Package 'multimedia'

September 18, 2024

Title Multimodal Mediation Analysis

Version 0.2.0

Description Multimodal mediation analysis is an emerging problem in microbiome data analysis. Multimedia make advanced mediation analysis techniques easy to use, ensuring that all statistical components are transparent and adaptable to specific problem contexts. The package provides a uniform interface to direct and indirect effect estimation, synthetic null hypothesis testing, bootstrap confidence interval construction, and sensitivity analysis. More details are available in Jiang et al. (2024) ``multimedia: Multimodal Mediation Analysis of Microbiome Data'' <doi:10.1101/2024.03.27.587024>.

License CC0

Encoding UTF-8

RoxygenNote 7.3.2

VignetteBuilder knitr

biocViews Coverage, Microbiome, Regression, Sequencing, Software, StatisticalMethod, StructuralEquationModels

Depends R (>= 4.1.0), brms, glmnetUtils, ranger, tidyselect

Imports MASS, S4Vectors, SummarizedExperiment, cli, dplyr, fansi, formula.tools, ggplot2, glue, methods, miniLNM, patchwork, phyloseq, progress, purrr, rlang, stats, tidygraph, tidyr

Suggests compositions, ggdist, ggraph, ggrepel, knitr, rmarkdown, testthat (>= 3.0.0), tidyverse, vroom

BugReports https://github.com/krisrs1128/multimedia/issues/

URL https://krisrs1128.github.io/multimedia/, https://github.com/krisrs1128/multimedia/

LazyData false

Config/testthat/edition 3

NeedsCompilation no

Author Kris Sankaran [aut, cre] (<https://orcid.org/0000-0002-9415-1971>), Hanying Jiang [aut]

Maintainer Kris Sankaran <ksankaran@wisc.edu>

Repository CRAN

Date/Publication 2024-09-18 12:30:05 UTC

Contents

`ansi_aware_handler` *Pretty Printing*

Description

Helper function for printing ANSI in Rmarkdown output. Use this at the start of your Rmarkdown files to include colors in the printed object names in the final compiled output.

Usage

```
ansi_aware_handler(x, options)
```

Arguments

x	A character vector potentially including ANSI.
options	Unused placeholder argument.

Details

Taken from the post at

https://blog.djnavarro.net/posts/2021-04-18_pretty-little-clis/

Value

A string with HTML reformatted to ensure colors appear in printed code blocks in rmarkdown output.

Examples

```
knitr::knit_hooks$set(output = ansi_aware_handler)
options(crayon.enabled = TRUE)
```

---

bind_mediation *Convert* mediation_data *to a single data.frame*

Description

It can be helpful to combine all the data in a mediation_data S4 object into a single data.frame. This concatenates all data sources horizontally, into an N samples x (all outcomes, mediators, treatments, ...) matrix.

Usage

```
bind_mediation(exper)
```

Arguments

exper	An object of class mediation_data containing the variables that we want horizontally concatenated.

Value

A data.frame containing all variables from the original mediation_data object.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
exper
bind_mediation(exper)

exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
exper
bind_mediation(exper)
```

`bootstrap` *Bootstrap Distribution for Estimators*

Description

Given a mediation model specification, estimators fs, and original dataset exper, this will re-estimate the mediation model on resampled versions of exper and apply each estimator in fs to construct bootstrap distributions associated wtih those estimators.

Usage

```
bootstrap(model, exper, fs = NULL, B = 1000, progress = TRUE)
```

Arguments

`model`	An object of class multimedia with specified mediation and outcome models that we want to re-estimate across B bootstrap samples.
`exper`	An object of class multimedia_data containing the mediation and outcome data from which the direct effects are to be estimated.
`fs`	The estimators whose bootstrap samples we are interested in. These are assumed to be a vector of functions (for example, direct_effect or indirect_effect), and they will each be applied to each bootstrap resample.
`B`	The number of bootstrap samples. Defaults to 1000.
`progress`	A logical indicating whether to show a progress bar.

Value

stats A list of length B containing the results of the fs applied on each of the B bootstrap resamples.

Examples

```
# example with null data. We set B to 5 just to execute quickly -- it's not
# actually a practical choice of B
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
```

```
    bootstrap(exper, B = 5)

# example with another dataset
exper <- demo_spline(n_samples = 100, tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
samples <- multimedia(exper, rf_model(num.trees = 1e3)) |>
    bootstrap(exper, B = 5)
ggplot2::ggplot(samples$direct_effect) +
    ggplot2::geom_histogram(
        ggplot2::aes(direct_effect, fill = indirect_setting),
        bins = 15
    ) +
    ggplot2::facet_wrap(~outcome, scales = "free")
```

`brms_model` *Bayesian Regression Model across Responses*

Description

Apply a Bayesian regression model in parallel across each response y in an outcome or mediation model. This can be helpful when we want to share information across related

Usage

```
brms_model(...)
```

Arguments

`...`	Keyword parameters passed to brm.

Value

model An object of class `model` with estimator, predictor, and sampler functions associated wtih a Bayesian regression model.

See Also

glmnet_model lnm_model rf_model lm_model

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper, brms_model()) # call estimate() on this to fit
```

brms_sampler *Sample from a Bayesian Regression Model*

Description

This samples from the posterior predictive for each component in a multiresponse Bayesian Regression model.

Usage

```
brms_sampler(fits, newdata = NULL, indices = NULL, ...)
```

Arguments

fits	The fitted 'BRMS' model model from which to draw samples.
newdata	A data.frame containing new inputs from which to sample responses. If NULL, defaults to the data used to estimate fit.
indices	The coordinates of the response from which we want to sample.
...	Additional arguments to pass to posterior_predict in the 'brms' package.

Value

A data.frame containing a single posterior predictive sample at each of the newdata rows passed into a fitted BRMS model. Each column corresponds to one outcome variable, each row to the associated row in the newdata input..

contrast_predictions *Estimate the Difference between Profiles*

Description

Given a fitted multimedia model, contrast the mediation and outcome predictions associated wtih two treatment profiles.

Usage

```
contrast_predictions(model, profile1, profile2, ...)
```

Arguments

model	An object of class multimedia containing the estimated mediation and outcome models whose mediation and outcome predictions we want to compare.
profile1	An object of class treatment_profile containing the first treatment profile to consider in the difference.
profile2	An object of class treatment_profile containing the second treatment profile to consider in the difference.
...	Additional arguments to pass to predict().

Value

A list with two elements, `mediators` and `outcomes`, containing the differences in the predicted M(T') - M(T) and Y(T', M(T')) - Y(T, M(T)) between the two profiles T and T'.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
model <- multimedia(exper) |>
    estimate(exper)
t1 <- data.frame(treatment = factor("Treatment"))
t2 <- data.frame(treatment = factor("Control"))
profile1 <- setup_profile(model, t1, t1)
profile2 <- setup_profile(model, t2, t2)
contrast_predictions(model, profile1, profile2)
```

`contrast_samples` *Difference between Samples at Contrasting Profiles*

Description

Given a fitted multimedia model, contrast sampled mediation and outcome data associated with two treatment profiles.

Usage

```
contrast_samples(model, profile1, profile2, ...)
```

Arguments

`model`	An object of class multimedia containing the estimated mediation and outcome models whose mediation and outcome predictions we want to compare.
`profile1`	An object of class `treatment_profile` containing the first treatment profile to consider in the difference.
`profile2`	An object of class `treatment_profile` containing the second treatment profile to consider in the difference.
`...`	Additional arguments to pass to `sample()`.

Value

A list with two elements, `mediators` and `outcomes`, containing the differences in the sampled M(T') - M(T) and Y(T', M(T')) - Y(T, M(T)) between the two profiles T and T'.

A list with two elements, mediators and outcomes. These contrast the values of the mediator and outcomes under the two profiles T and T'.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
model <- multimedia(exper) |>
    estimate(exper)
t1 <- data.frame(treatment = factor("Treatment"))
t2 <- data.frame(treatment = factor("Control"))
profile1 <- setup_profile(model, t1, t1)
profile2 <- setup_profile(model, t2, t2)
contrast_samples(model, profile1, profile2)

samples <- purrr::map(
    seq_len(100),
    ~ contrast_samples(model, profile1, profile2)
)
hist(sapply(samples, \(x) x[[1]]$ASV1))
hist(sapply(samples, \(x) x[[1]]$ASV2))
```

---

demo_joy *A Demo Dataset (Random)*

Description

This is a simple dataset with random data, used simply to illustrate the design of multimedia. There is no real association between any treatments, mediators, or outcomes. It always returns a single outcome (PHQ) SummarizedExperiment, and it randomly assigns samples to Treatment and Control (see the colData). It stores all the hypothetical mediator data in the assay slot.

