

Package ‘semmcci’

August 10, 2023

Title Monte Carlo Confidence Intervals in Structural Equation Modeling

Version 1.1.2

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 three main functions, namely, MC(), MCMI(), and MCStd(). The output of 'lavaan' is passed as the first argument to the MC() function or the MCMI() 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 or the MCMI() function to the MCStd() function. A description of the package and code examples are presented in Pesigan and Cheung (2023) <doi:10.3758/s13428-023-02114-4>.

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)

Imports stats, lavaan, mice

Suggests knitr, rmarkdown, testthat, MASS, psych, Amelia, bmemLavaan

RoxygenNote 7.2.3

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:

coef.semmcci	2
confint.semmcci	3
MC	5
MCMI	7
MCStd	9
print.semmcci	11
summary.semmcci	13
vcov.semmcci	15
Index	17

coef.semmcci	<i>Parameter Estimates</i>
--------------	----------------------------

Description

Parameter Estimates

Usage

```
## S3 method for class 'semmcci'
coef(object, ...)
```

Arguments

object	Object of class semmcci.
...	additional arguments.

Value

Returns a vector of parameter estimates.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
library(semmcci)
library(lavaan)

# Data -----
data("Tal.Or", package = "psych")
df <- mice::ampute(Tal.Or)$amp

# Monte Carlo -----
## Fit Model in lavaan -----
model <- "
```

```

    reaction ~ cp * cond + b * pmi
    pmi ~ a * cond
    cond ~~ cond
    indirect := a * b
    direct := cp
    total := cp + (a * b)
"
fit <- sem(data = df, model = model, missing = "fiml")

## MC() -----
unstd <- MC(
  fit,
  R = 20L # use a large value e.g., 20000L for actual research
)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
coef(unstd)
coef(std)

# Monte Carlo (Multiple Imputation) -----
## Multiple Imputation -----
mi <- mice::mice(
  data = df,
  print = FALSE,
  m = 5L, # use a large value e.g., 100L for actual research,
  seed = 42
)

## Fit Model in lavaan -----
fit <- sem(data = df, model = model) # use default listwise deletion

## MCMI() -----
unstd <- MCMI(
  fit,
  mi = mi,
  R = 20L # use a large value e.g., 20000L for actual research
)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
coef(unstd)
coef(std)

```

Description

Monte Carlo Confidence Intervals for the Parameter Estimates

Usage

```
## S3 method for class 'semmcci'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

object	Object of class semmcci.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required.
...	additional arguments.

Value

Returns a matrix of confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
library(semmcci)
library(lavaan)

# Data -----
data("Tal.Or", package = "psych")
df <- mice::ampute(Tal.Or)$amp

# Monte Carlo -----
## Fit Model in lavaan -----
model <- "
  reaction ~ cp * cond + b * pmi
  pmi ~ a * cond
  cond ~~ cond
  indirect := a * b
  direct := cp
  total := cp + (a * b)
"
fit <- sem(data = df, model = model, missing = "fiml")

## MC() -----
unstd <- MC(
  fit,
  R = 20L # use a large value e.g., 20000L for actual research
)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
```

```

confint(unstd)
confint(std)

# Monte Carlo (Multiple Imputation) -----
## Multiple Imputation -----
mi <- mice::mice(
  data = df,
  print = FALSE,
  m = 5L, # use a large value e.g., 100L for actual research,
  seed = 42
)

## Fit Model in lavaan -----
fit <- sem(data = df, model = model) # use default listwise deletion

## MCMC() -----
unstd <- MCMC(
  fit,
  mi = mi,
  R = 20L # use a large value e.g., 20000L for actual research
)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
confint(unstd)
confint(std)

```

MC

Monte Carlo Confidence Intervals

Description

Calculates Monte Carlo confidence intervals for free and defined parameters.

Usage

