Package 'modsem'

April 17, 2024

Type Package

Title Latent Interaction (and Moderation) Analysis in Structural Equation Models (SEM)

Version 0.1.2

Maintainer Kjell Solem Slupphaug <slupphaugkjell@gmail.com>

Description Estimation of interaction (i.e., moderation) effects between latent variables in structural equation models (SEM).
The supported methods are:
The constrained approach (Algina & Moulder, 2001).
The unconstrained approach (Marsh et al., 2004).
The residual centering approach (Little et al., 2006).
The double centering approach (Lin et al., 2010).
The latent moderated structural equations (LMS) approach (Klein & Moosbrugger, 2000).
The quasi-maximum likelihood (QML) approach (Klein & Muthén, 2007) (temporarily unavailable)
The constrained- unconstrained, residual- and double centering- approaches are estimated via 'lavaan' (Rosseel, 2012), whilst the LMS- and QML- approaches are estimated via by ModSEM it self. Alternatively model can be estimated via 'Mplus' (Muthén & Muthén, 1998-2017).
References:
Algina, J., & Moulder, B. C. (2001).
<doi:10.1207/S15328007SEM0801_3>.
``A note on estimating the Jöreskog-Yang model for latent variable interaction using 'LISREL' 8.3."
Klein, A., & Moosbrugger, H. (2000).
<doi:10.1007/BF02296338>.
``Maximum likelihood estimation of latent interaction effects with the LMS method."
Klein, A. G., & Muthén, B. O. (2007).
<doi:10.1080/00273170701710205>.
``Quasi-maximum likelihood estimation of structural equation models with multiple interaction and quadratic effects."
Lin, G. C., Wen, Z., Marsh, H. W., & Lin, H. S. (2010).
<doi:10.1080/10705511.2010.488999>.
``Structural equation models of latent interactions: Clarification of orthogonalizing and double-mean-centering strategies."

Little, T. D., Bovaird, J. A., & Widaman, K. F. (2006).
<doi:10.1207/s15328007sem1304_1>.
``On the merits of orthogonalizing powered and product terms: Implications for modeling interactions among latent variables."
Marsh, H. W., Wen, Z., & Hau, K. T. (2004).
<doi:10.1037/1082-989X.9.3.275>.
``Structural equation models of latent interactions: evaluation of alternative estimation strategies and indicator construction."
Muthén, L.K. and Muthén, B.O. (1998-2017).
``'Mplus' User's Guide. Eighth Edition."
<https://www.statmodel.com/>.
Rosseel Y (2012).
<doi:10.18637/jss.v048.i02>.
``'lavaan': An R Package for Structural Equation Modeling."

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp, purrr, stringr, lavaan, rlang, MplusAutomation, nlme, R6, dplyr, matlib, mvnfast, stats, gaussquad, mvtnorm

Depends R (>= 3.50)

URL https://github.com/Kss2k/modsem

NeedsCompilation yes

Author Kjell Solem Slupphaug [aut, cre] (<https://orcid.org/0009-0005-8324-2834>)

Repository CRAN

Date/Publication 2024-04-17 19:40:02 UTC

R topics documented:

`modsem` *Interaction between latent variables*

Description

modsem is a function for estimating interaction effects between latent variables, in structural equation models (SEM's). Methods for estimating interaction effects in SEM's can basically be split into two frameworks: 1. Product Indicator based approaches ("dblcent", "rca", "uca", "ca", "pind"), and 2. Distributionally based approaches ("lms", "qml"). For the product indicator based approaces, modsem() is essentially a just a fancy wrapper for lavaan::sem() which generates the necessary syntax, and variables for the estimation of models with latent product indicators. The distributionally based approaches are implemented in seperately, and are are not estimated using lavaan::sem(), but rather using custom functions (largely) written in C++ for performance reasons.

Usage

```
modsem(
  modelSyntax = NULL,
  data = NULL,
  method = "dblcent",
  match = FALSE,
  standardizeData = FALSE,
  centerData = FALSE,
  firstLoadingFixed = TRUE,
  centerBefore = NULL,
  centerAfter = NULL,
  residualsProds = NULL,
  residualCovSyntax = NULL,
  constrainedProdMean = NULL,
  constrainedLoadings = NULL,
  constrainedVar = NULL,
  constrainedResCovMethod = NULL,
  auto.scale = "none",
  auto.center = "none",
  estimator = "ML",
  removeFromParTable = NULL,
  addToParTable = NULL,
  macros = NULL,
  run = TRUE,
  ...
)
```