Usage

```
demo_joy(n_samples = 100, n_mediators = 5, n_pretreatment = 3)
```

Arguments

`n_samples`	How many random samples to generate?
`n_mediators`	How many random mediators to generate?
`n_pretreatment`	How many random pretreatment variables?

Value

SE The summarized experiment containing random data.

Examples

```
demo_joy()

demo_joy(n_samples = 2, n_mediators = 20)
```

demo_spline *A Demo Dataset (Spline)*

Description

This is a simple dataset with nonlinear relationships between the outcome and mediators. It is used simply to illustrate the design of multimedia. The mediator->outcome effect is generated from a random spline function.

Usage

```
demo_spline(n_samples = 5000, tau = c(2, 2))
```

Arguments

n_samples	The number of samples to generate in the toy example
tau	The true direct effects associated with the two outcomes. Defaults to 2, 2.

Value

xy A data.frame whose columns include the treatment, mediation, and outcome variables.

Examples

```
demo_spline()
```

direct_effect *Direct Effects from Estimated Model*

Description

Estimate direct effects associated with a multimedia model. These estimates are formed using Equation (10) of our paper. Rather than providing this average, this function returns the estimated difference for each j. To average across all j, this result can be passed to the ' `effect_summary` function.

Usage

```
direct_effect(model, exper = NULL, t1 = 1, t2 = 2)
```

Arguments

model	An object of class multimedia containing the estimated mediation and outcome models whose mediation and outcome predictions we want to compare.
exper	An object of class multimedia_data containing the mediation and outcome data from which the direct effects are to be estimated.
t1	The reference level of the treatment to be used when computing the direct effect.
t2	The alternative level of the treatment to be used when computing the direct effect.

Value

A data.frame summarizing the direct effects associated with different settings of j in the equation above.

See Also

effect_summary

Examples

```
# example with null data
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
fit <- multimedia(exper) |>
    estimate(exper)

direct_effect(fit)
direct_effect(fit, t1 = 2, t2 = 1)
direct_effect(fit, t1 = 2, t2 = 2)

# example with another dataset
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
fit <- multimedia(exper) |>
    estimate(exper)
direct_effect(fit)
```

---

edges *Graphical Structure for Mediation Objects*

Description

We often want to access the DAG for different mediation-related S4 objects. This generic helps us access these graphical model edges lists. See method instantiations for specific examples.

Usage

```
edges(object)
```

Arguments

object	An object whose DAG structure we want to access.

Value

The output depends on the S4 object that is passed. For multimedia objects, this will return an edgelist as a two column data.frame.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    edges()
```

---

edges,multimedia-method

Access Mediation Model DAG

Description

This is an accessor to the edges slot in a multimedia object. It is the internal representation of the variable conditional dependence graph encoded by the mediation model's DAG.

Usage

```
## S4 method for signature 'multimedia'
edges(object)
```

Arguments

object	An object of class multimedia.

Value

A data.frame whose rows give edges in the mediation analysis DAG.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    edges()
```

`effect_summary` *Average Effects across j*

Description

This averages direct or indirect effects across settings j, leading to the effect estimates given in equation (10) of the preprint.

Usage

```
effect_summary(effects)
```

Arguments

`effects`	The output from direct_effect or indirect_effect. A data.frame containing effect estimates for each variable and indirect/direct setting along rows.

Value

A version of the input with all indirect/direct settings averaged over.

See Also

direct_effect indirect_effect

Examples

```
# example with null data
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    estimate(exper) |>
    direct_effect() |>
    effect_summary()

# example with another dataset
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
multimedia(exper) |>
    estimate(exper) |>
    direct_effect() |>
    effect_summary()
```

`estimate` *Estimate a Mediation Model*

Description

`estimate` provides a unified interface to estimate all the models that can be encapsulated within a `multimedia` class. It simply calls the multimedia object. The resulting estimates can be used for downstream direct effect estimation.

Usage

```
estimate(model, exper)
```

Arguments

`model`	An object of class multimedia containing the estimated mediation and outcome models whose mediation and outcome predictions we want to compare.
`exper`	An object of class multimedia_data containing the mediation and outcome data from which the direct effects are to be estimated.

Value

A version of the input modified in place so that the @estimates slot has been filled.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    estimate(exper)

# example with another dataset
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
multimedia(exper) |>
    estimate(exper)

# example with another model
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
multimedia(exper, glmnet_model()) |>
    estimate(exper)
```

estimator *Accessor for Model Estimators*

Description

This defines a generic estimator function, which can be applied to different multimedia model objects. It creates a unified interface to estimating diverse mediation and outcome model families.

Usage

```
estimator(object)

estimator(object)
```

Arguments

object	A model object that we want to estimate.

Value

A function that can be called with formula and data arguments, like

A fitted version of the input model class.

Examples

```
m <- lm_model()
estimator(m)(mpg ~ hp + wt, data = mtcars)
m <- rf_model()
fit <- estimator(m)(mpg ~ hp + wt, data = mtcars)
```

estimator,model-method *Accessor for Estimators*

Description

Accessor for Estimators

Usage

```
## S4 method for signature 'model'
estimator(object)
```

Arguments

object	An object of class model, whose estimator function we want access to.

Value

A fitted version of the input model class.

Examples

```
m <- rf_model()
fit <- estimator(m)(mpg ~ hp + wt, data = mtcars)
```

---

exper_df *Convert a Summarized Experiment to a data.frame*

Description

This is a small helepr function to concatenate the colData and assays within SummarizedExperiment objects into a single data.frame. This is not necessary for the essential multimedia workflow, it is only exported for potential independent interest.

Usage

```
exper_df(exper)
```

Arguments

exper	An object of class SummarizedExperiment.

Value

A data.frame combining all slots of a multimedia object.

Examples

```
demo_joy() |>
    exper_df()
```

---

fdr_summary *Calibration using Synthetic Nulls*

Description

This function computes a threshold for indirect or direct effect estimates that controls the false discovery rate according to estimates made using real and synthetic null data, against the null hypotheses that effects are zero. It computes the proportion of synthetic null estimates that are among the top K largest effects (in magnitude) as an estimate of the FDR.

Usage

```
fdr_summary(contrast, effect = "indirect_overall", q_value = 0.15)
```

Arguments

contrast	A data.frame summarizing the differences between outcomes across hypothetical treatments, typically as output by null_contrast. Each row is one outcome in one hypothetical scenario.
effect	Either "indirect_overall" (the default), "indirect_pathwise", or "direct_effect" specifying the type of effect that we want to select.
q_value	The target for false discovery rate control. The last time the estimated FDR is above this threshold is smallest magnitude of effect size that we will consider.

Value

fdr A data.frame specifying, for each candidate effect, whether it should be selected.

Examples

```
# example with null data - notice synthetic data has larger effect.
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    estimate(exper) |>
    null_contrast(exper) |>
    fdr_summary("direct_effect")

multimedia(exper) |>
    estimate(exper) |>
    null_contrast(exper, "M->Y", indirect_overall) |>
    fdr_summary("indirect_overall")

