Package 'powerNLSEM'

May 9, 2026

Type Package

Title Simulation-Based Power Estimation (MSPE) for Nonlinear SEM

Version 0.1.2

Author Julien Patrick Irmer [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0002-7544-6483>)

Maintainer Julien Patrick Irmer <jpirmer@gmail.com>

Description Model-implied simulation-based power estimation (MSPE) for nonlinear (and linear) SEM, path analysis and regression analysis. A theoretical framework is used to approximate the relation between power and sample size for given type I error rates and effect sizes. The package offers an adaptive search algorithm to find the optimal N for given effect sizes and type I error rates. Plots can be used to visualize the power relation to N for different parameters of interest (POI). Theoretical justifications are given in Irmer et al. (2024a) <doi:10.31219/osf.io/pe5bj> and detailed description are given in Irmer et al. (2024b) <doi:10.3758/s13428-024-02476-3>.

License GPL-3

URL https://github.com/jpirmer/powerNLSEM

BugReports https://github.com/jpirmer/powerNLSEM/issues

Depends ggplot2, stats, utils

Imports crayon, lavaan (>= 0.6.16), mvtnorm, numDeriv, pbapply, rlang (>= 1.1.0), stringr

Suggests knitr, MplusAutomation (>= 0.7-2), rmarkdown, semTools, simsem

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.1

NeedsCompilation no

Repository CRAN

Date/Publication 2024-09-27 11:40:02 UTC

Contents

`FSR` *Factor Score Regression approach*

Description

Factor Score Regression approach

Usage

```
FSR(lavModel_Analysis, data, FSmethod = "SL", data_transformations = NULL)
```

Arguments

`lavModel_Analysis`	the lavModel_Analysis object
`data`	set to fit
`FSmethod`	Method to be used to extract factor scores. Default to `"SL"` for the Skrondal and Laake approach that uses regression (`"regression"`) factor scores for the independendent variables and `"Bartlett"` factor scores for the dependent variables.
`data_transformations`	Data transformations

Value

Returns a `data.frame` that includes parameter estimates estimated using FSR.

References

Similar to: Ng, J. C. K., & Chan, W. (2020). Latent moderation analysis: A factor score approach. *Structural Equation Modeling: A Multidisciplinary Journal, 27*(4), 629–648. doi:10.1080/10705511.2019.1664304.

Skrondal, A., & Laake, P. (2001). Regression among factor scores. *Psychometrika, 66*(4), 563-575. doi:10.1007/BF02296196

`LMS` *Latent moderated strctural equations by Klein and Moosbrugger (2000), the ML approach to nonlinear SEM*

Description

Latent moderated strctural equations by Klein and Moosbrugger (2000), the ML approach to nonlinear SEM

Usage

```
LMS(
  lavModel_Analysis,
  data,
  data_transformations = NULL,
  prefix = 1,
  pathLMS = tempdir(),
  algorithm = "INTEGRATION"
)
```

Arguments

`lavModel_Analysis`	the lavModel_Analysis object
`data`	set to fit
`data_transformations`	Object containing info on possible data transformations.
`prefix`	an arbitrary prefix for the data set. This prevents issues when using parallelization and Mplus.
`pathLMS`	path where (temporal) data and scripts for running LMS using Mplus are stored (using `MplusAutomation`). Default to `NULL`, then `tempdir()` is used.
`algorithm`	algorithm to use. Default to INTEGRATION.

Value

Returns a `data.frame` that includes parameter estimates estimated using LMS.

