Package ‘JSmediation’

February 28, 2024

Title Mediation Analysis Using Joint Significance
Version 0.2.2
Description A set of helper functions to conduct joint-significance tests for mediation analysis, as recommended by Yzerbyt, Muller, Batailler, & Judd. (2018) <doi:10.1037/pspa0000132>.
License MIT + file LICENSE
URL https://jsmediation.cedricbatailler.me/, https://github.com/cedricbatailler/JSmediation
BugReports https://github.com/cedricbatailler/JSmediation/issues
Depends R (>= 2.10)
Imports broom, data.table, dplyr, glue, knitr, magrittr, MASS, purrr, rlang (>= 0.4.9), stats, tibble, tidyselect
Suggests covr, lavaan, markdown, mediation, patchwork, performance, rmarkdown, roxygen2, see (>= 0.6.5), stringr, testthat (>= 3.0.2), vdiffr, withr
VignetteBuilder knitr
ByteCompile true
Config/testthat/edition 3
Config/testthat/parallel true
Encoding UTF-8
LazyData true
RoxygenNote 7.3.1
NeedsCompilation no
Author Cédric Batailler [aut, cre] (<https://orcid.org/0000-0003-0553-6827>), Dominique Muller [aut] (<https://orcid.org/0000-0001-9544-5317>), Vincent Yzerbyt [aut], Charles Judd [aut], Yoann Julliard [ctb] (<https://orcid.org/0000-0002-5428-6994>), Arnold Ho [dtc], Nour Kteily [dtc],
add_index

adds an indirect effect index to a fitted mediation model

Description

*add_index* is a generic function that adds a (moderated) indirect effect index to an object created with an *mdt_* family function. This index is computed using Monte Carlo methods. This function invokes particular methods depending of the class of the mediation model. For example, with a model fitted with *mdt_simple*, *add_index* will invoke *add_index.simple_mediation*.

Usage

```r
add_index(mediation_model, times = 5000, level = 0.05, ...)
```
Arguments

mediation_model
A mediation model fitted with an mdt_* family function.
times
Number of simulations to use to compute the Monte Carlo index’s confidence interval.
level
Alpha threshold to use for the confidence interval.
... Further arguments to be passed to specific methods.

Value
An object of the same class as mediation_model, but with index added for later use.

Description
Adds the confidence interval for the index of moderated mediation to a model fitted with mdt_moderated.

Usage
### S3 method for class 'moderated_mediation'

```
add_index(mediation_model, times = 5000, level = 0.05, stage = NULL, ...)
```

Arguments

mediation_model
A mediation model of class "moderated_mediation".
times
Number of simulations to use to compute the Monte Carlo indirect effect confidence interval.
level
Alpha threshold to use for the confidence interval.
stage
Moderated indirect effect’s stage for which to compute the confidence interval. Can be either 1 (or "first") or 2 (or "second"). To compute total indirect effect moderation index, use "total".
... Further arguments passed to or from other methods.

Details
Indirect effect moderation index for moderated mediation uses \(a, a \times Mod, b,\) and \(b \times Mod\) estimates and their standard errors to compute the appropriate index product distribution using Monte Carlo methods (see Muller, Judd, & Yzerbyt, 2005).

JSmediation supports different types of mediated indirect effect index:

- **Stage 1**: computes the product between \(a \times Mod\) and \(b\).
- **Stage 2**: computes the product between \(a\) and \(b \times Mod\).
- **Total**: computes the sum of Stage 1 and Stage 2 distribution.
add_index.simple_mediation

References

Examples

```r
## getting a stage 1 moderated indirect effect index
ho_et_al$condition_c <- build_contrast(ho_et_al$condition,
                                        "Low discrimination",
                                        "High discrimination")
ho_et_al <- standardize_variable(ho_et_al, c(linkedfate, sdo), suffix ="c")
moderated_model <- mdt_moderated(data = ho_et_al,
                                  IV = condition_c,
                                  DV = hypodescent,
                                  M = linkedfate_c,
                                  Mod = sdo_c)
add_index(moderated_model, stage = 1)
```

---

**add_index.simple_mediation**

*add_index method for simple mediation*

Description
Add confidence interval for the index of mediation to a model fitted with `mdt_simple`.

Usage
```r
## S3 method for class 'simple_mediation'
add_index(mediation_model, times = 5000, level = 0.05, ...)
```

Arguments
- `mediation_model` A mediation model of class "simple_mediation".
- `times` Number of simulations to use to compute the Monte Carlo indirect effect confidence interval.
- `level` Alpha threshold to use for the confidence interval.
- `...` Further arguments passed to or from other methods.

