parallel plan() using the future package. Typically, you'll want to run future::plan("multisession", workers = n), where n is the number of cores. Now parallel processing is enabled. Return to sequential processing using future::plan("sequential")

Value

A phylopath object, with the following components:

d_sep for each model a table with separation statements and statistics.

model_set the DAGs

data the supplied data

tree the supplied tree

model the employed model of evolution in phylolm

method the employed method in phylolm

dots any additional arguments given, these are passed on to downstream functions

warnings any warnings generated by the models

Examples

```
#see vignette('intro_to_phylopath') for more details
candidates <- define_model_set(
  A = NL ~ BM,
  B = NL ~ LS,
  .common = c(LS ~ BM, DD ~ NL)
)
p <- phylo_path(candidates, rhino, rhino_tree)

# Printing p gives some general information:
p
# And the summary gives statistics to compare the models:
summary(p)
```

plot.DAG	<i>Plot a directed acyclic graph.</i>
----------	---------------------------------------

Description

Plot a directed acyclic graph.

Usage

```
## S3 method for class 'DAG'
plot(
  x,
  labels = NULL,
  algorithm = "sugiyama",
  manual_layout = NULL,
  text_size = 6,
  box_x = 12,
  box_y = 8,
  edge_width = 1.5,
  curvature = 0.02,
  rotation = 0,
  flip_x = FALSE,
  flip_y = FALSE,
  arrow = grid::arrow(type = "closed", 18, grid::unit(15, "points")),
  ...
)
```

Arguments

x	A ‘DAG’ object, usually created with the define_model_set() or DAG() function.
labels	An optional set of labels to use for the nodes. This should be a named vector, of the form <code>c(var1 = "label1", var2 = "label2")</code> . If left at ‘NULL’, the variable names of the DAGs are used.
algorithm	A layout algorithm from <code>igraph</code> , see ggraph::create_layout() . By default, uses the Sugiyama layout algorithm, which is designed to minimize edge crossing in DAGs.
manual_layout	Alternatively, precisely define the layout yourself, by providing a <code>data.frame</code> that at least has a column name with all variable names, and columns <code>x</code> and <code>y</code> with positions to be plotted. Setting this parameter overrides <code>algorithm</code> but other changes, such as rotation and flips will still be applied.
text_size	Size of the node label text.
box_x	To avoid the arrows colliding with the nodes, specify the rectangular dimensions of an invisible box around each node. If you have long labels, you need to increase this.

box_y	To avoid the arrows colliding with the nodes, specify the rectangular dimensions of an invisible box around each node. If you have multi-line labels, you need to increase this.
edge_width	Width of the edges.
curvature	Curvature of the edges. A slight curvature can look pretty.
rotation	Supply the degrees you want to rotate the layout by. This is useful in order to put rotate your upstream nodes towards the top if needed.
flip_x	Whether to flip the node positions horizontally.
flip_y	Whether to flip the node positions vertically.
arrow	A <code>grid::arrow</code> object, specifying the shape and size of the arrowheads. The order of facets is taken from the ordering of the list, with the facet labels coming from the names of the list. If the list is unnamed, sequential lettering is used.
...	Not used.

Examples

```
d <- DAG(a ~ b + c + d)
plot(d)

# Plot with manually defined positions:
ml <- data.frame(
  name = c('a', 'b', 'c', 'd'),
  x = c(1, 1, 2, 2),
  y = c(1, 2, 1, 2)
)
plot(d, manual_layout = ml)
```

plot.fitted_DAG

Plot a directed acyclic graph with path coefficients.

Description

Plot a directed acyclic graph with path coefficients.

Usage

```
## S3 method for class 'fitted_DAG'
plot(
  x,
  type = "width",
  labels = NULL,
  algorithm = "sugiyama",
  manual_layout = NULL,
  text_size = 6,
```

```

    box_x = 12,
    box_y = 8,
    edge_width = 1.25,
    curvature = 0.02,
    rotation = 0,
    flip_x = FALSE,
    flip_y = FALSE,
    arrow = grid::arrow(type = "closed", 18, grid::unit(15, "points")),
    colors = c("firebrick", "navy"),
    show.legend = TRUE,
    width_const = NULL,
    ...
)

```

Arguments

x	An object of class fitted_DAG.
type	How to express the weight of the path. Either "width", or "color".
labels	An optional set of labels to use for the nodes. This should be a named vector, of the form <code>c(var1 = "label1", var2 = "label2")</code> . If left at 'NULL', the variable names of the DAGs are used.
algorithm	A layout algorithm from igraph, see ggraph::create_layout() . By default, uses the Sugiyama layout algorithm, which is designed to minimize edge crossing in DAGs.
manual_layout	Alternatively, precisely define the layout yourself, by providing a data.frame that at least has a column name with all variable names, and columns x and y with positions to be plotted. Setting this parameter overrides algorithm but other changes, such as rotation and flips will still be applied.
text_size	Size of the node label text.
box_x	To avoid the arrows colliding with the nodes, specify the rectangular dimensions of an invisible box around each node. If you have long labels, you need to increase this.
box_y	To avoid the arrows colliding with the nodes, specify the rectangular dimensions of an invisible box around each node. If you have multi-line labels, you need to increase this.
edge_width	Width of the edges.
curvature	Curvature of the edges. A slight curvature can look pretty.
rotation	Supply the degrees you want to rotate the layout by. This is useful in order to put rotate your upstream nodes towards the top if needed.
flip_x	Whether to flip the node positions horizontally.
flip_y	Whether to flip the node positions vertically.
arrow	A <code>grid::arrow</code> object, specifying the shape and size of the arrowheads. The order of facets is taken from the ordering of the list, with the facet labels coming from the names of the list. If the list is unnamed, sequential lettering is used.