```

MC(
  lav,
  R = 20000L,
  alpha = c(0.001, 0.01, 0.05),
  decomposition = "eigen",
  pd = TRUE,
  tol = 1e-06,
  seed = NULL
)

```

Arguments

lav	Object of class lavaan.
R	Positive integer. Number of Monte Carlo replications.
alpha	Numeric vector. Significance level α .
decomposition	Character string. Matrix decomposition of the sampling variance-covariance matrix for the data generation. If decomposition = "chol", use Cholesky decomposition. If decomposition = "eigen", use eigenvalue decomposition. If decomposition = "svd", use singular value decomposition.
pd	Logical. If pd = TRUE, check if the sampling variance-covariance matrix is positive definite using tol.
tol	Numeric. Tolerance used for pd.
seed	Integer. Random seed for reproducibility.

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:

call Function call.

args List of function arguments.

thetahat Parameter estimates $\hat{\theta}$.

thetahatstar Sampling distribution of parameter estimates $\hat{\theta}^*$.

fun Function used ("MC").

Author(s)

Ivan Jacob Agaloos Pesigan

References

- MacKinnon, D. P., Lockwood, C. M., & Williams, J. (2004). Confidence limits for the indirect effect: Distribution of the product and resampling methods. *Multivariate Behavioral Research*, 39(1), 99-128. doi:10.1207/s15327906mbr3901_4
- Pesigan, I. J. A., & Cheung, S. F. (2023). Monte Carlo confidence intervals for the indirect effect with missing data. *Behavior Research Methods*. doi:10.3758/s13428023021144
- Preacher, K. J., & Selig, J. P. (2012). Advantages of Monte Carlo confidence intervals for indirect effects. *Communication Methods and Measures*, 6(2), 77-98. doi:10.1080/19312458.2012.679848

See Also

Other Monte Carlo in Structural Equation Modeling Functions: [MCMC\(\)](#), [MCStd\(\)](#)

Examples

```
library(semmcci)
library(lavaan)

# Data -----
data("Tal.Or", package = "psych")
df <- mice::ampute(Tal.Or)$amp

# Monte Carlo -----
## Fit Model in lavaan -----
model <- "
  reaction ~ cp * cond + b * pmi
  pmi ~ a * cond
  cond ~~ cond
  indirect := a * b
  direct := cp
  total := cp + (a * b)
"
fit <- sem(data = df, model = model, missing = "fiml")

## MC() -----
MC(
  fit,
  R = 20L, # use a large value e.g., 20000L for actual research
  alpha = 0.05
)
```

MCMC

*Monte Carlo Confidence Intervals (Multiple Imputation)***Description**

Calculates Monte Carlo confidence intervals for free and defined parameters. Missing values are handled using multiple imputation.

Usage

```
MCMC(
  lav,
  mi,
  R = 20000L,
  alpha = c(0.001, 0.01, 0.05),
  decomposition = "eigen",
  pd = TRUE,
```

```

    tol = 1e-06,
    seed = NULL
  )

```

Arguments

<code>lav</code>	Object of class <code>lavaan</code> .
<code>mi</code>	Object of class <code>mids</code> (output of <code>mice::mice()</code>), object of class <code>amelia</code> (output of <code>Amelia::amelia()</code>), or a list of multiply imputed data sets.
<code>R</code>	Positive integer. Number of Monte Carlo replications.
<code>alpha</code>	Numeric vector. Significance level α .
<code>decomposition</code>	Character string. Matrix decomposition of the sampling variance-covariance matrix for the data generation. If <code>decomposition = "chol"</code> , use Cholesky decomposition. If <code>decomposition = "eigen"</code> , use eigenvalue decomposition. If <code>decomposition = "svd"</code> , use singular value decomposition.
<code>pd</code>	Logical. If <code>pd = TRUE</code> , check if the sampling variance-covariance matrix is positive definite using <code>tol</code> .
<code>tol</code>	Numeric. Tolerance used for <code>pd</code> .
<code>seed</code>	Integer. Random seed for reproducibility.

Details

A sampling distribution of parameter estimates is generated from the multivariate normal distribution using the parameter estimates and the sampling variance-covariance matrix obtained using multiple imputation. 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 `semccci` which is a list with the following elements:

call Function call.

args List of function arguments.

thetahat Parameter estimates $\hat{\theta}$.

thetahatstar Sampling distribution of parameter estimates $\hat{\theta}^*$.

fun Function used ("MCMI").

References

Pesigan, I. J. A., & Cheung, S. F. (2023). Monte Carlo confidence intervals for the indirect effect with missing data. *Behavior Research Methods*. doi:10.3758/s13428023021144

Rubin, D. B. (1987). *Multiple imputation for nonresponse in surveys*. John Wiley & Sons, Inc.

See Also

Other Monte Carlo in Structural Equation Modeling Functions: `MCStd()`, `MC()`

Examples