Details
Indirect effect index for simple mediation uses $a$ and $b$ estimates and their standard errors to compute the $ab$ product distribution using Monte Carlo methods (see MacKinnon, Lockwood, & Williams, 2004).
add_index.within_participant_mediation

References

Examples

```r
## getting an indirect effect index
ho_et_al$condition_c <- build_contrast(ho_et_al$condition,
    "Low discrimination",
    "High discrimination")
simple_model <- mdt_simple(data = ho_et_al,
    IV = condition_c,
    DV = hypodescent,
    M = linkedfate)
add_index(simple_model)
```

add_index.within_participant_mediation

add_index method for within-participant mediation

Description
Adds the confidence interval for the index of within-participant mediation to a model fitted with `mdt_within` or `mdt_within_wide`.

Usage

```r
## S3 method for class 'within_participant_mediation'
add_index(mediation_model, times = 5000, level = 0.05, ...)
```

Arguments

- `mediation_model`: A mediation model of class "within_participant_mediation".
- `times`: Number of simulations to use to compute the Monte Carlo indirect effect confidence interval.
- `level`: Alpha threshold to use for the confidence interval.
- `...`: Further arguments passed to or from other methods.

Details
Indirect effect index for within-participant mediation uses $a$ and $b$ estimates and their standard error to compute the $ab$ product distribution using Monte Carlo methods (see MacKinnon, Lockwood, & Williams, 2004).
References


Examples

```r
## getting an indirect effect index
within_model <- mdt_within(data = dohle_siegrist,
                           IV = name,
                           DV = willingness,
                           M = hazardousness,
                           grouping = participant)
add_index(within_model)
```

---

**apastylr**

*Creates an APA formatted report from a significance test*

**Description**

Create an APA formatted report from the test of a specific term in a linear model.

**Usage**

```r
apastylr(model, term)
```

**Arguments**

- `model` A linear model created using `lm()`.
- `term` A character string representing a term in the linear model.

**Value**

An APA formatted character string.

**Examples**

```r
data(ho_et_al)
test <- lm(hypodescent ~ linkedfate, ho_et_al)
apastylr(test, "linkedfate")
```
**build_contrast**

Builds a contrast code from character vector

---

**Description**

This function constructs a contrast code from a character variable. It is useful when one needs to recode a two-category character variable to a numeric one.

**Usage**

```r
build_contrast(vector, cond_a, cond_b)
```

**Arguments**

- `vector`: A character vector.
- `cond_a`: A character string to be coded -0.5.
- `cond_b`: A character string to be coded 0.5.

**Details**

The `lm` method supports factor and character variables by dummy coding them. Dummy coding can make the interpretation of regression coefficient difficult or at least more difficult than contrast coding. Contrast-coded-variable coefficients interpretation is particularly useful when conducting a joint-significance test.

**Value**

A numeric vector.

**See Also**

- `scale` for centering continuous numeric variable.

**Examples**

```r
data(ho_et_al)

ho_et_al$condition_contrast <- build_contrast(ho_et_al$condition, 
                                           "Low discrimination", 
                                           "High discrimination")

head(ho_et_al)
```
check_assumptions

Test assumptions for models underlying the mediation

Description

When conducting a joint-significant test, different models are fitted to the data. This function tests assumptions regarding these models using the performance package.

The assumptions test are performed using check_normality, check_heteroscedasticity, and check_outliers.

Note that check_assumptions returns a mediation_model object.

Usage

check_assumptions(
  mediation_model,
  tests = c("normality", "heteroscedasticity")
)

Arguments

  mediation_model
    An object of class mediation_model.

  tests
    A character vector indicating which test to run. Supported test includes "normality", "heteroscedasticity", and "outliers"

Value

  Invisibly returns an object of class mediation_model.

See Also

  Other assumption checks: plot_assumptions()

Examples

data(ho_et_al)

ho_et_al$condition_c <- build_contrast(ho_et_al$condition,
  "Low discrimination",
  "High discrimination")

my_model <-
  mdt_simple(data = ho_et_al,
  IV = condition_c,
  DV = hypodescent,
  M = linkedfate)
compute_indirect_effect_for

Compute the indirect effect index for a specific value of the moderator

Description

When computing a moderated mediation, one assesses whether an indirect effect changes according a moderator value (Muller et al., 2005). mdt_moderated makes it easy to assess moderated mediation, but it does not allow accessing the indirect effect for a specific moderator values. compute_indirect_effect_for fills this gap.

Usage

compute_indirect_effect_for(
  mediation_model,
  Mod = 0,
  times = 5000,
  level = 0.05
)

Arguments

mediation_model  A moderated mediation model fitted with mdt_moderated.
Mod              The moderator value for which to compute the indirect effect. Must be a numeric value, defaults to 0.
times            Number of simulations to use to compute the Monte Carlo indirect effect confidence interval. Must be numeric, defaults to 5000.
level             Alpha threshold to use for the indirect effect’s confidence interval. Defaults to .05.

Details

The approach used by compute_indirect_effect_for is similar to the approach used for simple slope analyses. Specifically, it will fit a new moderated mediation model, but with a different variable coding. Behind the scenes, compute_indirect_effect_for adjusts the moderator variable coding, so that the value we want to compute the indirect effect for is now 0.

Once done, a new moderated mediation model is applied using the new data set. Because of the new coding, and because of how one interprets coefficients in a linear regression, $a \times b$ is now the indirect effect we wanted to compute (see the Models section).