colors	The end points of the continuous color scale. Keep in mind that red and green are obvious colors to use, but are better to be avoided because of color blind users.
show.legend	Whether a legend for the color scale should be shown.
width_const	Deprecated.
...	Not used.

Examples

```
d <- DAG(LS ~ BM, NL ~ BM, DD ~ NL + LS)
d_fitted <- est_DAG(d, rhino, rhino_tree, 'lambda')
plot(d_fitted)
```

```
plot.phylopath_summary
```

Plot the comparison of a set of causal models.

Description

Shows the relative support for each causal model as a bar chart of model weights, ordered from best to worst. Models within cut_off CICc of the best model are highlighted, since those are usually considered to have comparable support. Each bar is labelled with the p-value of the d-separation test, where a significant value means the model is rejected by the data.

Usage

```
## S3 method for class 'phylopath_summary'
plot(x, cut_off = 2, ...)
```

Arguments

x	A phylopath_summary object, obtained by calling summary() on the result of phylo_path() .
cut_off	The CICc difference within which models are highlighted as having support comparable to the best model.
...	Not used.

Value

A ggplot object.

Examples

```

candidates <- define_model_set(
  A = NL ~ BM,
  B = NL ~ LS,
  .common = c(LS ~ BM, DD ~ NL)
)
p <- phylo_path(candidates, rhino, rhino_tree)
plot(summary(p))

```

plot_model_set	<i>Plot several causal hypothesis at once.</i>
----------------	--

Description

Plot several causal hypothesis at once.

Usage

```

plot_model_set(
  model_set,
  labels = NULL,
  algorithm = "kk",
  manual_layout = NULL,
  text_size = 5,
  box_x = 12,
  box_y = 10,
  edge_width = 1,
  curvature = 0.05,
  rotation = 0,
  flip_x = FALSE,
  flip_y = FALSE,
  nrow = NULL,
  arrow = grid::arrow(type = "closed", 15, grid::unit(10, "points"))
)

```

Arguments

model_set	A list of DAG objects, usually created with define_model_set() .
labels	An optional set of labels to use for the nodes. This should be a named vector, of the form <code>c(var1 = "label1", var2 = "label2")</code> . If left at 'NULL', the variable names of the DAGs are used.
algorithm	A layout algorithm from igraph, see ggraph::create_layout() . By default, uses the Kamada-Kawai layout algorithm. Another good option is "sugiyama", which is designed to minimize edge crossing in DAGs. However, it can often plot nodes too close together.

manual_layout	Alternatively, precisely define the layout yourself, by providing a <code>data.frame</code> that at least has a column name with all variable names, and columns <code>x</code> and <code>y</code> with positions to be plotted. Setting this parameter overrides <code>algorithm</code> but other changes, such as rotation and flips will still be applied.
text_size	Size of the node label text.
box_x	To avoid the arrows colliding with the nodes, specify the rectangular dimensions of an invisible box around each node. If you have long labels, you need to increase this.
box_y	To avoid the arrows colliding with the nodes, specify the rectangular dimensions of an invisible box around each node. If you have multi-line labels, you need to increase this.
edge_width	Width of the edges.
curvature	Curvature of the edges. A slight curvature can look pretty.
rotation	Supply the degrees you want to rotate the layout by. This is useful in order to put rotate your upstream nodes towards the top if needed.
flip_x	Whether to flip the node positions horizontally.
flip_y	Whether to flip the node positions vertically.
nrow	Number of rows to display the models on.
arrow	A <code>grid::arrow</code> object, specifying the shape and size of the arrowheads. The order of facets is taken from the ordering of the list, with the facet labels coming from the names of the list. If the list is unnamed, sequential lettering is used.

Value

A `ggplot` object.

Examples

```
m <- list(one = DAG(a ~ b + c + d), two = DAG(a ~ b, b ~ c, d ~ d))
plot_model_set(m)
plot_model_set(m, algorithm = "sugiyama")
```

red_list

Data on brain size, life history and vulnerability to extinction

Description

A dataset with continuous variables affecting the conservation Status of mammalian species (the IUCN red list of threatened species).

Usage

```
red_list
```

Format

An object of class `data.frame` with 474 rows and 7 columns.

Details

It includes the following variables: brain size (Br), body size (B), gestation period (G), litter size (L), weaning age (W), population density (P) and vulnerability to extinction (Status).