# example with another dataset - synthetic effect is smaller.
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
multimedia(exper) |>
    estimate(exper) |>
    null_contrast(exper) |>
    fdr_summary("direct_effect")

multimedia(exper) |>
    estimate(exper) |>
    null_contrast(exper, "M->Y", indirect_overall) |>
    fdr_summary("indirect_overall")
```

---

glmnet_model *Regularized 'Glmnet' Model across Responses*

Description

Apply a regularized (generalized) linear model in parallel across each response y in an outcome or mediation model. This can be helpful when we have many mediators or pretreatment variables, making the input high-dimensional.

Usage

```
glmnet_model(progress = TRUE, ...)
```

Arguments

progress	A logical indicating whether to show a progress bar during estimation.
...	Keyword parameters passed to package 'glmnet'.

Value

model An object of class `model` with estimator, predictor, and sampler functions associated wtih a lienar model.

See Also

model lm_model rf_model

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper, glmnet_model(lambda = 1)) |>
    estimate(exper)

multimedia(exper, glmnet_model(lambda = 1), glmnet_model()) |>
    estimate(exper)

# example with another dataset
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
multimedia(exper, glmnet_model(lambda = 0.1)) |>
    estimate(exper)
```

---

glmnet_sampler *Sample from a 'Glmnet' Package Model*

Description

This assumes a continuous response, so that the out-of-sample MSE can be used to estimate the outcome variability sigma.

Usage

```
glmnet_sampler(fits, newdata = NULL, indices = NULL, lambda_ix = 1, ...)
```

Arguments

fits	The fitted 'glmnet' package model model from which to draw samples.
newdata	A data.frame containing new inputs from which to sample responses. If NULL, defaults to the data used to estimate fit.
indices	The coordinates of the response from which to draw samples.
lambda_ix	A regularization strength parameter used to maintain consistency with estimation. Not used during sampling.
...	Additional parameters to pass to predict.glmnet

Value

y_star A data.frame of samples y associated with the new inputs.

Examples

```
m <- glmnet_model()
fit <- estimator(m)(mpg ~ hp + wt, data = mtcars)
glmnet_sampler(fit, mtcars)

plm <- parallelize(glmnetUtils::glmnet)
fit <- plm(mpg + disp ~ hp + wt, data = mtcars)
glmnet_sampler(fit, mtcars)
```

---

indirect_overall *Overall Indirect Effect*

Description

Direct Effects from Estimated Model

Usage

```
indirect_overall(model, exper = NULL, t1 = 1, t2 = 2)
```

Arguments

model	An object of class multimedia containing the estimated mediation and outcome models whose mediation and outcome predictions we want to compare.
exper	An object of class multimedia_data containing the mediation and outcome data from which the direct effects are to be estimated.
t1	The reference level of the treatment to be used when computing the indirect effect.
t2	The alternative level of the treatment to be used when computing the indirect effect.

Details

Estimate direct effects associated with a multimedia model. These estimates are formed using Equation (10) of our preprint.

Value

A data.frame summarizing the overall indirect effects associated with different settings of j in the equation above.

Examples

```
# example with null data
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
fit <- multimedia(exper) |>
    estimate(exper)

indirect_overall(fit)

# example with another dataset
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
fit <- multimedia(exper) |>
    estimate(exper)
indirect_overall(fit)
```

---

indirect_pathwise *Indirect Effects via Single Mediation Paths*

Description

Indirect Effects via Single Mediation Paths

Usage

```
indirect_pathwise(model, exper = NULL, t1 = 1, t2 = 2, progress = TRUE)
```

Arguments

model	An object of class multimedia containing the estimated mediation and outcome models whose mediation and outcome predictions we want to compare.
exper	An object of class multimedia_data containing the mediation and outcome data from which the direct effects are to be estimated.
t1	The reference level of the treatment to be used when computing the (pathwise) indirect effect.
t2	The alternative level of the treatment to be used when computing the (pathwise) indirect effect.
progress	A logical indicating whether to show a progress bar.

Value

A data.frame summarizing the pathwise (per-mediator) indirect effects associated with different settings of the direct effect.

Examples

```
# example with null data
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
fit <- multimedia(exper) |>
    estimate(exper)
indirect_pathwise(fit)

# example with another dataset
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
fit <- multimedia(exper) |>
    estimate(exper)
indirect_pathwise(fit)
```

---

lm_model *Linear Model across Responses*

Description

Apply a linear model in parallel across each response y in an outcome or mediation model. This is often useful for mediator models with few pretreatment variables, since each input is low-dimensional, even when there are many responses.

Usage

```
lm_model(progress = TRUE)
```

Arguments

progress	A logical indicating whether to show a progress bar during estimation.

Value

model An object of class `model` with estimator, predictor, and sampler functions associated wtih a linear model.

See Also

model

Examples

```
m <- lm_model()
estimator(m)(mpg ~ hp + wt, data = mtcars)
```

lm_sampler *Sample a Linear Model*

Description

Draw samples from a fitted linear model.

Usage

```
lm_sampler(fits, newdata = NULL, indices = NULL, ...)
```

Arguments

fits	The fitted linear model model from which to draw samples.
newdata	A data.frame containing new inputs from which to sample responses. If NULL, defaults to the data used to estimate fit.
indices	The coordinates of the response from which to draw samples.
...	Additional parameters passed to lm.predict

Value

y_star A data.frame of samples y associated with the new inputs.

Examples

```
fit <- lm(mpg ~ hp + wt, data = mtcars)
lm_sampler(list(f = fit))

plm <- parallelize(lm)
fit <- plm(mpg + disp ~ hp + wt, data = mtcars)
lm_sampler(fit)
```

lnm_model *Logistic Normal Multinomial Model*

Description

Apply a logistic normal multinomial model to jointly model a vector of count responses y in an outcome or mediation model. This is a common choice for data where the parameter of interest is the composition across responses (e.g., microbiome).

Usage

```
lnm_model(...)
```

Arguments

...	Keyword parameters passed to lnm in the 'miniLNM' package.

Value

model An object of class `model` with estimator, predictor, and sampler functions associated wtih a linear model.

See Also

model lm_model rf_model glmnet_model brms_model

Examples

```
m <- lnm_model()
mat <- data.frame(matrix(rpois(250, 10), 25, 10))
colnames(mat) <- paste0("y", seq_len(6))
fit <- estimator(m)(y1 + y2 + y3 + y4 ~ y5 + y6, mat)
```

`lnm_sampler` *Sample from the Logistic Normal Multinomial*

Description

This samples from the posterior predictive of a fitted logistic-normal multinomial model.

Usage

```
lnm_sampler(fit, newdata = NULL, indices = NULL, ...)
```

Arguments

`fit`	The fitted LNM model from which to draw posterior predictive samples.
`newdata`	A data.frame containing new inputs from which to sample responses. If NULL, defaults to the data used to estimate fit.
`indices`	The coordinates of the response from which to draw samples.
`...`	Additional parameters passed to sample.

Value

y_star A data.frame of samples y associated wtih the new inputs.

Examples

```
m <- lnm_model()
mat <- data.frame(matrix(rpois(250, 10), 25, 10))
colnames(mat) <- paste0("y", 1:6)
fit <- estimator(m)(y1 + y2 + y3 + y4 ~ y5 + y6, mat)
lnm_sampler(fit, depth = 10)
lnm_sampler(fit, depth = 100)
```

`mediation_data` `mediation_data` *Constructor*

Description

Convert a SummarizedExperiment, phyloseq object, or data.frame into a `mediation_data` object. This conversion helps to organize the variables that lie on different parts of the mediation analysis graph, so that they are not all kept in a homogeneous experiment or data.frame. It's possible to specify outcome, mediator, or treatment variables using either string vectors or tidyselect syntax, e.g., `starts_with("mediator_")` will match all columns of the input data starting with the string mediator.

Usage

```
mediation_data(x, outcomes, treatments, mediators, pretreatments = NULL)
```

Arguments

`x`	A SummarizedExperiment, phyloseq object, or data.frame whose columns contain all the variables needed for the mediation analysis.
`outcomes`	A vector or tidyselect specification of the names of all outcome variables.
`treatments`	A vector or tidyselect specification of the names of all treatment variables.
`mediators`	A vector or tidyselect specification of the names of all mediators.
`pretreatments`	A vector containing names of all pretreatment variables. Defaults to NULL, in which case the pretreatments slot will remain empty.