Thanks to the returned values of $a$ and $b$ ($b_{0.1}$ and $b_{0.4}$, see the Models section), it is now easy to compute $a \times b$. compute_indirect_effect_for uses the same approach than the add_index function. A Monte Carlo simulation is used to compute the indirect effect index (MacKinnon et al., 2004).
Models

In a moderated mediation model, three models are used. compute_indirect_effect_for uses the same model specification as mdt_moderated:

- $Y_i = b_{40} + b_{41}X_i + b_{42}Mo_i + b_{43}XMo_i$
- $M_i = b_{50} + b_{51}X_i + b_{52}Mo_i + b_{53}XMo_i$
- $Y_i = b_{60} + c'_{61}X_i + b_{62}Mo_i + b_{63}XMo_i + b_{64}Me_i + b_{65}MeMo_i$

with $Y_i$, the outcome value for the $i$th observation, $X_i$, the predictor value for the $i$th observation, $Mo_i$, the moderator value for the $i$th observation, and $M_i$, the mediator value for the $i$th observation. Coefficients associated with $a$, $a \times Mod$, $b$, $b \times Mod$, $c$, $c \times Mod$, $c'$, and $c' \times Mod$, paths are respectively $b_{51}$, $b_{53}$, $b_{64}$, $b_{65}$, $b_{41}$, $b_{43}$, $b_{61}$, and $b_{63}$ (see Muller et al., 2005).

References


Examples

```r
# compute an indirect effect index for a specific value in a moderated mediation.
data(ho_et_al)
ho_et_al$condition_c <- build_contrast(ho_et_al$condition, 
  "Low discrimination",
  "High discrimination")
ho_et_al <- standardize_variable(ho_et_al, c(linkedfate, sdo))
moderated_mediation_model <- mdt_moderated(data = ho_et_al, 
  DV = hypodescent, 
  IV = condition_c, 
  M = linkedfate, 
  Mod = sdo)
compute_indirect_effect_for(moderated_mediation_model, Mod = 0)
```

---

**display_models** Displays models from a mediation object

**Description**

When conducting a joint-significance test, different models are fitted to the data. This function helps you see a summary of the models that have been used in an object of class mediation_model.
Usage

display_models(mediation_model)

Arguments

mediation_model

An object of class mediation_model.

Value

A list of summary.lm objects.

Examples

data(ho_et_al)
ho_et_al$condition_c <- build_contrast(ho_et_al$condition,
   "Low discrimination",
   "High discrimination")

my_model <-
 mdt_simple(data = ho_et_al,
 IV = condition_c,
 DV = hypodescent,
 M = linkedfate)

display_models(my_model)

dohle_siegrist

Dohle and Siegrist (2014, Exp 1) illustrating within-subject analysis (long-format)

Description

A data set containing data from Dohle and Siegrist (2014)’s Experiment 1 that can be used to conduct within-subject joint-significance test. In this experiment, researchers are interested in the effect of name complexity on willingness to buy a drug. The specific hypothesis is that complex drug names are perceived as more hazardous, which makes someone less likely to buy the drug. Researchers used a within-subject design.

This data set is in a long-format, see mdt_within to conduct a within-participant mediation analysis with this data set.

Usage

data("dohle_siegrist")
Format

A data frame with 44 rows and 4 variables:

- **participant**: Participant number.
- **name**: Name of the drugs ("simple" vs. "complex").
- **hazardousness**: Mean estimated hazardousness.
- **willingness**: Mean willingness to buy.

References


dohle_siegrist_wide

**Dohle and Siegrist (2014, Exp 1) illustrating within-subject analysis (wide-format)**

Description

A data set containing data from Dohle and Siegrist (2014)’s Experiment 1 that can be used to conduct within-subject joint-significance test. In this experiment, researchers are interested in the effect of name complexity on willingness to buy a drug. The specific hypothesis is that complex drug names are perceived as more hazardous, which makes someone less likely to buy the drug. Researchers used a within-subject design.

This data set is in a wide format, see `mdt_within_wide` to conduct a within-participant mediation analysis with this dataset.

Usage

```r
data("dohle_siegrist_wide")
```

Format

A data frame with 22 rows and 5 variables:

- **participant**: Participant number.
- **hazardousness_c**: Hazardousness for complex drug name.
- **hazardousness_s**: Hazardousness for simple drug name.
- **willingness_c**: Willingness to buy for complex drug name.
- **willingness_s**: Willingness to buy for simple drug name.

References

Explain the function `extract_model` and its arguments in a clear manner.

**Description**
When conducting a joint-significant test, different models are fitted to the data. This function helps you access the models used in an object of class `mediation_model`.

**Usage**
```r
extract_model(mediation_model, step = NULL)
```

**Arguments**
- `mediation_model`: An object of class `mediation_model`.
- `step`: An integer or a string corresponding to the model to extract.