Arguments

`modelSyntax`	lavaan syntax
`data`	dataframe

`method`	method to use: "rca" = residual centering approach (passed to lavaan), "uca" = unconstrained approach (passed to lavaan), "dblcent" = double centering approach (passed to lavaan), "pind" = prod ind approach, with no constraints or centering (passed to lavaan), "lms" = laten model structural equations (not passed to lavaan). "qml" = quasi maximum likelihood estimation of laten model structural equations (not passed to lavaan). "custom" = use parameters specified in the function call (passed to lavaan)
`match`	should the product indicators be created by using the match-strategy
`standardizeData`	should data be scaled before fitting model
`centerData`	should data be centered before fitting model
`firstLoadingFixed`	Sould the first factorloading in the latent prod be fixed to one?
`centerBefore`	should inds in prods be centered before computing prods (overwritten by method, if method != NULL)
`centerAfter`	should ind prods be centered after they have been computed?
`residualsProds`	should ind prods be centered using residuals (overwritten by method, if method != NULL)
`residualCovSyntax`	should syntax for residual covariances be produced (overwritten by method, if method != NULL)
`constrainedProdMean`	should syntax prod mean be produced (overwritten by method, if method != NULL)
`constrainedLoadings`	should syntax for constrained loadings be produced (overwritten by method, if method != NULL)
`constrainedVar`	should syntax for constrained variances be produced (overwritten by method, if method != NULL)
`constrainedResCovMethod`	method for constraining residual covariances
`auto.scale`	methods which should be scaled automatically (usually not useful)
`auto.center`	methods which should be centered automatically (usually not useful)
`estimator`	estimator to use in lavaan
`removeFromParTable`	rows to remove from partable before sending to lavaan (for advanced users)
`addToParTable`	rows to add to partable before sending to lavaan (for advanced users)
`macros`	macros to replace in the syntax before sending to lavaan
`run`	should the model be estimated
`...`	arguments passed to other functions, e.g,. lavaan

Value

ModSEM object

Examples

```
library(modsem)
# For more examples check README and/or GitHub.
# One interaction
m1 <- '
  # Outer Model
  X =~ x1 + x2 +x3
  Y =~ y1 + y2 + y3
  Z =~ z1 + z2 + z3

  # Inner model
  Y ~ X + Z + X:Z
'

# Double centering approach
est1 <- modsem(m1, oneInt)
summary(est1)

## Not run:
# The Constrained Approach
est1Constrained <- modsem(m1, oneInt, method = "ca")
summary(est1Constrained)

# LMS approach
est1LMS <- modsem(m1, oneInt, method = "lms")
summary(est1LMS)

# QML approach
est1QML <- modsem(m1, oneInt, method = "qml")
summary(est1QML)


## End(Not run)

# Theory Of Planned Behavior
tpb <- '
# Outer Model (Based on Hagger et al., 2007)
  LATT =~ att1 + att2 + att3 + att4 + att5
  LSN =~ sn1 + sn2
  LPBC =~ pbc1 + pbc2 + pbc3
  LINT =~ int1 + int2 + int3
  LBEH =~ b1 + b2

# Inner Model (Based on Steinmetz et al., 2011)
  # Covariances
  LATT ~~ LSN + LPBC
  LPBC ~~ LSN
  # Causal Relationsships
  LINT ~ LATT + LSN + LPBC
  LBEH ~ LINT + LPBC
  LBEH ~ LINT:LPBC
'
```

```
# double centering approach
estTpb <- modsem(tpb, data = TPB)
summary(estTpb)

## Not run:
# The Constrained Approach
estTpbConstrained <- modsem(tpb, data = TPB, method = "ca")
summary(estTpbConstrained)

# LMS approach
estTpbLMS <- modsem(tpb, data = TPB, method = "lms")
summary(estTpbLMS)

## End(Not run)
```

`modsemify` *Generate parameter table for lavaan syntax*

Description

Generate parameter table for lavaan syntax

Usage

```
modsemify(syntax)
```

Arguments

`syntax`	model syntax

Value

data.frame with columns lhs, op, rhs, mod

Examples

```
library(modsem)
m1 <- '
  # Outer Model
  X =~ x1 + x2 +x3
  Y =~ y1 + y2 + y3
  Z =~ z1 + z2 + z3

  # Inner model
  Y ~ X + Z + X:Z
'
modsemify(m1)
```

`multiplyIndicatorsCpp` *Multiply indicators*

Description

Multiply indicators

Usage

```
multiplyIndicatorsCpp(df)
```

Arguments

`df`	A data DataFrame

Value

A NumericVector

`oneInt` *oneInt*

Description

A simulated dataset with one interaction effect

`summary.ModSEM` *summary.ModSEM*

Description

summary.ModSEM

Usage

```
## S3 method for class 'ModSEM'
summary(object, ...)
```

Arguments

`object`	modsem object to summarized
`...`	arguments passed to lavaan::summary(), and nlsem::summary()

TPB *TPB*

Description

A simulated dataset based on the Theory of Planned Behaviora

tracePath *Estimate formulas for (co-)variance paths using Wright's path tracing rules*

Description

This function estimates the path from x to y using the path tracing rules, note that it only works with structural parameters, so "=~" are ignored. If you you want to use the measurement model, it should work if you replace it "=~" with "~" in the mod column of pt.

Usage

```
tracePath(pt, x, y, parenthesis = TRUE, ...)
```

Arguments

pt	A data frame with columns lhs, op, rhs, and mod, from modsemify(syntax)
x	source variable
y	destination variable
parenthesis	if TRUE, the output will be enclosed in parenthesis
...	additional arguments passed to tracePath

Value

A string with the estimated path (simplified if possible)

Examples

```
library(modsem)
m1 <- '
  # Outer Model
  X =~ x1 + x2 +x3
  Y =~ y1 + y2 + y3
  Z =~ z1 + z2 + z3

  # Inner model
  Y ~ X + Z + X:Z
'
pt <- modsemify(m1)
tracePath(pt, "Y", "Y") # variance of Y
```

tripleInt *tripleInt*

Description

A simulated dataset with three interaction effects

twoInt *twoInt*

Description

A simulated dataset with two interaction effects

Index