References

Klein, A. G., & Moosbrugger, H. (2000). Maximum likelihood estimation of latent interaction effects with the LMS method. *Psychometrika, 65*(4), 457–474. doi:10.1007/BF02296338

`plot.powerNLSEM` *plot powerNLSEM object*

Description

plot powerNLSEM object

Usage

```
## S3 method for class 'powerNLSEM'
plot(
  x,
  test = NULL,
  plot = "power_model",
  power_modeling_method = NULL,
  se = FALSE,
  power_aim = NULL,
  alpha = NULL,
  alpha_power_modeling = NULL,
  min_num_bins = 10,
  defaultgg = FALSE,
  ...
)
```

Arguments

`x`	object of class powerNLSEM
`test`	Should the parameter be tested with a directed hypothesis (onesided) or with an undirected hypothesis (twosided, also equivalent to Wald-Test for single parameter). Default to `NULL`, then the same as in fitted `powerNLSEM` object in `x` is used.
`plot`	Character indicating what type of plot to create. Default to `"power_model"`, referencing to the prediction of significant parameters using the model specified in `power_modeling_method`.
`power_modeling_method`	Character indicating the power modeling method used. This is only relevant when `plot = "power_model"` is used. Default to `NULL`, indicating to use the same power modeling method as was used in the `powerNLSEM` function.
`se`	Logical indicating to use confidence intervals based on normal approximation using the standard errors. Default to `FALSE`.

power_aim — Power level to be included into the plot with respective N. If NULL the same power level as in the powerNLSEM function will be used. If set to 0 no power level and corresponding N will be plotted. Default to NULL, indicating to use the same power modeling method as was used in the powerNLSEM function.

alpha — Alpha value used for confidence intervals, when se = TRUE. Default to NULL, indicating to use the same alpha as was used in the powerNLSEM function. This does not influence the significance decision, although same alpha is used per default.

alpha_power_modeling — Type I-error rate for confidence band around predicted power rate. Used to ensure that the computed N keeps the desired power value (with the given Type I-error rate alpha_power_modeling divided by 2). If set to 1, no confidence band is used. Default to .05.

min_num_bins — minimal number of bins used for aggregating results. Default to 10.

defaultgg — Logical to return default ggplot object. Default to FALSE, which returns theme_minimal and other changes in theme.

... — Additional arguments passed on to the plot function.

Value

Returns ggplot object of the type specified in plot.

Examples

```
# write model in lavaan syntax
model <- "
# measurement models
          X =~ 1*x1 + 0.8*x2 + 0.7*x3
          Y =~ 1*y1 + 0.85*y2 + 0.78*y3
          Z =~ 1*z1 + 0.9*z2 + 0.6*z3

# structural models
          Y ~ 0.3*X + .2*Z +  .2*X:Z

# residual variances
         Y~~.7975*Y
         X~~1*X
         Z~~1*Z

# covariances
         X~~0.5*Z

# measurement error variances
         x1~~.1*x1
         x2~~.2*x2
         x3~~.3*x3
         z1~~.2*z1
         z2~~.3*z2
         z3~~.4*z3
         y1~~.5*y1
```

```
        y2~~.4*y2
        y3~~.3*y3
"
# run model-implied simulation-based power estimation
# for the effects: c("Y~X", "Y~Z", "Y~X:Z")
Result_Power <- powerNLSEM(model = model, POI = c("Y~X", "Y~Z", "Y~X:Z"),
                           method = "UPI", search_method = "adaptive",
                           steps = 10, power_modeling_method = "probit",
                           R = 1000, power_aim = .8, alpha = .05,
                           alpha_power_modeling = .05,
                           CORES = 1, seed = 2024)

Result_Power
plot(Result_Power)
```

---

powerNLSEM *powerNLSEM function*

Description

powerNLSEM function

Usage

```
powerNLSEM(
  model,
  POI,
  method,
  test = "onesided",
  power_modeling_method = "probit",
  search_method = "adaptive",
  R = 2000,
  power_aim = 0.8,
  alpha = 0.05,
  alpha_power_modeling = 0.05,
  CORES = max(c(parallel::detectCores() - 2, 1)),
  verbose = TRUE,
  seed = NULL,
  ...
)
```

Arguments

model	Model in lavaan syntax. See documentation for help and examples.
POI	Parameter Of Interest as a vector of strings. Must be in lavaan-syntax without any spaces. Nonlinear effects should have the same ordering as in model.