**Value**
An `lm` object.

**See Also**
- `extract_models()` to access a list of every model relevant to joint-significance testing.
- Other extract functions: `extract_models()`, `extract_tidy_models()`

**Examples**
```r
data(ho_et_al)
ho_et_al$condition_c <- build_contrast(ho_et_al$condition,
                                       "Low discrimination",
                                       "High discrimination")

my_model <-
  mdt_simple(data = ho_et_al,  
             IV = condition_c,  
             DV = hypodescent,  
             M = linkedfate)

extract_model(my_model, step = "X -> M")
```
extract_models

Description

When conducting a joint-significant test, different models are fitted to the data. This function helps accessing the models used in an object of class mediation_model.

Usage

extract_models(mediation_model)

Arguments

mediation_model

An object of class mediation_model.

Value

A list of lm objects.

See Also

Other extract functions: extract_model(), extract_tidy_models()

Examples

data(ho_et_al)
ho_et_al$condition_c <- build_contrast(ho_et_al$condition,
    "Low discrimination",
    "High discrimination")

my_model <-
    mdt_simple(data = ho_et_al,
        IV = condition_c,
        DV = hypodescent,
        M = linkedfate)

extract_models(my_model)
extract_tidy_models

Extracts models from a mediation object as a data frame

Description

When conducting a joint significant test, different models are fitted to the data. This function helps you access the models used in an object of class mediation_model.

Usage

extract_tidy_models(mediation_model)

Arguments

mediation_model

An object of class mediation_model.

Value

A data frame.

See Also

Other extract functions: extract_model(), extract_models()

Examples

data(ho_et_al)
ho_et_al$condition_c <- build_contrast(ho_et_al$condition, "Low discrimination", "High discrimination")

my_model <-
mdt_simple(data = ho_et_al,
  IV = condition_c,
  DV = hypodescent,
  M = linkedfate)

extract_tidy_models(my_model)
Description

A data set containing data from Experiment 3 from Ho, Kteiley, and Chen (2017). In this experiment, the authors hypothesized that presenting a text stating that Black-White biracials were discriminated against would lead Black participants to associate Black-White biracials more with their lower status parent group than their higher status parent group, according to the rule of hypodescent. In this experiment, the authors tested if this effect was mediated by the sense of linked fate between discriminated Black-White biracials and Black participants.

Note that this data set does not include the participants who were in the discrimination control condition in the study conducted by Ho, Kteiley and Chen (2017).

See mdt_simple and mdt_moderated to conduct a simple mediation or a moderated mediation analysis with this dataset.

Usage

data("ho_et_al")

Format

A data frame with 824 rows and 5 variables:

- **id**: An incremental index.
- **condition**: Experimental condition (High discrimination vs. Low discrimination).
- **sdo**: Score at an SDO scale.
- **linkedfate**: Score at an 8-item linked fate measure.
- **hypodescent**: Score at a 3-item measure of hypodescent.

References

Fits a moderated mediation model

**Description**

Given a data frame, a predictor (IV), an outcome (DV), a mediator (M), and a moderator (Mod) conducts a joint-significant test for moderated mediation (see Yzerbyt, Muller, Batailler, & Judd, 2018). You can learn about moderated mediation in vignette("moderated-mediation")

`add_index.moderated_mediation` computes the moderated mediation index. `compute_indirect_effect_for` is used to compute the indirect effect index for a specific value of the moderator.

**Usage**

```r
mdt_moderated(data, IV, DV, M, Mod)
```

**Arguments**

- `data`: A data frame containing the variables in the model.
- `IV`: An unquoted variable in the data frame which will be used as the independent variable.
- `DV`: An unquoted variable in the data frame which will be used as the dependent variable.
- `M`: An unquoted variable in the data frame which will be used as the mediator.
- `Mod`: An unquoted variable in the data frame which will be used as the moderator.

**Details**

With moderated mediation analysis, one tests whether the indirect effect of X on Y through M is moderated by Mod. The hypothesis behind this test is that X has an effect on M (a) which has an effect on Y (b), meaning that X has an indirect effect on Y through M.

Total moderation of the indirect effect of X on Y can be described as follows:

\[ c \times Mod = c' \times Mod + (a \times Mod) \times b + a \times (b \times Mod) \]

with \( c \times Mod \) the total moderation of the indirect effect, \( c' \times Mod \) the moderation of the direct effect, \( (a \times Mod) \times b \), the moderation of the indirect effect passing by the moderation of a, and \( a \times (b \times Mod) \), the moderation of the indirect effect passing by the moderation of b (see Models section; Muller et al., 2005).

Either both \( a \times Mod \) and \( b \) or both \( a \) and \( b \times Mod \) need to be simultaneously significant for a moderation of the indirect effect to be claimed (Muller et al., 2005).

**Value**

Returns an object of class "mediation_model".

An object of class "mediation_model" is a list containing at least the components:
**type**  A character string containing the type of model that has been conducted (e.g., "simple mediation").

**method**  A character string containing the approach that has been used to conduct the mediation analysis (usually "joint significance").