Value

result An object of class `mediation_data`, with separate slots for each of the node types in a mediation analysis diagram.

See Also

mediation_data-class

Examples

```
# multiple outcomes, one mediator
mediation_data(
    demo_spline(), starts_with("outcome"), "treatment", "mediator"
)

# one outcome, multiple mediators
mediation_data(demo_joy(), "PHQ", "treatment", starts_with("ASV"))
```

`mediation_models` *Accessor for Outcome Models*

Description

Accessor for Outcome Models

Usage

```
mediation_models(object)
```

Arguments

`object`	An object of class multimedia whose outcome model estimates we would like to extract.

Value

A list containing all the fitted mediation models.

Examples

```
data(mindfulness)
exper <- mediation_data(
    mindfulness,
    phyloseq::taxa_names(mindfulness),
    "treatment",
    starts_with("mediator"),
    "subject"
)

m <- multimedia(exper)
mediation_models(m)
```

mediators *Access Class-Specific Mediators*

Description

This is a shorthand for accessing mediator-related slots in classes exported by the multimedia package.

Usage

```
mediators(object)
```

Arguments

object	An object whose mediators we want to access (either their names or values).

Value

The output will depend on the class of the object that is passed in. For multimedia objects, this returns a character vector of mediators. For mediation data objects, this returns the mediator mediator data.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
mediators(exper)
```

mediators,mediation_data-method *Access to @mediators in Mediation Data*

Description

This is an accessor returns the @mediators slot in a mediation_data object.

Usage

```
## S4 method for signature 'mediation_data'
mediators(object)
```

Arguments

object	An object of class mediation_data.

Value

m A data.frame whose rows are samples and columns are values of mediators across those samples.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
mediators(exper)
```

---

mediators,multimedia-method

Names of Mediators in a Multimedia Object

Description

This is a helper that returns the names of the mediator variables in an object of class multimedia. It parses the graph in the DAG specifying the mediation analysis, and it returns all variables between treatment and outcome.

Usage

```
## S4 method for signature 'multimedia'
mediators(object)
```

Arguments

object	An object of class multimedia.

Value

m A vector of strings containing the names of all the mediators.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    mediators()
```

mediators<- *Set Mediators*

Description

This is an setter method for mediators in an S4 object, usually of class mediation_data.

Usage

```
mediators(object) <- value
```

Arguments

object	An object whose mediators slot to modify.
value	The new mediator values to set within object.

Value

Modifies the mediators slot of the input object in place.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
mediators(exper) <- data.frame(new_ASV = rnorm(100))
mediators(exper)
```

mediators<-,mediation_data-method *Set the Mediators in a Mediation Data Object*

Description

This is an setter method for the mediators slot in a mediation data object. It lets you supply a new mediators data.frame for the object.

Usage

```
## S4 replacement method for signature 'mediation_data'
mediators(object) <- value
```

Arguments

object	An object of class mediation_data.
value	The new mediators values for the object.

Value

A version of `object` whose mediators slot has been replaced.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
mediators(exper) <- data.frame(m = 1:10)
exper
```

`mindfulness` *Mindfulness Dataset*

Description

Data from a study of the relationship between mindfulness and the microbiome, stored as a phyloseq object. Measurements are from 50 subjects before and after a real and active control mindfulness intervention. We are interested in changes in subject-level metadata, stored in the sample_data slot

Format

An object of class phyloseq.

Value

A phyloseq object with 307 samples and 55 taxa. Samples are described by 7 variables (potential mediators) in the sample_data slot. There is no associated phylogenetic tree.

Examples

```
data(mindfulness)
```

`model-class` *Representation of an Outcome or Mediation Model*

Description

To work with many model types simultaneously, multimedia uses a model class with the necessary mediation model functionality that wraps any specific implementation. The slots below define the generally required functionality for any specific implementation.

Slots

`estimator` A function that takes a formula, input data frame X, and an response data.frame Y and returns a model. For example, for the random forest model, this is created by wrapping `parallelize()` on the ranger() function for random forest estimation function using the 'ranger' package.

`estimates` A list containing the estimated model.

`sampler` A function that supports sampling new responses from the estimated model.

`model_type` A string specifying the type of model associated with the class. For example, "rf_model()" denotes a random forest model.

`predictor` A function that returns fitted predictions given new inputs. For example, this can be the original predict() method for a multivariate response model, or it can be a loop over predicts for each feature in the mediation or outcome model.

Examples

```
m <- lm_model()
estimator(m)(mpg ~ hp + wt, data = mtcars)

m <- rf_model()
estimator(m)(mpg ~ hp + wt, data = mtcars)
```

`multimedia` `multimedia` *Constructor*

Description

`multimedia` objects encapsulate the model and data that underlie a mediation analysis, together with metadata (like graph structure) that contextualize the estimation. This function can be used to construct a new multimedia instances from a `mediation_data` dataset and pair of estimators.

Usage

```
multimedia(
  mediation_data,
  outcome_estimator = lm_model(),
  mediation_estimator = lm_model()
)
```

Arguments

`mediation_data` An object of class `mediation_data`, with separate slots for each of the node types in a mediation analysis diagram.

`outcome_estimator`
An object of class `model` that will be used to estimate the outcome model.

`mediation_estimator`
An object of class `model` that will be used to estimate the mediation model.

Value

An object of class `multimedia` encapsulating the full mediation model and data.

See Also

multimedia-class

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper)

exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
multimedia(exper)

# real data example with a pretreatment variable
data(mindfulness)
exper <- mediation_data(
    mindfulness,
    phyloseq::taxa_names(mindfulness),
    "treatment",
    starts_with("mediator"),
    "subject"
)
multimedia(exper)
```

---

nrow,mediation_data-method

How many samples in the mediation dataset?

Description

How many samples in the mediation dataset?

Usage

```
## S4 method for signature 'mediation_data'
nrow(x)
```

Arguments

`x`	The `mediation_data` object whose number of samples we want to return.

Value

An integer giving the number of samples in the mediation object.

`nullify` *Nullify Active Edges*

Description

For inference, we often want to work with synthetic negative controls. One way to define them is to specify submodels of the full mediation analysis model. This function defines submodels by removing estimated edges according to a prespecified vector of IDs. For example, setting nulls = "T -> Y" will remove any direct effect when sampling or obtaining predictions for the full mediation analysis model $hatY$.

Usage

```
nullify(multimedia, nulls = NULL)
```

Arguments

`multimedia`	A fitted object of class multimedia with estimates along all paths in the mediation analysis DAG.
`nulls`	A string specifying the indices of edges to ignore. "T->Y", "T->M", and "M->Y" will match all edges between treatment to outcome, treatment to mediator, etc. Otherwise, the vector of indices specifying which edges to ignore.

Value

multimedia A version of the input multimedia model with all edges matching `nulls` removed. Enables sampling of synthetic null controls.

Examples

```
# example with null data
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
fit <- multimedia(exper) |>
    estimate(exper)

nullify(fit, "T->M") |>
    estimate(exper) |>
    indirect_overall()
nullify(fit, "T->Y") |>
    estimate(exper) |>
    direct_effect()

# example with another dataset
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
fit <- multimedia(exper) |>
    estimate(exper)
```

```
nullify(fit, "T->M") |>
    estimate(exper) |>
    indirect_overall()
nullify(fit, "T->Y") |>
    estimate(exper) |>
    direct_effect()
```

`null_contrast` *Compare Effects from Experimental vs. Null Mediation Data*

Description

One way to calibrate our conclusions from complex workflows is to see how they would look on data where we know that there is no effect. This function compares estimators f between real and synthetic null data, where the null removes a set of edges according to the nullfication argument.