method	Method used to fit to the data. Implemented methods are "LMS" (Klein & Moosbrugger, 2000) (requires an installation of Mplus and the MplusAutomation pacakge), "UPI" (Kelava & Brandt, 2009, Marsh et al., 2004) for the unconstrained product indicator approach, "FSR" (Ng and Chan, 2020) for the naïve factor score approach, and "SR", for using scale means (i.e., scale regression/path modeling).
test	Should the parameter be tested with a directed hypothesis (onesided) or with an undirected hypothesis (twosided, also equivalent to Wald-Test for single parameter). Default to "onesided".
power_modeling_method	Power modeling method used to model significant parameter estimates. Default to "probit" indicating glm with probit link function with sqrt(n) as predictor. Alternative is "logit".
search_method	String stating the search method. Default to "adaptive" (synonyme is "smart"). Alternative is "bruteforce".
R	Total number of models to be fitted. Higher number results in higher precision and longer runtime. Default to 2000.
power_aim	Minimal power value to approximate. Default to .8.
alpha	Type I-error rate for significance decision. Default to .05.
alpha_power_modeling	Type I-error rate for confidence band around predicted power rate. Used to ensure that the computed N keeps the desired power value (with the given Type I-error rate alpha_power_modeling divided by 2). If set to 1, no confidence band is used. Default to .05.
CORES	Number of cores used for parallelization. Default to number of available cores - 2.
verbose	Logical whether progress should be printed in console. Default to TRUE.
seed	Seed for replicability. Default to NULL, then a seed is drawn at random, which will also be saved in the output.
...	Additional arguments passed on to the search functions.

Value

Returns an list object of class powerNLSEM.

References

Klein, A. G., & Moosbrugger, H. (2000). Maximum likelihood estimation of latent interaction effects with the LMS method. *Psychometrika, 65*(4), 457–474. doi:10.1007/BF02296338

Kelava, A., & Brandt, H. (2009). Estimation of nonlinear latent structural equation models using the extended unconstrained approach. *Review of Psychology, 16*(2), 123–132.

Lin, G. C., Wen, Z., Marsh, H. W., & Lin, H. S. (2010). Structural equation models of latent interactions: Clarification of orthogonalizing and double-mean-centering strategies. *Structural Equation Modeling, 17*(3), 374–391. doi:10.1080/10705511.2010.488999

Little, T. D., Bovaird, J. A., & Widaman, K. F. (2006). On the merits of orthogonalizing powered and product terms: Implications for modeling interactions among latent variables. *Structural Equation Modeling, 13*(4), 497–519. doi:10.1207/s15328007sem1304_1

Marsh, H. W., Wen, Z. & Hau, K. T. (2004). Structural equation models of latent interactions: Evaluation of alternative estimation strategies and indicator construction. *Psychological Methods, 9*(3), 275–300. doi:10.1037/1082989X.9.3.275

Ng, J. C. K., & Chan, W. (2020). Latent moderation analysis: A factor score approach. *Structural Equation Modeling: A Multidisciplinary Journal, 27*(4), 629–648. doi:10.1080/10705511.2019.1664304.

Irmer, J. P., Klein, A. G., & Schermelleh-Engel, K. (2024a). A General Model-Implied Simulation-Based Power Estimation Method for Correctly and Misspecfied Models: Applications to Nonlinear and Linear Structural Equation Models. *Behavior Research Methods.* doi:10.31219/osf.io/pe5bj

Irmer, J. P., Klein, A. G., & Schermelleh-Engel, K. (2024b). Estimating Power in Complex Nonlinear Structural Equation Modeling Including Moderation Effects: The `powerNLSEM` R-Package. *Behavior Research Methods.* doi:10.3758/s13428024024763

See Also

For further details for specific uses see corresponding functions: `power_search()` for all inputs possible, `UPI()` for specifics for the unconstrained product indicator approach, `LMS()` for the latent moderated structural equations approach, `FSR()` for factor score approaches, `SR()` for scale regression approaches.