```

library(semmcci)
library(lavaan)

# Data -----
data("Tal.Or", package = "psych")
df <- mice::ampute(Tal.Or)$amp

# Monte Carlo (Multiple Imputation) -----
## Multiple Imputation -----
mi <- mice::mice(
  data = df,
  print = FALSE,
  m = 5L, # use a large value e.g., 100L for actual research,
  seed = 42
)

## Fit Model in lavaan -----
model <- "
  reaction ~ cp * cond + b * pmi
  pmi ~ a * cond
  cond ~~ cond
  indirect := a * b
  direct := cp
  total := cp + (a * b)
"

fit <- sem(data = df, model = model) # use default listwise deletion

## MCMI() -----
MCMI(
  fit,
  mi = mi,
  R = 20L, # use a large value e.g., 20000L for actual research
  alpha = 0.05
)

```

MCStd

Standardized Monte Carlo Confidence Intervals

Description

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

Usage

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

Arguments

<code>mc</code>	Output of the <code>MC()</code> or <code>MCMI()</code> function.
<code>alpha</code>	Numeric vector. Significance level α .

Details

The empirical sampling distribution of parameter estimates from the argument `mc` 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` which is a list with the following elements:

call Function call.

args List of function arguments.

thetahat Parameter estimates $\hat{\theta}$.

thetahatstar Sampling distribution of parameter estimates $\hat{\theta}^*$.

fun Function used ("MCStd").

Author(s)

Ivan Jacob Agaloos Pesigan

References

Pesigan, I. J. A., & Cheung, S. F. (2023). Monte Carlo confidence intervals for the indirect effect with missing data. *Behavior Research Methods*. doi:10.3758/s13428023021144

See Also

Other Monte Carlo in Structural Equation Modeling Functions: `MCMI()`, `MC()`

Examples

```
library(semmcci)
library(lavaan)

# Data -----
data("Tal.Or", package = "psych")
df <- mice::ampute(Tal.Or)$amp

# Monte Carlo -----
## Fit Model in lavaan -----
model <- "
  reaction ~ cp * cond + b * pmi
  pmi ~ a * cond
  cond ~~ cond
```

```

    indirect := a * b
    direct := cp
    total := cp + (a * b)
  "
fit <- sem(data = df, model = model, missing = "fiml")

## MC() -----
unstd <- MC(
  fit,
  R = 20L, # use a large value e.g., 20000L for actual research
  alpha = 0.05
)

## Standardized Monte Carlo -----
MCStd(unstd, alpha = 0.05)

# Monte Carlo (Multiple Imputation) -----
## Multiple Imputation -----
mi <- mice::mice(
  data = df,
  print = FALSE,
  m = 5L, # use a large value e.g., 100L for actual research,
  seed = 42
)

## Fit Model in lavaan -----
fit <- sem(data = df, model = model) # use default listwise deletion

## MCMI() -----
unstd <- MCMI(
  fit,
  mi = mi,
  R = 20L, # use a large value e.g., 20000L for actual research
  alpha = 0.05
)

## Standardized Monte Carlo -----
MCStd(unstd, alpha = 0.05)

```

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, alpha = NULL, digits = 4, ...)

```

Arguments

<code>x</code>	an object of class <code>semmcci</code> .
<code>alpha</code>	Numeric vector. Significance level α . If <code>alpha = NULL</code> , use the argument <code>alpha</code> used in <code>x</code> .
<code>digits</code>	Integer indicating the number of decimal places to display.
<code>...</code>	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)

# Data -----
data("Tal.Or", package = "psych")
df <- mice::ampute(Tal.Or)$amp

# Monte Carlo -----
## Fit Model in lavaan -----
model <- "
  reaction ~ cp * cond + b * pmi
  pmi ~ a * cond
  cond ~~ cond
  indirect := a * b
  direct := cp
  total := cp + (a * b)
"
fit <- sem(data = df, model = model, missing = "fiml")

## MC() -----
unstd <- MC(
  fit,
  R = 20L # use a large value e.g., 20000L for actual research
)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
print(unstd)
print(std)

# Monte Carlo (Multiple Imputation) -----
## Multiple Imputation -----
```

```

mi <- mice::mice(
  data = df,
  print = FALSE,
  m = 5L, # use a large value e.g., 100L for actual research,
  seed = 42
)

