Package 'semmcci'

September 19, 2022

Title Monte Carlo Confidence Intervals in Structural Equation Modeling

Version 1.0.2

Description Monte Carlo confidence intervals for free and defined parameters in models fitted in the structural equation modeling package 'lavaan' can be generated using the 'semmcci' package. 'semmcci' has two main functions, namely, MC() and MCStd(). The output of 'lavaan' is passed as the first argument to the MC() function to generate Monte Carlo confidence intervals. Monte Carlo confidence intervals for the standardized estimates can also be generated by passing the output of the MC() function to the MCStd() function. Preacher and Selig (2012) <doi:10.1080/19312458.2012.679848>.

URL https://github.com/jeksterslab/semmcci, https://jeksterslab.github.io/semmcci/

BugReports https://github.com/jeksterslab/semmcci/issues

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 3.0.0), stats, lavaan, methods

Suggests knitr, rmarkdown, testthat, MASS

RoxygenNote 7.2.1

NeedsCompilation no

Author Ivan Jacob Agaloos Pesigan [aut, cre, cph] (<https://orcid.org/0000-0003-4818-8420>), Shu Fai Cheung [ctb] (<https://orcid.org/0000-0002-9871-9448>)

Maintainer Ivan Jacob Agaloos Pesigan <r.jeksterslab@gmail.com>

R topics documented:

MC *Monte Carlo Confidence Intervals*

Description

Calculates Monte Carlo confidence intervals for free and defined parameters

Usage

```
MC(object, R = 20000L, alpha = c(0.001, 0.01, 0.05))
```

Arguments

`object`	object of class `lavaan`.
`R`	Positive integer. Number of Monte Carlo replications.
`alpha`	Numeric vector. Significance level. Default value is `alpha = c(0.001, 0.01, 0.05)`.

Details

A sampling distribution of parameter estimates is generated from the multivariate normal distribution using the parameter estimates and the sampling variance-covariance matrix. Confidence intervals for free and defined parameters are generated using the simulated sampling distribution. Parameters can be defined using the `:=` operator in the `lavaan` model syntax.

Value

Returns an object of class `semmcci` which is a list with the following elements:

`R` Number of Monte Carlo replications.

`alpha` Significance level specified.

`lavaan` `lavaan` object.

`mvn` Method used to generate multivariate normal random variates.

`thetahat` Parameter estimates.

`thetahatstar` Sampling distribution of parameter estimates.

`ci` Confidence intervals.

The list element `ci` is a matrix with the following columns:

`est` Parameter estimates.

se Standard errors or the square root of the diagonals of the Monte Carlo sampling distribution of parameter estimates.

R Number of valid Monte Carlo replications.

... Percentiles that correspond to the confidence intervals defined by alpha.

Note that the rows in ci correspond to the model parameters.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
library(semmcci)
library(lavaan)

# Generate Data ------------------------------------------------------------
n <- 1000
x <- rnorm(n = n)
m <- 0.50 * x + rnorm(n = n)
y <- 0.25 * x + 0.50 * m + rnorm(n = n)
data <- data.frame(x, m, y)

# Fit Model in lavaan ------------------------------------------------------
model <- "
  y ~ cp * x + b * m
  m ~ a * x
  ab := a * b
"
fit <- sem(data = data, model = model)

# Monte Carlo --------------------------------------------------------------
MC(
  fit,
  R = 100L, # use a large value e.g., 20000L for actual research
  alpha = c(0.001, 0.01, 0.05)
)
```

MCStd *Standardized Monte Carlo Confidence Intervals*

Description

Calculates standardized Monte Carlo confidence intervals for free and defined parameters.

Usage