Examples

```
# write model in lavaan syntax
model <- "
# measurement models
          X =~ 1*x1 + 0.8*x2 + 0.7*x3
          Y =~ 1*y1 + 0.85*y2 + 0.78*y3
          Z =~ 1*z1 + 0.9*z2 + 0.6*z3

# structural models
          Y ~ 0.3*X + .2*Z +  .2*X:Z

# residual variances
         Y~~.7975*Y
         X~~1*X
         Z~~1*Z

# covariances
         X~~0.5*Z

# measurement error variances
         x1~~.1*x1
         x2~~.2*x2
         x3~~.3*x3
         z1~~.2*z1
         z2~~.3*z2
         z3~~.4*z3
         y1~~.5*y1
```

```
        y2~~.4*y2
        y3~~.3*y3
"
# run model-implied simulation-based power estimation
# for the effects: c("Y~X", "Y~Z", "Y~X:Z")
Result_Power <- powerNLSEM(model = model, POI = c("Y~X", "Y~Z", "Y~X:Z"),
                           method = "UPI", search_method = "adaptive",
                           steps = 10, power_modeling_method = "probit",
                           R = 1000, power_aim = .8, alpha = .05,
                           alpha_power_modeling = .05,
                           CORES = 1, seed = 2024)

Result_Power
```

---

power_search *Search function to find N for desired power*

Description

The function that initializes the search process. The powerNLSEM function actually is a wrapper function for power_search.

Usage

```
power_search(
  POI,
  method,
  lavModel,
  lavModel_Analysis,
  data_transformations,
  search_method,
  power_modeling_method,
  R = 1000,
  power_aim = 0.8,
  alpha = 0.05,
  alpha_power_modeling = 0.05,
  CORES,
  verbose,
  Ns = NULL,
  N_start = nrow(lavModel[lavModel$op != "~1", ]) * 10,
  distRj = "increasing",
  steps = 10,
  nlb = nrow(lavModel[lavModel$op != "~1", ]) * 5,
  switchStep = round(steps/2),
  FSmethod = "SL",
  test = "onesided",
```

```
  matchPI = TRUE,
  PIcentering = "doubleMC",
  liberalInspection = FALSE,
  constrainRelChange = TRUE,
  seeds,
  pathLMS = tempdir()
)
```

Arguments

POI	Parameter Of Interest as a vector of strings. Must be in lavaan-syntax without any spaces. Nonlinear effects should have the same ordering as in model.
method	Method used to fit to the data. Can be LMS or UPI.
lavModel	lavModel object describing the model.
lavModel_Analysis	lavModel object containg the parameters to be estimated.
data_transformations	Object containing info on data transformations.
search_method	String stating the search method. Default to "adaptive" (synonyme is "smart"). Alternative is "bruteforce".
power_modeling_method	Power modeling method used to model significant parameter estimates. Default to "probit" indicating glm with probit link function with sqrt(n) as predictor. Alternative is "logit".
R	Total number of models to be fitted. Higher number results in higher precision and longer runtime.
power_aim	Minimal power value to approximate. Default to .8.
alpha	Type I-error rate for significance decision. Default to .05.
alpha_power_modeling	Type I-error rate for confidence band around predicted power rate. Used to ensure that the computed N keeps the desired power value (with the given Type I-error rate alpha_power_modeling divided by 2). If set to 1, no confidence band is used. Default to .05.
CORES	Number of cores used for parallelization. Default to number of available cores - 2.
verbose	Logical whether progress should be printed in console. Default to TRUE.
Ns	Sample sizes used in power estimation process. Default to NULL.
N_start	Starting sample size for smart algorithm. Default to 10*nrow(lavModel[lavModel$op != "~1",]) (10 times the number of parameters, excluding the mean structure, without the generation of e.g., factor scores or product indicators).
distRj	Indicator how the samples sizes should be used in the steps of the smart algorithm: "u" for many to few to many, "increasing" for increasing replications and "even" for evenly distributed replications across steps. Default to "u".
steps	Steps used in search_method = "smart", i.e., the smart algorithm. This is ignored if bruteforce is used. Default to 10.