**params**  A named list of character strings describing the variables used in the model.

**paths**  A named list containing information on each relevant path of the mediation model.

**indirect_index**  A boolean indicating whether an indirect effect index has been computed or not. Defaults to FALSE. See add_index to compute mediation index.

**indirect_index_infos**  (Optional) An object of class "indirect_index". Appears when one applies add_index to an object of class "mediation_model".

**js_models**  A list of objects of class "lm". Contains every model relevant to joint-significance testing.

**data**  The original data frame that has been passed through data argument.

---

**Models**

In a moderated mediation model, three models will be used:

\[
Y_i = b_{40} + b_{41}X_i + b_{42}Mo_i + b_{43}XMo_i \\
M_i = b_{50} + b_{51}X_i + b_{52}Mo_i + b_{53}XMo_i \\
Y_i = b_{60} + c'_{61}X_i + b_{62}Mo_i + b_{63}Xmo_i + b_{64}Me_i + b_{65}MeMo_i
\]

with \(Y_i\), the outcome value for the \(i\)th observation, \(X_i\), the predictor value for the \(i\)th observation, \(Mo_i\), the moderator value for the \(i\)th observation, and \(M_i\), the mediator value for the \(i\)th observation.

Coefficients associated with \(a\), \(a \times Mod\), \(b\), \(b \times Mod\), \(c\), \(c \times Mod\), \(c'\), and \(c' \times Mod\), paths are respectively \(b_{51}\), \(b_{53}\), \(b_{64}\), \(b_{41}\), \(b_{43}\), \(b_{61}\), and \(b_{63}\) (see Muller et al., 2005).

---

**Variable coding**

Because joint-significance tests use linear models behind the scenes, variables involved in the model have to be numeric. mdt_simple will give an error if non-numeric variables are specified in the model.

If you need to convert a dichotomous categorical variable to a numeric one, please refer to the build_contrast function.

Note that variable coding is especially important in models with multiple predictors as is the case in the model used to conduct a joint-significance test of moderated mediation. Muller et al. (2005) recommend using variables that are either contrast-coded or centered. Using mdt_moderated with a DV, a mediator, or a moderator that is neither contrast-coded nor centered will give a warning message.
References


See Also

Other mediation models: mdt_simple(), mdt_within()

---

**mdt_simple**

*Joint-significance test for simple mediation*

**Description**

Given a data frame, a predictor (IV), an outcome (DV), and a mediator (M), conducts a joint-significant test for simple mediation (see Yzerbyt, Muller, Batailler, & Judd, 2018).

**Usage**

```r
mdt_simple(data, IV, DV, M)
```

**Arguments**

- `data`: A data frame containing the variables to be used in the model.
- `IV`: An unquoted numeric variable in the data frame which will be used as independent variable.
- `DV`: An unquoted numeric variable in the data frame which will be used as dependent variable.
- `M`: An unquoted numeric variable in the data frame which will be used as mediator.

**Details**

With simple mediation analysis, one is interested in finding if the effect of $X$ on $Y$ goes through a third variable $M$. The hypothesis behind this test is that $X$ has an effect on $M$ ($a$) that has an effect on $Y$ ($b$), meaning that $X$ has an indirect effect on $Y$ through $M$. The total effect of $X$ on $Y$ can be described as follows:

$$ c = c' + ab $$

with $c$ the total effect of $X$ on $Y$, $c'$ the direct of $X$ on $Y$, and $ab$ the indirect effect of $X$ on $Y$ through $M$ (see Models section).

To assess whether the indirect effect is different from the null, one has to assess the significance against the null for both $a$ (the effect of $X$ on $M$) and $b$ (effect of $M$ on $Y$ controlling for the effect of $X$). Both $a$ and $b$ need to be simultaneously significant for an indirect effect to be claimed (Cohen & Cohen, 1983; Yzerbyt, Muller, Batailler, & Judd, 2018).
Value

Returns an object of class "mediation_model". An object of class "mediation_model" is a list containing at least the components:

- **type**: A character string containing the type of model that has been conducted (e.g., "simple mediation").
- **method**: A character string containing the approach that has been used to conduct the mediation analysis (usually "joint significance").
- **params**: A named list of character strings describing the variables used in the model.
- **paths**: A named list containing information on each relevant path of the mediation model.
- **indirect_index**: A boolean indicating whether an indirect effect index has been computed or not. Defaults to FALSE. See add_index to compute mediation index.
- **indirect_index_infos**: (Optional) An object of class "indirect_index". Appears when one applies add_index to an object of class "mediation_model".
- **js_models**: A list of objects of class "lm". Contains every model relevant to joint-significance testing.
- **data**: The original data frame that has been passed through data argument.

Models

In a simple mediation model, three models will be fitted:

- \[ Y_i = b_{10} + c_{11}X_i \]
- \[ M_i = b_{20} + a_{21}X_i \]
- \[ Y_i = b_{30} + c'_{31}X_i + b_{32}M_i \]

with \( Y_i \), the outcome value for the \( i \)th observation, \( X_i \), the predictor value for the \( i \)th observation, and \( M_i \), the mediator value for the \( i \)th observation (Cohen & Cohen, 1983; Yzerbyt, Muller, Batailler, & Judd, 2018).