Usage

```
null_contrast(model, exper, nullification = "T->Y", f = direct_effect)
```

Arguments

`model`	An object of class multimedia with specified mediation and outcome models that we want to re-estimate across B bootstrap samples.
`exper`	An object of class multimedia_data containing the mediation and outcome data from which the direct effects are to be estimated.
`nullification`	A string specifying the types of edges whose effects we want to remove in the null samples. Valid options are "T->Y" (the default), "T->M", "M->Y", which remove direct effects, treatment to mediator effects, and mediator to treatment effects, respectively.
`f`	The estimator that we want to compare between real and null data. This is assumed to be a function taking counterfactual samples, for example `direct_effects` or `indirect_effects`.

Value

A data.frame containing estimates on the real and synthetic data for every coordinate in the estimator f. The column `source` specifies whether the estimate was calculated using real or synthetic null data.

See Also

null_contrast fdr_summary

Examples

```
# example with null data - notice synthetic data has larger effect.
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    estimate(exper) |>
    null_contrast(exper)

# example with another dataset - synthetic effect is smaller.
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
multimedia(exper) |>
    estimate(exper) |>
    null_contrast(exper)
```

---

n_mediators *Number of Mediators in a Multimedia Object*

Description

Number of Mediators in a Multimedia Object

Usage

```
n_mediators(object)
```

Arguments

`object`	An object of class multimedia.

Value

An integer specifying the number of mediators.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    n_mediators()
```

n_outcomes *Number of Outcomes in a Multimedia Object*

Description

Number of Outcomes in a Multimedia Object

Usage

```
n_outcomes(object)
```

Arguments

object	An object of class multimedia.

Value

An integer specifying the number of outcomes.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    n_outcomes()
```

outcomes *Access Outcomes*

Description

This is an getter method for outcomes in an S4 object, usually of class mediation_data.

Usage

```
outcomes(object)
```

Arguments

object	An object whose outcomes slot to modify.

Value

The output depends on the S4 class of the input object. If it is a multimedia model object, it will return a character vector of the outcome variable names. Otherwise it returns the outcome data.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
outcomes(exper)
```

`outcomes,mediation_data-method`

Outcomes Data in a Mediation Data Object

Description

This is an accessor function to the @outcomes slot in a mediation data object. It returns the entire outcomes dataset, in contrast to outcomes() applied to a multimedia object, which only returns the names of the outcome variables.

Usage

```
## S4 method for signature 'mediation_data'
outcomes(object)
```

Arguments

`object`	An object of class mediation_data.

Value

A data.frame whose rows are samples and columns different outcomes.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
outcomes(exper)
```

`outcomes,multimedia-method`

Names of Outcomes in a Multimedia Object

Description

This is a helper that returns the names of the outcome variables in an object of class multimedia. It parses the graph in the DAG specifying the mediation analysis, and it returns all variables of node type outcome.

Usage

```
## S4 method for signature 'multimedia'
outcomes(object)
```

Arguments

object	An object of class multimedia.

Value

m A vector of strings containing the names of all the outcomes.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    outcomes()

exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
multimedia(exper) |>
    outcomes()
```

---

outcomes<- *Set Outcomes This is an setter method for outcomes in an S4 object, usually of class mediation_data.*

Description

Set Outcomes This is an setter method for outcomes in an S4 object, usually of class mediation_data.

Usage

```
outcomes(object) <- value
```

Arguments

object	An object whose outcomes slot to modify.
value	The new outcome values to set within object.

Value

Modifies the outcomes slot of the input object in place.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
outcomes(exper) <- data.frame(new_PHQ = rnorm(100))
```

outcomes<-,mediation_data-method *Set the Outcomes in a Mediation Data Object*

Description

This is an setter method for the outcomes slot in a mediation data object. It lets you supply a new outcomes data.frame for the object.

Usage

```
## S4 replacement method for signature 'mediation_data'
outcomes(object) <- value
```

Arguments

object	An object of class mediation_data.
value	The new outcome values for the object.

Value

A version of `object` whose outcomes slot has been replaced.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
outcomes(exper) <- data.frame(y = 1:10)
exper
```

outcome_model *Access the Outcome Model in a Multimedia Object*

Description

This is an accessor to the outcome slot of a multimedia object.

Usage

```
outcome_model(object)
```

Arguments

object	An object of class multimedia.

Value

NULL, if not fitted, or the model learned from the training mediation data object. For models fit in parallel across outcomes (e.g., glmnet_model()) a list of separate model objects. For models fitted jointly across outcomes (e.g., lnm_model()), a single model object of that class.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    outcome_model()
```

---

outcome_models *Accessor for Outcome Models*

Description

Accessor for Outcome Models

Usage

```
outcome_models(object)
```

Arguments

object	An object of class multimedia whose outcome model estimates we would like to extract.

Value

A list containing all the fitted outcome models.

Examples

```
data(mindfulness)
exper <- mediation_data(
    mindfulness,
    phyloseq::taxa_names(mindfulness),
    "treatment",
    starts_with("mediator"),
    "subject"
)

m <- multimedia(exper)
outcome_models(m)
```

parallelize *Parallelize Estimation across Responses*

Description

For many mediation and outcome models, we simply want to apply a univariate model across all response variable. Parallelize enables this conversion. For example, applying parallelize to ranger() returns a function that estimates separate random forest models for each response on the left hand side of a formula.

Usage

```
parallelize(f, progress = TRUE)
```

Arguments

f	A function for estimating a single response model given a formula and input dataset. This is the model that we would like to parallelize across responses.
progress	A logical indicating whether to show a progress bar.

Value

f_multi A function that takes a formula and dataset and applies f to each response on the left hand side of the original formula.

Examples

```
mat <- data.frame(matrix(rnorm(100), 25, 4))
colnames(mat) <- c("y1", "y2", "x1", "x2")
plm <- parallelize(lm)
plm(y1 + y2 ~ x1 + x2, mat)

prf <- parallelize(ranger::ranger)
prf(mpg + hp ~ wt + disp + cyl, data = mtcars)
```

plot_mediators *Visualize Indirect Effects*

Description

This is a helper function to visualize the raw data responsible for the largest indirect effects. It returns a faceted plot of outcome vs. mediator pairs for those with high pathwise indirect effects.

Usage

```
plot_mediators(
  indirect_effects,
  exper,
  n_digit = 3,
  n_panels = NULL,
  treatment = "treatment",
  ...
)
```

Arguments

indirect_effects	A data.frame containing estimated indirect effects for each variable, under different counterfactual settings for the "direct treatment." This is the output of `indirect_pathwise`.
exper	An object of class `mediation_data` containing all mediation analysis data.
n_digit	The number of digits of the indirect effects to print next to each panel. Defaults to 3.
n_panels	The number of mediator-outcome pairs to show. Defaults to 12, or the number of pathways, if there are fewer than 12.
treatment	What is the name of the treatment variable that we want to overlay on points? This is necessary when there are several potential treatment variables. Defaults to "treatment."
...	Further keyword arguments passed to `patchwork::wrap_plots`.

Value

A patchwork-based arrangement of ggplot2 grobs.

Examples

```
# dataset with no true effects
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
ie <- multimedia(exper) |>
    estimate(exper) |>
    indirect_pathwise() |>
    effect_summary()
plot_mediators(ie, exper)

# another dataset
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
ie <- multimedia(exper, rf_model()) |>
    estimate(exper) |>
    indirect_pathwise() |>
    effect_summary()
plot_mediators(ie, exper)
```

plot_sensitivity *Generic Sensitivity Plot*

Description

This function draws a curve of indirect effect against the sensitivity parameter, allowing users to specify the name of x and y-axis variables using the x_var and y_var inputs.

Usage

```
plot_sensitivity(sensitivity_curve, x_var = "rho", y_var = "indirect_effect")
```

Arguments

sensitivity_curve	The output of a call to sensitivity or sensitivity_perturb. A data.frame whose columns are: outcome, {x_var}, {y_var}, and {y_var}_standard_error, where x_var and y_var are defined in the two arguments below.
x_var	The type of perturbation variable to plot along the x-axis. Defaults to "rho", following the sensitivity approach implemented in sensitivity_subset.
y_var	The type of effect to plot along the y-axis. Defaults to indirect_effect.