nlb — Lower bound of N used in search. Default to `5*nrow(lavModel[lavModel$op != "~1", ])` (5 times the number of parameters, excluding the mean structure, in the model without the generation of e.g., factor scores or product indicators), however, some methods can deal with much smaller sample sizes so this can be adjusted. The rule of thumb of 5 times number of parameters is motivated by Wolf et al. (2013)

switchStep — Steps after which smart search method changes from exploration to exploitation. Default to `round(steps/2)`. Exploration phase searches for the interval for N so that the resulting power is within `[.15, .85]` since the power curve is steepest at .5 and becomes less step towards plus/min `Inf`. Exploitation phase searches for an interval for N around the `power_aim` argument which shrinks from plus/minus .1 to .01. If `swicthStep = Inf`, then only exploration is used. If `switchStep` is used then the search process is reset at that point, which results in a new estimation in the bounds of the interval of N independent of the previous ones which might be restricted in change (see also argument(`constrainRelChange`).

FSmethod — Method to be used to extract factor scores. Default to `"SL"` for the Skrondal and Laake approach that uses regression (`"regression"`) factor scores for the independendent variables and `"Bartlett"` factor scores for the dependent variables.

test — Should the parameter be tested with a directed hypothesis (onesided) or with an undirected hypothesis (twosided, also equivalent to Wald-Test for single parameter). Default to `"onesided"`.

matchPI — Logical passed to `semTools::indProd` in order to compute the product indicators: Specify TRUE to use match-paired approach (Marsh, Wen, & Hau, 2004). If FALSE, the resulting products are all possible products. Default to `TRUE`. The observations are matched by order given when specifying the measurement model.

PIcentering — String indicating which method of centering should be used when constructing product indicators. String is converted to the arguments `meanC`, `doubleMC`, and `residualMC`, of the `semTools::indProd` function. Default to `"doubleMC"` for double mean centering the resulting products (Lin et. al., 2010). Use `"meanC"` for mean centering the main effect indicator before making the products or `"residualC"` for residual centering the products by the main effect indicators (Little, Bovaird, & Widaman, 2006). `"none"` or any other input than the previously described results in no centering (use with caution!).

liberalInspection — Logical whether the inspection of estimation truthworthiness should be very liberal (i.e., allowing for non-positive definite Hessians in standard error estimation or non-positive residual covariance matrices or latent covariance matrices). Default to `FALSE`. Being liberal is not adviced and should be checked for a single data set!

constrainRelChange — Logical whether the change in the bounds of the interval for N using the smart algorithm should be constrained. This prevents divergence (which is especially an issue for small effect sizes and small `R`) but results in biased estimates if the number of steps is too small. Default to `TRUE`.

seeds — Seeds for reproducibility.

pathLMS	path where (temporal) data and scripts for running LMS using Mplus are stored (using MplusAutomation). Default to NULL, then tempdir() is used.

Value

Returns a list that includes the results on model-implied simulation-based power estimation.

References

Wolf, E. J., Harrington, K. M., Clark, S. L., & Miller, M. W. (2013). Sample Size Requirements for Structural Equation Models: An Evaluation of Power, Bias, and Solution Propriety. *Educational and Psychological Measurement, 76*(6), 913–934. doi:10.1177/0013164413495237

Irmer, J. P., Klein, A. G., & Schermelleh-Engel, K. (2024). *Behavior Research Methods, 0*(00), Advance Online Publication.

---

print.powerNLSEM *print powerNLSEM objects*

Description

print powerNLSEM objects

Usage

```
## S3 method for class 'powerNLSEM'
print(x, ...)
```

Arguments

x	object of class powerNLSEM
...	Additional parameters for print

Value

powerNLSEM object

print.summary.powerNLSEM *print summary for powerNLSEM objects*

Description

print summary for powerNLSEM objects

Usage

```
## S3 method for class 'summary.powerNLSEM'
print(x, ...)
```

Arguments

x	Prints the summary of a powerNLSEM object. x must be of class "summary.powerNLSEM".
...	Further arguments to use in print.