```
MCStd(object, alpha = c(0.001, 0.01, 0.05))
```

Arguments

object	object of class semmcci. Output of the MC() function.
alpha	Numeric vector. Significance level. Default value is alpha = c(0.001, 0.01, 0.05).

Details

The empirical sampling distribution of parameter estimates from the argument object is standardized, that is, each randomly generated vector of parameters is standardized. Defined parameters are computed from the standardized component parameters. Confidence intervals are generated using the standardized empirical sampling distribution.

Value

Returns an object of class semmcci_std which is a list with the following elements:

R Number of Monte Carlo replications.

alpha Significance level specified.

lavaan lavaan object.

mvn Method used to generate multivariate normal random variates.

thetahat Parameter estimates.

thetahatstar Sampling distribution of parameter estimates.

ci Confidence intervals.

thetahat_std Standardized parameter estimates.

thetahatstar_std Standardized sampling distribution of parameter estimates.

ci_std Standardized confidence intervals.

The list element ci_std is a matrix with the following columns:

est Standardized parameter estimates.

se Standard errors or the square root of the diagonals of the standardized Monte Carlo sampling distribution of parameter estimates.

R Number of valid Monte Carlo replications.

... Percentiles that correspond to the confidence intervals defined by alpha.

Note that the rows in ci_std correspond to the standardized model parameters.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
library(semmcci)
library(lavaan)

# Generate Data ------------------------------------------------------------
n <- 1000
x <- rnorm(n = n)
m <- 0.50 * x + rnorm(n = n)
y <- 0.25 * x + 0.50 * m + rnorm(n = n)
data <- data.frame(x, m, y)

# Fit Model in lavaan ------------------------------------------------------
model <- "
  y ~ cp * x + b * m
  m ~ a * x
  ab := a * b
"
fit <- sem(data = data, model = model, fixed.x = FALSE)

# Monte Carlo --------------------------------------------------------------
output <- MC(
  fit,
  R = 100L, # use a large value e.g., 20000L for actual research
  alpha = c(0.001, 0.01, 0.05)
)

# Standardized Monte Carlo -------------------------------------------------
MCStd(output)
```

---

print.semmcci *Print Method for Object of Class* semmcci

Description

Print Method for Object of Class semmcci

Usage

```
## S3 method for class 'semmcci'
print(x, digits = 4, ...)
```

Arguments

x	an object of class semmcci.
digits	Integer indicating the number of decimal places to display.
...	further arguments.

Value

Returns a matrix of estimates, standard errors, number of Monte Carlo replications, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
library(semmcci)
library(lavaan)

# Generate Data ------------------------------------------------------------
n <- 1000
x <- rnorm(n = n)
m <- 0.50 * x + rnorm(n = n)
y <- 0.25 * x + 0.50 * m + rnorm(n = n)
data <- data.frame(x, m, y)

# Fit Model in lavaan ------------------------------------------------------
model <- "
  y ~ cp * x + b * m
  m ~ a * x
  ab := a * b
"
fit <- sem(data = data, model = model)

# Monte Carlo --------------------------------------------------------------
print(
  MC(
    fit,
    R = 100L, # use a large value e.g., 20000L for actual research
    alpha = c(0.001, 0.01, 0.05)
  )
)
```

`print.semmcci_std` *Print Method for Object of Class* `semmcci_std`

Description

Print Method for Object of Class `semmcci_std`

Usage

```
## S3 method for class 'semmcci_std'
print(x, digits = 4, ...)
```

Arguments

x	an object of class semmcci_std.
digits	Integer indicating the number of decimal places to display.
...	further arguments.

Value

Returns a matrix of estimates, standard errors, number of Monte Carlo replications, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
library(semmcci)
library(lavaan)

# Generate Data ------------------------------------------------------------
n <- 1000
x <- rnorm(n = n)
m <- 0.50 * x + rnorm(n = n)
y <- 0.25 * x + 0.50 * m + rnorm(n = n)
data <- data.frame(x, m, y)

# Fit Model in lavaan ------------------------------------------------------
model <- "
  y ~ cp * x + b * m
  m ~ a * x
  ab := a * b
"
fit <- sem(data = data, model = model, fixed.x = FALSE)

# Monte Carlo --------------------------------------------------------------
output <- MC(
  fit,
  R = 100L, # use a large value e.g., 20000L for actual research
  alpha = c(0.001, 0.01, 0.05)
)

# Standardized Monte Carlo -------------------------------------------------
print(MCStd(output))
```

Index