Coefficients associated with \( a \), \( b \), \( c \), and \( c' \) paths are respectively \( a_{21} \), \( b_{32} \), \( c_{11} \), and \( c'_{31} \).

Variable coding

Because joint-significance tests uses linear models behind the scenes, variables involved in the model have to be numeric. mdt_simple will give an error if non-numeric variables are specified in the model.

To convert a dichotomous categorical variable to a numeric one, please refer to the build_contrast function.

References


See Also

Other mediation models: `mdt_moderated()`, `mdt_within()`

Examples

```r
## fit a simple mediation model
data(ho_et_al)
ho_et_al$condition_c <- build_contrast(ho_et_al$condition, 
    "Low discrimination", 
    "High discrimination")

mdt_simple(data = ho_et_al, 
    IV = condition_c, 
    DV = hypodescent, 
    M = linkedfate)
```

---

**mdt_within**

*Joint-significance test for within-participant mediation*

Description

Given a data frame, a predictor (IV), an outcome (DV), a mediator (M), and a grouping variable (group) conducts a joint-significant test for within-participant mediation (see Yzerbyt, Muller, Batailler, & Judd, 2018).

Usage

```r
mdt_within(data, IV, DV, M, grouping, default_coding = TRUE)
```

Arguments

- `data`: a data frame containing the variables in the model.
- `IV`: an unquoted variable in the data frame which will be used as the independent variable.
- `DV`: an unquoted variable in the data frame which will be used as the dependent variable.
- `M`: an unquoted variable in the data frame which will be used as the mediator.
- `grouping`: an unquoted variable in the data frame which will be used as the grouping variable.
- `default_coding`: should the variable coding be the default? Defaults to `TRUE`. 
Details

With within-participant mediation analysis, one tests whether the effect of $X$ on $Y$ goes through a third variable $M$. The specificity of within-participant mediation analysis lies in the repeated measures design it relies on. With such a design, each sampled unit (e.g., participant) is measured on the dependent variable $Y$ and the mediator $M$ in the two conditions of $X$. The hypothesis behind this test is that $X$ has an effect on $M$ ($a$) which has an effect on $Y$ ($b$), meaning that $X$ has an indirect effect on $Y$ through $M$.

As with simple mediation, the total effect of $X$ on $Y$ can be conceptually described as follows:

$$c = c' + ab$$

with $c$ the total effect of $X$ on $Y$, $c'$ the direct of $X$ on $Y$, and $ab$ the indirect effect of $X$ on $Y$ through $M$ (see Models section).

To assess whether the indirect effect is different from the null, one has to assess the significance against the null for both $a$ (the effect of $X$ on $M$) and $b$ (effect of $M$ on $Y$ controlling for the effect of $X$). Both $a$ and $b$ need to be simultaneously significant for an indirect effect to be claimed (Judd, Kenny, & McClelland, 2001; Montoya & Hayes, 2011).

Value

Returns an object of class "mediation_model". An object of class "mediation_model" is a list containing at least the components:

- **type**: A character string containing the type of model that has been conducted (e.g., "simple mediation").
- **method**: A character string containing the approach that has been used to conduct the mediation analysis (usually "joint significance").
- **params**: A named list of character strings describing the variables used in the model.
- **paths**: A named list containing information on each relevant path of the mediation model.
- **indirect_index**: A boolean indicating whether an indirect effect index has been computed or not. Defaults to FALSE. See **add_index** to compute mediation index.
- **indirect_index_infos**
  (Optional) An object of class "indirect_index". Appears when one applies **add_index** to an object of class "mediation_model".
- **js_models**: A list of objects of class "lm". Contains every model relevant to joint-significance testing.
- **data**: The original data frame that has been passed through data argument.

Models

For within-participant mediation, three models will be fitted:

- $Y_{2i} - Y_{1i} = c_{11}$
- $M_{2i} - M_{1i} = a_{21}$
\[ Y_{2i} - Y_{1i} = c'_{31} + b_{32}(M_{2i} - M_{1i}) + d_{33}[0.5(M_{1i} + M_{2i}) - 0.5(M_1 + M_2)] \]

with \( Y_{2i} - Y_{1i} \) the difference score between DV conditions for the outcome variable for the \( i \)th observation, \( M_{2i} - M_{1i} \) the difference score between DV conditions for the mediator variable for the \( i \)th observation, \( M_{1i} + M_{2i} \) the sum of mediator variables values for DV conditions for the \( i \)th observation, and \( M_1 + M_2 \) the mean sum of mediator variables values for DV conditions across observations (see Montoya & Hayes, 2011).

Coefficients associated with \( a, b, c, \) and \( c' \) paths are respectively \( a_{21}, b_{32}, c_{11}, \) and \( c'_{31} \).