## Fit Model in lavaan -----
fit <- sem(data = df, model = model) # use default listwise deletion

## MCMC() -----
unstd <- MCMC(
  fit,
  mi = mi,
  R = 20L # use a large value e.g., 20000L for actual research
)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
print(unstd)
print(std)

```

summary.semmcci

*Summary Method for an Object of Class semmcci***Description**

Summary Method for an Object of Class semmcci

Usage

```

## S3 method for class 'semmcci'
summary(object, alpha = NULL, digits = 4, ...)

```

Arguments

object	Object of class semmcci.
alpha	Numeric vector. Significance level α . If alpha = NULL, use the argument alpha used in object.
digits	Digits to print.
...	additional 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)

# Data -----
data("Tal.Or", package = "psych")
df <- mice::ampute(Tal.Or)$amp

# Monte Carlo -----
## Fit Model in lavaan -----
model <- "
  reaction ~ cp * cond + b * pmi
  pmi ~ a * cond
  cond ~~ cond
  indirect := a * b
  direct := cp
  total := cp + (a * b)
"
fit <- sem(data = df, model = model, missing = "fiml")

## MC() -----
unstd <- MC(
  fit,
  R = 20L # use a large value e.g., 20000L for actual research
)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
summary(unstd)
summary(std)

# Monte Carlo (Multiple Imputation) -----
## Multiple Imputation -----
mi <- mice::mice(
  data = df,
  print = FALSE,
  m = 5L, # use a large value e.g., 100L for actual research,
  seed = 42
)

## Fit Model in lavaan -----
fit <- sem(data = df, model = model) # use default listwise deletion

## MCMI() -----
unstd <- MCMI(
  fit,
  mi = mi,
  R = 20L # use a large value e.g., 20000L for actual research
```

```

)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
summary(unstd)
summary(std)

```

vcov.semmcci

*Sampling Covariance Matrix of the Parameter Estimates***Description**

Sampling Covariance Matrix of the Parameter Estimates

Usage

```

## S3 method for class 'semmcci'
vcov(object, ...)

```

Arguments

```

object      Object of class semmcci.
...         additional arguments.

```

Value

Returns a matrix of the variance-covariance matrix of parameter estimates.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```

library(semmcci)
library(lavaan)

# Data -----
data("Tal.Or", package = "psych")
df <- mice::ampute(Tal.Or)$amp

# Monte Carlo -----
## Fit Model in lavaan -----
model <- "
  reaction ~ cp * cond + b * pmi
  pmi ~ a * cond
  cond ~~ cond
  indirect := a * b

```

```

    direct := cp
    total := cp + (a * b)
  "
fit <- sem(data = df, model = model, missing = "fiml")

## MC() -----
unstd <- MC(
  fit,
  R = 20L # use a large value e.g., 20000L for actual research
)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
vcov(unstd)
vcov(std)

# Monte Carlo (Multiple Imputation) -----
## Multiple Imputation -----
mi <- mice::mice(
  data = df,
  print = FALSE,
  m = 5L, # use a large value e.g., 100L for actual research,
  seed = 42
)

## Fit Model in lavaan -----
fit <- sem(data = df, model = model) # use default listwise deletion

## MCMI() -----
unstd <- MCMI(
  fit,
  mi = mi,
  R = 20L # use a large value e.g., 20000L for actual research
)

## Standardized Monte Carlo -----
std <- MCStd(unstd)
vcov(unstd)
vcov(std)

```


Index

* Monte Carlo in Structural Equation

Modeling Functions

MC, [5](#)

MCMI, [7](#)

MCStd, [9](#)

* mc

MC, [5](#)

MCMI, [7](#)

MCStd, [9](#)

* method

coef.semmcci, [2](#)

confint.semmcci, [3](#)

print.semmcci, [11](#)

summary.semmcci, [13](#)

vcov.semmcci, [15](#)

* semmcci

MC, [5](#)

MCMI, [7](#)

MCStd, [9](#)

Amelia::amelia(), [8](#)

coef.semmcci, [2](#)

confint.semmcci, [3](#)

MC, [5](#), [8](#), [10](#)

MC(), [10](#)

MCMI, [7](#), [7](#), [10](#)

MCMI(), [10](#)

MCStd, [7](#), [8](#), [9](#)

mice::mice(), [8](#)

print.semmcci, [11](#)

summary.semmcci, [13](#)

vcov.semmcci, [15](#)