Value

A ggplot2 grob plotting the sensitivity parameter against the effect specified by `y_var`.

Examples

```
sensitivity_curve <- read.csv(url("https://go.wisc.edu/j2kvcj"))
plot_sensitivity(sensitivity_curve)
```

predict,multimedia-method *Predictions from a Multimedia Class*

Description

This generalizes the built-in predict method to the multimedia class. Given an estimated multimedia object, this function supports prediction along the estimated DAG. It first predicts `hat[M] | T, X` and then `hat[Y] | hat[M], T, X`. Each prediction step will call the prediction method within the mediation and outcome models that make up the multimedia object on which this is called. By passing in new treatment, mediator, or pretreatment data, you can predict forward along the DAG using these as inputs.

Usage

```
## S4 method for signature 'multimedia'
predict(object, profile = NULL, mediators = NULL, pretreatment = NULL, ...)
```

Arguments

object	An object of class multimedia containing the estimated mediation and outcome models whose mediation and outcome predictions we want to obtain.
profile	An object of class treatment_profile containing the treatment profile to consider in the difference. Defaults to a profile with all the unique treatment configurations observed in the original data, shared across both the mediators and outcomes.
mediators	By default, we will return outcome predictions using the predicted mediators from the mediation model. Modify this argument if you would like to directly control the mediation inputs for the outcome model. Must be a data.frame whose columns are named to match the mediators(object).
pretreatment	By default, we will return mediation and outcome model predictions using the same pretreatment variables as used when initially estimating the models (like setting newdata = NULL in usual predict). To pass in different pretreatment variables, provide a data.frame here whose columns match the pretreatments as the originally trained mediation and outcome models.
...	A placeholder to agree with predict in base R. Not ever used.

Value

A list with two elements: $mediators: A data.frame containing predicted values for the mediators. Each row corresponds to one row of the newdata, or one row of the default treatment profile, if no newdata is given.

$outcomes: A data.frame containing predicted values for the outcomes, given either (i) the predicted values of the mediators or (ii) the provided values of the mediators. Each row corresponds to one row of the newdata, or one row of the default treatment profile, if no newdata is given.

Examples

```
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
fit <- multimedia(exper, glmnet_model()) |>
    estimate(exper)
predict(fit)

# at new treatment configurations
t1 <- data.frame(treatment = factor(rep(c(0, 1), each = 5)))
profile <- setup_profile(fit, t_mediator = t1, t_outcome = t1)
predict(fit, profile)

# at new treatment and mediator configurations
mediators <- data.frame(mediator = rnorm(10, 0, 1))
predict(fit, profile, mediators)
```

predict_across *Predict a Subset of Responses*

Description

Predict across selected responses in a mediation model object. This is a lower-level version of the predict method that applies to objets of class mediation_model. Rather than giving predictions across all outcomes, it supports predictions across a subset specified as a vector. This can be convenient when we want to analyze how a specific subset of outcomes changes and do not need to run predictions across all possible.

Usage

```
predict_across(object, newdata, name)
```

Arguments

object	An object of class model containing an estimated model.
newdata	A data.frame containing new inputs from which to sample responses. If NULL, defaults to the data used to estimate fit.
name	A string or index specifying which of the dimensions of a multiresponse prediction to extract.

Value

A vector of predicted values for the outcome of interest.

Examples

```
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
fit <- multimedia(exper) |>
    estimate(exper)
predict_across(outcome_model(fit), NULL, "outcome_1")

# predict at newdata
newdata <- bind_mediation(exper)
predict_across(
    outcome_model(fit),
    newdata[seq_len(5), ],
    c("outcome_1", "outcome_2")
)
```

pretreatments *Access Pretreatments*

Description

This is an getter method for pretreatments in an S4 object, usually of class mediation_data.

Usage

```
pretreatments(object)
```

Arguments

object	An object whose pretreatments slot to modify.

Value

The output depends on the S4 class of the input object. If it is a multimedia model object, it will return a character vector of the outcome variable names. Otherwise it returns the outcome data.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
pretreatments(exper)
```

pretreatments,mediation_data-method *Pretreatments in a Mediation Data Object*

Description

This is an accessor function to the @pretreatments slot in a mediation data object. It returns the entire set of observed pretreatments, in contrast to pretreatments() applied to a multimedia object, which only returns the names of the pretreatment variables.

Usage

```
## S4 method for signature 'mediation_data'
pretreatments(object)
```

Arguments

object	An object of class mediation_data.

Value

A data.frame whose rows are samples and columns different pretreatments.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
pretreatments(exper)
```

`pretreatments<-` *Set Pretreatments This is an setter method for pretreatments in an S4 object, usually of class mediation_data.*

Description

Set Pretreatments This is an setter method for pretreatments in an S4 object, usually of class mediation_data.

Usage

```
pretreatments(object) <- value
```

Arguments

`object`	An object whose pretreatments slot to modify.
`value`	The new pretreatment values to set within object.

Value

Modifies the pretreatments slot of the input object in place.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
pretreatments(exper) <- data.frame(new_var = rnorm(100))
```

pretreatments<-,mediation_data-method *Set the Pretreatments in a Mediation Data Object*

Description

This is an setter method for the pretreatments slot in a mediation data object. It lets you supply a new pretreatments data.frame for the object.

Usage

```
## S4 replacement method for signature 'mediation_data'
pretreatments(object) <- value
```

Arguments

`object`	An object of class mediation_data.
`value`	The new pretreatment values for the object.

Value

A version of `object` whose pretreatments slot has been replaced.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
pretreatments(exper) <- data.frame(x = 1:10)
exper
```

retrieve_names *Variables in a Multimedia Object*

Description

This returns all the variables modeled within a multimedia object. This can be helpful for overviewing an experiment and is called by the print methods in this package. Also supports filtering to specific node types, e.g., mediators.

Usage

```
retrieve_names(object, nm)
```

Arguments

object	An object of class multimedia
nm	A string specifying the node type to filter down to, e.g., 'treatment' or 'mediator'.

Value

A character vector containing all the names of the variables of the node type nm.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    retrieve_names("mediator")
```

---

rf_model *Random Forest Model*

Description

Apply a random forest model in parallel across a vector of responses y in either an outcome or mediation model. This is a natural choice when the relationship between inputs and outputs is thought to be nonlinear. Internally, each of the models across the response are estimated using the 'ranger' package.

Usage

```
rf_model(progress = TRUE, ...)
```

Arguments

progress	A logical indicating whether to show a progress bar during estimation.
...	Keyword parameters passed to ranger() in the 'ranger' package.

Value

model An object of class model with estimator, predictor, and sampler functions associated wtih a lienar model.

See Also

model lm_model rf_model glmnet_model brms_model

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper, rf_model(num.trees = 10)) |>
    estimate(exper)

# example with another dataset
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
multimedia(exper, rf_model(num.trees = 20, max.depth = 2)) |>
    estimate(exper)
```

---

rf_sampler *Sample from a Random Forest Model*

Description

This assumes a continuous response, so that the out-of-sample MSE can be used to estimate the outcome variability σ.

Usage

```
rf_sampler(fits, newdata = NULL, indices = NULL, ...)
```

Arguments

fits	The fitted RF model from which to draw samples.
newdata	A data.frame containing new inputs from which to sample responses. If NULL, defaults to the data used to estimate fit.
indices	The coordinates of the response from which to draw samples.
...	Additional parameters passed to rf_model's predict method.

Value

y_star A data.frame of samples y associated with the new inputs.