Value

Prints output of summmary of powerNLSEM object into the console (objects of class summary.powerNLSEM), but does not change object itself.

reanalyse.powerNLSEM *Reanalyse powerNLSEM object*

Description

Reanalyse powerNLSEM object

Usage

```
reanalyse.powerNLSEM(
  out,
  test = NULL,
  powerLevels = NULL,
  power_modeling_method = NULL,
  alpha = NULL,
  alpha_power_modeling = NULL
)
```

Arguments

out	object of class powerNLSEM
test	Should the parameter be tested with a directed hypothesis (onesided) or with an undirected hypothesis (twosided, also equivalent to Wald-Test for single parameter). Default to NULL, then the same as in fitted powerNLSEM object is used.
powerLevels	Power levels for which the desired sample sizes should be computed. Needs to be a vector. Default to NULL indicating to use same power rate used in powerNLSEM object.
power_modeling_method	Character indicating the power modeling method used. Default to NULL, indicating to use the same power modeling method as was used in the powerNLSEM object.
alpha	Type I-error rate for significance decision. Default to .05.
alpha_power_modeling	Type I-error rate for confidence band around predicted power rate. Used to ensure that the computed N keeps the desired power value (with the given Type I-error rate alpha_power_modeling divided by 2). If set to 1, no confidence band is used. Default to .05.

Value

Returns list of desired sample sizes per effect for each powerLevel. Nall refers to the sample size required per power level for all coefficients. Npower is a matrix containing the desired sample sizes per effect for every power level.

---

simulateNLSEM *simulate data from lavModel object*

Description

simulate data from lavModel object

Usage

```
simulateNLSEM(
  n,
  lavModel,
  appendLVs = FALSE,
  lavModel_attributes = NULL,
  matrices = NULL,
  seed = NULL
)
```

Arguments

n	sample size
lavModel	lavModel object
appendLVs	logical whether latent variables should be observed. Default to FALSE. (For developmental purposes)
lavModel_attributes	attributes of the lavModel object. If NULL, this is computed from lavModel. Default to NULL.
matrices	computed matrices for simulation. If NULL, this is computed from lavModel and lavModel_attributes. Default to NULL.
seed	a seed for reproducability. Default to NULL.

Value

Returns a data.frame of a simulated NLSEM.

SR *Scale Regression approach*

Description

Scale Regression approach

Usage

```
SR(lavModel_Analysis, data, data_transformations = NULL)
```

Arguments

lavModel_Analysis	the lavModel_Analysis object
data	set to fit
data_transformations	Data transformations

Value

Returns a data.frame that includes parameter estimates estimated using SR.

summary.powerNLSEM *Summary function for powerNLSEM objects*

Description

Summary function for powerNLSEM objects

Usage

```
## S3 method for class 'powerNLSEM'
summary(object, test = NULL, alpha = NULL, ...)
```

Arguments

object	Result of powerNLSEM function estimating the MSPE. object must be of class "powerNLSEM".
test	Should the parameter be tested with a directed hypothesis (onesided) or with an undirected hypothesis (twosided, also equivalent to Wald-Test for single parameter). Default to NULL (if NULL, test of the original MSPE is used).
alpha	Type I-error rate for significance decision. Default to NULL (if NULL, alpha of the original MSPE is used).
...	Further arguments to use in summary.