**Data formatting**

To be consistent with other mdt_* family functions, `mdt_within` takes a long-format data frame as `data` argument. With this kind of format, each sampled unit has two rows, one for the first within-participant condition and one for the second within-participant condition. In addition, each row has one observation for the outcome and one observation for the mediator (see `dohle_siegrist` for an example).

Because such formatting is not the most common among social scientists interested in within-participant mediation, JSmediation contains the `mdt_within_wide` function which handles wide-formatted data input (but is syntax-inconsistent with other mdt_* family functions).

**Variable coding**

Models underlying within-participant mediation use difference scores as DV (see Models section). Because the function input does not allow the user to specify how the difference scores should be computed, `mdt_within` has a default coding.

`mdt_within`’s default behavior is to compute the difference score so the total effect (the effect of \( X \) on \( Y \)) will be positive and compute the other difference scores accordingly. That is, if `mdt_within` has to use \( Y_{2i} - Y_{1i} \) (instead of \( Y_{1i} - Y_{2i} \)) so that \( c_{11} \) is positive, it will use \( M_{2i} - M_{1i} \) (instead of \( M_{1i} - M_{2i} \)) in the other models.

User can choose to have a negative total effect by using the `default_coding` argument.

Note that DV and M have to be numeric.

**References**


**See Also**

Other mediation models: `mdt_moderated()`, `mdt_simple()`
**Description**

Given a data frame, a predictor (IV), an outcome (DV), a mediator (M), and a grouping variable (group) conducts a joint-significant test for within-participant mediation (see Yzerbyt, Muller, Batailler, & Judd, 2018).

**Usage**

```r
mdt_within_wide(data, DV_A, DV_B, M_A, M_B)
```

**Arguments**

- `data` a data frame containing the variables in the model.
- `DV_A` an unquoted numeric variable in the data frame which will be used as the dependent variable value for the "A" independent variable condition.
- `DV_B` an unquoted numeric variable in the data frame which will be used as the dependent variable value for the "B" independent variable condition.
- `M_A` an unquoted numeric variable in the data frame which will be used as the mediator variable value for the "A" independent variable condition.
- `M_B` an unquoted numeric variable in the data frame which will be used as the mediator variable value for the "b" independent variable condition.

**Details**

With within-participant mediation analysis, one tests whether the effect of X on Y goes through a third variable M. The specificity of within-participant mediation analysis lies in the repeated measures design it relies on. With such a design, each sampled unit (e.g., participant) is measured on the dependent variable Y and the mediator M in the two conditions of X. The hypothesis behind this test is that X has an effect on M (a) which has an effect on Y (b), meaning that X has an indirect effect on Y through M.

As with simple mediation, the total effect of X on Y can be conceptually described as follows:

\[ c = c' + ab \]

with c the total effect of X on Y, c’ the direct of X on Y, and ab the indirect effect of X on Y through M (see Models section).

To assess whether the indirect effect is different from the null, one has to assess the significance against the null for both a (the effect of X on M) and b (effect of M on Y controlling for the effect of X). Both a and b need to be simultaneously significant for an indirect effect to be claimed (Judd, Kenny, & McClelland, 2001; Montoya & Hayes, 2011).
**Value**

Returns an object of class "mediation_model".

An object of class "mediation_model" is a list containing at least the components:

- **type** A character string containing the type of model that has been conducted (e.g., "simple mediation").
- **method** A character string containing the approach that has been used to conduct the mediation analysis (usually "joint significance").
- **params** A named list of character strings describing the variables used in the model.
- **paths** A named list containing information on each relevant path of the mediation model.
- **indirect_index** A boolean indicating whether an indirect effect index has been computed or not. Defaults to FALSE. See `add_index` to compute mediation index.
- **indirect_index_infos** (Optional) An object of class "indirect_index". Appears when one applies `add_index` to an object of class "mediation_model".
- **js_models** A list of objects of class "lm". Contains every model relevant to joint-significance testing.
- **data** The original data frame that has been passed through `data` argument.

**Data formatting**

To be consistent with other `mdt_*` family functions, `mdt_within` takes a long-format data frame as `data` argument. With this kind of format, each sampled unit has two rows, one for the first within-participant condition and one for the second within-participant condition. In addition, each row has one observation for the outcome and one observation for the mediator (see `dohle_siegrist` for an example).

Because such formatting is not the most common among social scientists interested in within-participant mediation, `JSmediation` contains the `mdt_within_wide` function which handles wide-formatted data input (but is syntax-inconsistent with other `mdt_*` family functions).

**Variable coding**

Models underlying within-participant mediation use difference scores as DV (see Models section). `mdt_within_wide` uses $M_A - M_B$ and $DV_A - DV_B$ in these models.