Source

Gonzalez-Voyer A, Gonzalez-Suarez M, Vila C, Revilla E (2016) Larger brain size indirectly increases vulnerability to extinction in mammals. *Evolution* 70:1364-1375. doi: 10.1111/evo.12943

red_list_tree	<i>Mammalian phylogeny</i>
---------------	----------------------------

Description

This is the accompanying phylogeny for the red_list data set. It is based on the updated mammalian supertree by Bininda-Emonds et al. 2007 & Fritz et al. 2009.

Usage

```
red_list_tree
```

Format

An object of class `phylo` of length 4.

Source

Gonzalez-Voyer, A. Gonzalez-Suarez M. Vila C. and Revilla E. 2016. Larger brain size indirectly increases vulnerability to extinction in mammals. *Evolution* 70:1364-1375. doi: 10.1111/evo.12943.

Bininda-Emonds, O. R. P., M. Cardillo, K. E. Jones, R. D. E. MacPhee, R. M. D. Beck, R. Grenyer, S. A. Price, R. A. Vos, J. L. Gittleman, and A. Purvis. 2007. The delayed rise of present-day mammals. *Nature* 446:507-512.

Fritz, S. A., O. R. P. Bininda-Emonds, and A. Purvis. 2009. Geographical variation in predictors of mammalian extinction risk: big is bad, but only in the tropics. *Ecol. Lett.* 12:538-549.

rhino	<i>Rhinogrades traits.</i>
-------	----------------------------

Description

A simulated dataset, as used by Gonzalez-Voyer and Von Hardenberg as an example, containing variables on body mass (BM), litter size (LS), nose length (NL), dispersal distance (DD) and range size (RS).

Usage

rhino

Format

An object of class `data.frame` with 100 rows and 6 columns.

Source

Gonzalez-Voyer A & von Hardenberg A. 2014. An Introduction to Phylogenetic Path Analysis. Chapter 8. In: Garamszegi LZ (ed.), Modern Phylogenetic Comparative Methods and Their Application in Evolutionary Biology. pp. 201-229. Springer-Verlag Berlin Heidelberg. doi:10.1111/j.1558-5646.2012.01790.x

rhino_tree	<i>Rhinogrades phylogeny.</i>
------------	-------------------------------

Description

A phylogenetic tree for the 100 species of the rhino dataset.

Usage

rhino_tree

Format

An object of class `phylo` of length 4.

Source

Gonzalez-Voyer A & von Hardenberg A. 2014. An Introduction to Phylogenetic Path Analysis. Chapter 8. In: Garamszegi LZ (ed.), Modern Phylogenetic Comparative Methods and Their Application in Evolutionary Biology. pp. 201-229. Springer-Verlag Berlin Heidelberg. doi:10.1111/j.1558-5646.2012.01790.x

`show_warnings`*Print out warnings from a phylopath analysis.*

Description

Use this function after running `phylo_path()` to conveniently print any generated warnings to the screen. You can either provide no arguments, which will only work if you run it directly after the analysis, or you have to provide the phylopath object manually.

Usage

```
show_warnings(phylopath = NULL)
```

Arguments

`phylopath` A phylopath object of which the warnings should be printed.

Index

- * **datasets**
 - cichlids, [8](#)
 - cichlids_tree, [8](#)
 - red_list, [22](#)
 - red_list_tree, [23](#)
 - rhino, [24](#)
 - rhino_tree, [24](#)
- as.data.frame.fitted_DAG, [2](#)
- as.data.frame.fitted_DAG(), [9](#)
- average, [3](#)
- average(), [3](#), [9](#), [10](#)
- average_DAGs, [4](#)
- best, [6](#)
- best(), [3](#), [10](#)
- choice, [7](#)
- choice(), [3](#)
- cichlids, [8](#)
- cichlids_tree, [8](#)
- coef.fitted_DAG, [9](#)
- coef_plot, [10](#)
- confint.fitted_DAG (coef.fitted_DAG), [9](#)
- DAG, [11](#)
- DAG(), [17](#)
- define_model_set, [12](#)
- define_model_set(), [11](#), [17](#), [21](#)
- est_DAG, [13](#)
- est_DAG(), [3–7](#), [10](#)
- ggm::DAG(), [12](#)
- ggplot2::coord_flip(), [10](#)
- ggraph::create_layout(), [17](#), [19](#), [21](#)
- phylo_path, [3](#), [6](#), [7](#), [14](#)
- phylo_path(), [20](#)
- phylolm::phylolm, [3](#), [6](#), [7](#), [13–15](#)
- phylolm::phylolm, [3](#), [6](#), [7](#), [13–15](#)
- plot.DAG, [17](#)
- plot.fitted_DAG, [18](#)
- plot.phylopath_summary, [20](#)
- plot_model_set, [21](#)
- red_list, [22](#)
- red_list_tree, [23](#)
- rhino, [24](#)
- rhino_tree, [24](#)
- show_warnings, [25](#)