Value

summary of powerNLSEM object

Examples

```
# write model in lavaan syntax
model <- "
# measurement models
          X =~ 1*x1 + 0.8*x2 + 0.7*x3
          Y =~ 1*y1 + 0.85*y2 + 0.78*y3
          Z =~ 1*z1 + 0.9*z2 + 0.6*z3

# structural models
          Y ~ 0.3*X + .2*Z +  .2*X:Z

# residual variances
         Y~~.7975*Y
         X~~1*X
         Z~~1*Z

# covariances
         X~~0.5*Z

# measurement error variances
```

```
          x1~~.1*x1
          x2~~.2*x2
          x3~~.3*x3
          z1~~.2*z1
          z2~~.3*z2
          z3~~.4*z3
          y1~~.5*y1
          y2~~.4*y2
          y3~~.3*y3
"
# run model-implied simulation-based power estimation
# for the effects: c("Y~X", "Y~Z", "Y~X:Z")
Result_Power <- powerNLSEM(model = model, POI = c("Y~X", "Y~Z", "Y~X:Z"),
                           method = "UPI", search_method = "adaptive",
                           steps = 10, power_modeling_method = "probit",
                           R = 1000, power_aim = .8, alpha = .05,
                           alpha_power_modeling = .05,
                           CORES = 1, seed = 2024)

Result_Power
summary(Result_Power)
```

UPI *Unconstrained Product Indicator approach by Marsh et al. (2004), with extensions by Kelava and Brandt (2009)*

Description

Unconstrained Product Indicator approach by Marsh et al. (2004), with extensions by Kelava and Brandt (2009)

Usage

```
UPI(
  lavModel_Analysis,
  data,
  data_transformations = NULL,
  matchPI = TRUE,
  PIcentering = "doubleMC",
  liberalInspection = FALSE
)
```

Arguments

lavModel_Analysis	the lavModel_Analysis object
data	set to fit

`data_transformations`
: Data transformations

`matchPI`
: Logical passed to `semTools::indProd` in order to compute the product indicators: Specify TRUE to use match-paired approach (Marsh, Wen, & Hau, 2004). If FALSE, the resulting products are all possible products. Default to `TRUE`. The observations are matched by order given when specifying the measurement model.

`PIcentering`
: String indicating which method of centering should be used when constructing product indicators. String is converted to the arguments `meanC`, `doubleMC`, and `residualMC`, of the `semTools::indProd` function. Default to `"doubleMC"` for double mean centering the resulting products (Lin et. al., 2010). Use `"meanC"` for mean centering the main effect indicator before making the products or `"residualC"` for residual centering the products by the main effect indicators (Little, Bovaird, & Widaman, 2006). `"none"` or any other input than the previously described results in no centering (use with caution!).

`liberalInspection`
: Logical whether the inspection of estimation truthworthiness should be very liberal (i.e., allowing for non-positive definite Hessians in standard error estimation or non-positive residual covariance matrices or latent covariance matrices). Default to `FALSE`. Being liberal is not adviced and should be checked for a single data set!

Value

Returns a `data.frame` that includes parameter estimates estimated using UPI.

References

Kelava, A., & Brandt, H. (2009). Estimation of nonlinear latent structural equation models using the extended unconstrained approach. *Review of Psychology, 16*(2), 123–132.

Lin, G. C., Wen, Z., Marsh, H. W., & Lin, H. S. (2010). Structural equation models of latent interactions: Clarification of orthogonalizing and double-mean-centering strategies. *Structural Equation Modeling, 17*(3), 374–391. doi:10.1080/10705511.2010.488999

Little, T. D., Bovaird, J. A., & Widaman, K. F. (2006). On the merits of orthogonalizing powered and product terms: Implications for modeling interactions among latent variables. *Structural Equation Modeling, 13*(4), 497–519. doi:10.1207/s15328007sem1304_1

Marsh, H. W., Wen, Z. & Hau, K. T. (2004). Structural equation models of latent interactions: Evaluation of alternative estimation strategies and indicator construction. *Psychological Methods, 9*(3), 275–300. doi:10.1037/1082989X.9.3.275

Marsh, H. W., Wen, Z., Hau, K. T., Little, T. D., Bovaird, J. A., & Widaman, K. F. (2007). Unconstrained Structural Equation Models of Latent Interactions: Contrasting Residual- and Mean-Centered Approaches. *Structural Equation Modeling: A Multidisciplinary Journal, 14*(4), 570-580. doi:10.1080/10705510701303921

Index