**Models**

For within-participant mediation, three models will be fitted:

- $Y_{2i} - Y_{1i} = c_{11}$
- $M_{2i} - M_{1i} = a_{21}$
- $Y_{2i} - Y_{1i} = c_{31} + b_{32}(M_{2i} - M_{1i}) + d_{33}[0.5(M_{1i} + M_{2i}) - 0.5(M_1 + M_2)]$
with $Y_{2i} - Y_{1i}$ the difference score between DV conditions for the outcome variable for the $i$th observation, $M_{2i} - M_{1i}$ the difference score between DV conditions for the mediator variable for the $i$th observation, $M_{1i} + M_{2i}$ the sum of mediator variables values for DV conditions for the $i$th observation, and $M_{1i} + M_{2i}$ the mean sum of mediator variables values for DV conditions across observations (see Montoya & Hayes, 2011).

Coefficients associated with $a$, $b$, $c$, and $c'$ paths are respectively $a_{21}$, $b_{32}$, $c_{11}$, and $c'_{31}$.

References


---

plot_assumptions

Returns diagnostic plots for the linear model used in a mediation

Description

When conducting a joint-significant test, different models are fitted to the data. This function returns diagnostic plots for each of the model used in the mediation model. check_assumptions_plot uses the performance and see packages behind the scenes to provide the different plots.

This function is best used in an interactive context.

Usage

```r
plot_assumptions(
  mediation_model,
  tests = c("normality", "heteroscedasticity", "outliers")
)
```

Arguments

- `mediation_model`  
  An object of class `mediation_model`.
- `tests`  
  A character vector indicating which test to run. Supported test includes "normality", "heteroscedasticity", and "outliers"

Value

Invisibly returns an object of class `mediation_model`.
See Also

Other assumption checks: `check_assumptions()`

Examples

data(ho_et_al)

ho_et_al$condition_c <- build_contrast(ho_et_al$condition,
                                        "Low discrimination",
                                        "High discrimination")

my_model <-
    mdt_simple(data = ho_et_al,
                IV = condition_c,
                DV = hypodescent,
                M = linkedfate)

plot_assumptions(my_model)

print.mediation_model

Print method for object of class `mediation_model`

Description

Print a summary for a mediation model represented by a `mediation_model` object.

Usage

```r
## S3 method for class 'mediation_model'
print(x, digits = 3, ...)
```

Arguments

- `x` An object of class `mediation_model`.
- `digits` How many significant digits are to be used for numerics.
- `...` Further arguments.
standardize_variable  

Standardize variables in a data set.

Description

standardize_variable() standardizes the selected columns in a data frame using base::scale(). By default, this function overwrites the column to be scaled. Use the suffix argument to avoid this behavior.

standardize_variable() and standardise_variable() are synonyms.

Usage

standardize_variable(data, cols = dplyr::everything(), suffix = NULL)
standardise_variable(data, cols = dplyr::everything(), suffix = NULL)

Arguments

data  A data frame containing the variables to standardize.
cols  <tidy-select> Columns to standardize. Defaults to dplyr::everything().
suffix  A character suffix to be added to the scaled variables names. When suffix is set to NULL, the standardize_variable() function will overwrite the scaled variables. Defaults to NULL.

Value

A data frame with the standardized columns.

standardize_variable and grouped_df

Note that standardize_variable ignores grouping. Meaning that if you call this function on a grouped data frame (see dplyr::grouped_df), the overall variables' mean and standard deviation will be used for the standardization.

Examples

ho_et_al %>%
  standardize_variable(sdo)

ho_et_al %>%
  standardize_variable(c(sdo, linkedfate), suffix = "scaled")
Index

* assumption checks
  check_assumptions, 8
  plot_assumptions, 26
* datasets
  dohle_siegrist, 11
  dohle_siegrist_wide, 12
  ho_et_al, 16
* extract functions
  extract_model, 13
  extract_models, 14
  extract_tidy_models, 15
* mediation models
  mdt_moderated, 17
  mdt_simple, 19
  mdt_within, 21
  add_index, 2, 2, 9, 18, 20, 22, 25
  add_index.moderated_mediation, 3, 17
  add_index.simple_mediation, 2, 4
  add_index.within_participant_mediation, 5
  apastylr, 6
  base::scale(), 28
  build_contrast, 7, 18, 20
  check_assumptions, 8, 27
  check_heteroscedasticity, 8
  check_normality, 8
  check_outliers, 8
  compute_indirect_effect_for, 9, 17
  display_models, 10
  dohle_siegrist, 11, 23, 25
  dohle_siegrist_wide, 12
  dplyr::everything(), 28
  dplyr::grouped_df, 28
  extract_model, 13, 14, 15
  extract_models, 13, 14, 15
  extract_tidy_models, 13, 14, 15
  ho_et_al, 16
  lm, 7
  mdt_moderated, 3, 9, 10, 16, 17, 21, 23
  mdt_simple, 2, 4, 16, 19, 19, 23
  mdt_within, 5, 11, 19, 21, 21, 23, 25
  mdt_within_wide, 5, 12, 23, 24, 25
  plot_assumptions, 8, 26
  print.mediation_model, 27
  scale, 7
  standardise_variable
    (standardize_variable), 28
  standardize_variable, 28

29