Examples

```
m <- rf_model()
fit <- estimator(m)(mpg ~ hp + wt, data = mtcars)
rf_sampler(fit, mtcars)

prf <- parallelize(ranger::ranger)
fit <- prf(mpg + disp ~ hp + wt, data = mtcars)
rf_sampler(fit, mtcars)
```

`sample,multimedia-method`

Sample New Mediator/Outcome Data

Description

This generalizes the built-in sample method to the multimedia class. Given an estimated multimedia object, this function supports sampling along the estimated DAG. It first samples `M* | T, X` and then `Y* | M*, T, X`. Each sampling step will call the sample method within the mediation and outcome models that make up the multimedia object on which this is called.

Usage

```
## S4 method for signature 'multimedia'
sample(x, size, pretreatment = NULL, profile = NULL, mediators = NULL, ...)
```

Arguments

`x`	An object of class multimedia containing the estimated mediation and outcome models whose mediation and outcome samples we want to obtain.
`size`	A placeholder argument to agree with the default `sample` method in R base. We always return the number of samples as set in either the original input `x` or a new input `profile`.
`pretreatment`	By default, we will return mediation and outcome model predictions using the same pretreatment variables as used when initially estimating the models (like setting `newdata = NULL` in usual `predict`). To pass in different pretreatment variables, provide a data.frame here whose columns match the pretreatments as the originally trained mediation and outcome models.
`profile`	An object of class `treatment_profile` containing the treatment profile to consider in the difference. Defaults to a profile with all the unique treatment configurations observed in the original data, shared across both the mediators and outcomes.
`mediators`	By default, we will return outcome predictions using the predicted mediators from the mediation model. Modify this argument if you would like to directly control the mediation inputs for the outcome model. Must be a data.frame whose columns are named to match the `mediators(object)`.
`...`	Additional options to pass to the @sampler method in the estimated mediation model.

Value

An object of class `multimedia` with mediator and outcome slots sampled according to the description above.

Examples

```
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
fit <- multimedia(exper) |>
    estimate(exper)
samples <- sample(fit)
mediators(samples)
outcomes(samples)

# sampling with just different "n" has no effect.
samples <- sample(fit, 100)

# Instead sample at a new treatment configuration
t1 <- data.frame(treatment = factor(rep(c(0, 1), each = 50)))
profile <- setup_profile(fit, t_mediator = t1, t_outcome = t1)
samples <- sample(fit, profile = profile)
mediators(samples)
outcomes(samples)
```

`sensitivity` *Sensitivity Analysis for Overall Indirect Effect*

Description

For causal identification, mediation analysis relies on several untestable assumptions. One important one is that there is no confounding between the counterfactual mediator and outcome variables. Even though we can never know whether this exists, we can measure the sensitivity of our conclusions to the existence/strength of such a confounder. In this function, we approach this by inducing (unallowable) correlation between the mediator and outcome model residuals, simulate forward, and see how the estimated overall indirect effect changes.

Usage

```
sensitivity(
  model,
  exper,
  confound_ix = NULL,
  rho_seq = NULL,
  n_bootstrap = 100,
  progress = TRUE,
  ...
)
```

Arguments

`model`	A `multimedia` object containing the fitted models for sensitivity analysis. Note that since our approach relies on correlating simulated residual error, it is only applicable to models of class `lm_model()`, `glmnet_model()` and `rf_model()`.

exper	The original mediation_data class object used to fit model. These observations will be resampled to support bootstrap confidence interval construction of the sensitivity curve.
confound_ix	A data.frame specifying which mediator/outcome should be allowed to be correlated. Should have two columns: 'mediator' and 'outcome' specifying which pairs of mediators and outcomes should be correlated. Defaults to NULL, which creates a data.frame with no rows (and so enforcing independence between mediators and outcomes)
rho_seq	We will evaluate correlations Cor(e', e) between mediation and outcome model errors ranging along this grid. Defaults to NULL, which internally sets the sequence to rho = (-0.9, -0.7 ..., 0.7, 0.9).
n_bootstrap	The number of bootstrap resamples used to build confidence bands around the sensitivity curves. Defaults to 100.
progress	A logical indicating whether to show a progress bar.
...	Additional keyword arguments passed to indirect_overall.

Value

A data.frame giving the outputs of indirect_overall across many values of the correlation rho.

Examples

```
xy_data <- demo_spline()
exper <- mediation_data(
    xy_data, starts_with("outcome"), "treatment", "mediator"
)
model <- multimedia(
    exper,
    outcome_estimator = glmnet_model(lambda = 1e-2)
) |>
    estimate(exper)
rho_seq <- c(-0.2, 0, 0.2)
confound_ix <- expand.grid(mediator = 1, outcome = 1:2)
sensitivity(model, exper, confound_ix, rho_seq, n_bootstrap = 2)
```

---

sensitivity_pathwise *Sensitivity Analysis for Pathwise Indirect Effects*

Description

For causal identification, mediation analysis relies on several untestable assumptions. One important one is that there is no confounding between the counterfactual mediator and outcome variables. Even though we can never know whether this exists, we can measure the sensitivity of our conclusions to the existence/strength of such a confounder. In this function, we approach this by inducing (unallowable) correlation between the mediator and outcome model residuals, simulate forward, and see how the estimated pathwise indirect effects change.

Usage

```
sensitivity_pathwise(
  model,
  exper,
  confound_ix = NULL,
  rho_seq = NULL,
  n_bootstrap = 100,
  progress = TRUE,
  ...
)
```

Arguments

model	A multimedia object containing the fitted models for sensitivity analysis. Note that since our approach relies on correlating simulated residual error, it is only applicable to models of class lm_model(), glmnet_model() and rf_model().
exper	The original mediation_data class object used to fit model. These observations will be resampled to support bootstrap confidence interval construction of the sensitivity curve.
confound_ix	A data.frame specifying which mediator/outcome should be allowed to be correlated. Should have two columns: 'mediator' and 'outcome' specifying which pairs of mediators and outcomes should be correlated. Defaults to NULL, which creates a data.frame with no rows (and so enforcing independence between mediators and outcomes)
rho_seq	We will evaluate correlations Cor(e', e) between mediation and outcome model errors ranging along this grid. Defaults to NULL, which internally sets the sequence to rho = (-0.9, -0.7 ..., 0.7, 0.9).
n_bootstrap	The number of bootstrap resamples used to build confidence bands around the sensitivity curves. Defaults to 100.
progress	A logical indicating whether to show a progress bar.
...	Additional arguments passed to indirect_pathwise.

Value

A date.frame giving the outputs of indirect_overall across many values of the correlation rho.

Examples

```
xy_data <- demo_spline()
exper <- mediation_data(
    xy_data, starts_with("outcome"), "treatment", "mediator"
)
model <- multimedia(
    exper,
    outcome_estimator = glmnet_model(lambda = 1e-2)
) |>
    estimate(exper)
rho_seq <- c(-0.2, 0, 0.2)
```

```
subset_indices <- expand.grid(
    mediator = n_mediators(model), outcome = n_outcomes(model)
)
sensitivity_pathwise(model, exper, subset_indices, rho_seq, n_bootstrap = 2)
```

`sensitivity_perturb` *Sensitivity to User-Specified Perturbations*

Description

The more standard sensitivity and sensitivity_pathwise functions support sensitivity analysis to violations in assumptions restricted to specific mediator-outcome pairs. For more general violations, this function allows arbitrary modification of the default, diagonal covariance matrix structure across both mediators and outcomes. This makes it possible to ask what happens when mediators are correlated with one another or when more some pairs of mediator-outcome pairs have much stronger correlation than others.

Usage

```
sensitivity_perturb(
  model,
  exper,
  perturb,
  nu_seq = NULL,
  n_bootstrap = 100,
  progress = TRUE
)
```

Arguments

`model`	A `multimedia` object containing the fitted models for sensitivity analysis. Note that since our approach relies on correlating simulated residual error, it is only applicable to models of class `lm_model()`, `glmnet_model()` and `rf_model()`.
`exper`	The original `mediation_data` class object used to fit `model`. These observations will be resampled to support bootstrap confidence interval construction of the sensitivity curve.
`perturb`	A matrix towards which the original mediator-outcome covariance should be perturbed. Must have dimension (n_mediators + n_outcomes) x (n_mediators + n_outcomes).
`nu_seq`	The strength of the perturbation towards the matrix perturb.
`n_bootstrap`	The number of bootstrap resamples used to build confidence bands around the sensitivity curves. Defaults to 100.
`progress`	A logical indicating whether to show a progress bar.

Details

Specifically, it defines a new covariance matrix across mediators and outcomes according to diag(sigma^2_mediator, sigma^2_outcome) + nu * perturb. The estimates sigma^2 are taken from the residuals in the original mediation and outcome models, and perturb and nu are provided by the user.

Value

A `date.frame` giving the outputs of `indirect_overall` across many values of the correlation rho.

Examples

```
xy_data <- demo_spline()
exper <- mediation_data(
    xy_data, starts_with("outcome"), "treatment", "mediator"
)
model <- multimedia(
    exper,
    outcome_estimator = glmnet_model(lambda = 1e-2)
) |>
    estimate(exper)
nu_seq <- c(-0.2, 0.2)
perturb <- matrix(
    c(
        0, 3, 0,
        3, 0, 0,
        0, 0, 0
    ),
    nrow = 3, byrow = TRUE
)
sensitivity_perturb(model, exper, perturb, nu_seq, n_bootstrap = 2)
```

`setup_profile` *Define a* `treatment_profile` *object*

Description

For general mediation analysis, we need to provide counterfactuals for both the outcome and mediator components of each sample. That is, we need to understand Y(t, M(t')) where t and t' may not be the same. `treatment_profile` classes place some more structural requirements on treatment profiles, so that later effect estimation can make simplifying assumptions. This function creates a treatment profile from a collection of possible mediator and outcome treatments.

Usage

```
setup_profile(x, t_mediator = NULL, t_outcome = NULL)
```

Arguments

x	An object of class multimedia specifying the complete mediation analysis DAG. The treatment, mediator, and outcome names are necessary to build a profile of counterfactual treatments across each of these variables.
t_mediator	A data.frame whose columns store treatment names and whose values are the treatment assignments to each sample (row). Defaults to NULL, in which case this type of data.frame is constructed from the treatment assignments in the mediation_data's @treatment slot. Each column must be either a numeric or factor variable.
t_outcome	A data.frame analogous to t_mediator, but applying to the outcome node.

Value

An object of class treatment_profile giving treatment assignments for both mediation and outcome terms.

See Also

check_profile

Examples

```
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
fit <- multimedia(exper) |>
    estimate(exper)

t1 <- data.frame(treatment = factor(rep(c(0, 1), each = 5)))
profile <- setup_profile(fit, t_mediator = t1, t_outcome = t1)
profile

t2 <- data.frame(treatment = factor(rep(0, 10)))
profile <- setup_profile(fit, t_mediator = t1, t_outcome = t2)
profile
```

---

spline_fun *Generate Random Spline*

Description

This generates random spline functions. It is used in the toy nonlinear dataset created in demo_spline(). This is not necessary for the essential multimedia workflow, it is only exported for potential independent interest.

Usage

```
spline_fun(D = 2, knots = NULL, h_ix = seq_len(10), ...)
```

Arguments

D	The number random spline functions to generate internally.
knots	The location of knots to use in the spline functions. Defaults to -4, -2, 0, 2, 4.
h_ix	The locations along which to generate the underlying spline function.
...	Additional arguments to pass to rnorm during the random noise generation for each call of the returned function.

Value

A function that can be used to sample points along the random spline functions. Takes argument 'x' giving the input to the spline and returns a D-dimensional response y giving random samples for each of the D splines..

See Also

demo_spline

---

sub_formula *Helper to Modify Formulas*

Description

Helper to Modify Formulas

Usage

```
sub_formula(formula, yj)
```

Arguments

formula	The original formula whose response we want to modify.
yj	The desired response term for the formula.

Value

A new formula object with the LHS replaced by yj.

`treatments` *Access Treatments*

Description

This is an getter method for treatments in an S4 object, usually of class mediation_data.

Usage

```
treatments(object)
```

Arguments

`object`	An object whose treatments slot to modify.

Value

The output depends on the S4 class of the input object. If it is a multimedia model object, it will return a character vector of the outcome variable names. Otherwise it returns the outcome data.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
treatments(exper)
```

`treatments,mediation_data-method`
Treatments in a Mediation Data Object

Description

This is an accessor function to the @treatments slot in a mediation data object. It returns the entire set of observed treatments, in contrast to treatments() applied to a multimedia object, which only returns the names of the treatment variables.

Usage

```
## S4 method for signature 'mediation_data'
treatments(object)
```

Arguments

`object`	An object of class mediation_data.

Value

A data.frame whose rows are samples and columns different treatments.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
treatments(exper)
```

---

treatments,multimedia-method
Names of Treatments in a Multimedia Object

Description

This is a helper that returns the names of the treatment variables in an object of class multimedia. It parses the graph in the DAG specifying the mediation analysis, and it returns all variables of node type treatment.

Usage

```
## S4 method for signature 'multimedia'
treatments(object)
```

Arguments

object	An object of class multimedia.

Value

m A vector of strings containing the names of all the treatments.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
multimedia(exper) |>
    treatments()
```

`treatments<-` *Set Treatments This is an setter method for treatments in an S4 object, usually of class mediation_data.*

Description

Set Treatments This is an setter method for treatments in an S4 object, usually of class mediation_data.

Usage

```
treatments(object) <- value
```

Arguments

`object`	An object whose treatments slot to modify.
`value`	The new treatment values to set within object.

Value

Modifies the treatments slot of the input object in place.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
treatments(exper) <- data.frame(new_treatment = rnorm(100))
```

`treatments<-,mediation_data-method` *Set the Treatments in a Mediation Data Object*

Description

This is an setter method for the treatments slot in a mediation data object. It lets you supply a new treatments data.frame for the object.

Usage

```
## S4 replacement method for signature 'mediation_data'
treatments(object) <- value
```

Arguments

`object`	An object of class mediation_data.
`value`	The new treatment values for the object.

Value

A version of `object` whose treatment slot has been replaced.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
treatments(exper) <- data.frame(t = rep(0, 10))
exper
```

`treatment_profile-class`

Define a Treatment Profile

Description

This class ensures appropriate structure of the treatment assignments for mediator and outcome variables. It enforces certain structural requirements (e.g., that the number of samples is the same under the mediator and outcome counterfactuals) using the `check_profile` function.

See Also

setup_profile check_profile

Examples

```
exper <- demo_spline(tau = c(2, 1)) |>
    mediation_data(starts_with("outcome"), "treatment", "mediator")
fit <- multimedia(exper) |>
    estimate(exper)

# helpers for defining treatment profiles
t1 <- data.frame(treatment = factor(rep(c(0, 1), each = 5)))
profile <- setup_profile(fit, t_mediator = t1, t_outcome = t1)
profile

t2 <- data.frame(treatment = factor(rep(0, 10)))
profile <- setup_profile(fit, t_mediator = t1, t_outcome = t2)
profile
```

`[,mediation_data,ANY,ANY,ANY-method`
Subset a mediation dataset

Description

We can subset samples by indexing into a mediation dataset. It will subsample all fields – pretreatments, treatments, mediators, and outcomes. Note that there is no way to subset columns in this way, since they would be different across each source.

Usage

```
## S4 method for signature 'mediation_data,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]
```

Arguments

`x`	An object of class `mediation_data` whose samples we want to subset.
`i`	An integer or integer/logical vector specifying the samples to subset to.
`j`	A placeholder to argree with R's `[` function. Never used.
`...`	A placeholder to agree with R's `[` function. Never used.
`drop`	A placeholder to agree with R's `[` function. Never used.

Value

A version of the input mediation_data object whose @mediators, @outcomes, @treatments, and @pretreatments rows have all been subsetted according to `i`.

Examples

```
exper <- demo_joy() |>
    mediation_data("PHQ", "treatment", starts_with("ASV"))
exper[1]
exper[1:10]
```

Index