Package: semmcci (via r-universe)

November 7, 2026

Title Monte Carlo Confidence Intervals in Structural Equation Modeling

Version 1.1.6.9000

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 (2024) <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

LazyData true

Roxygen list(markdown = TRUE)

Depends R (>= 3.5.0)

Imports stats, lavaan (>= 0.6), mice, parallel

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

Config/roxygen2/version 8.0.0.9000

Config/pak/sysreqs cmake make libicu-dev libx11-dev zlib1g-dev

Repository https://jeksterslab.r-universe.dev

Date/Publication 2026-06-14 00:15:53 UTC

RemoteUrl https://github.com/jeksterslab/semmcci

RemoteRef HEAD

RemoteSha 4cf752e82f4e6640073ef93d4804e80755d603d4

Contents

coef.semmcci . . . 2
confint.semmcci . . . 3
Func . . . 5
MC . . . 7
MCFunc . . . 9
MCGeneric . . . 11
MCMI . . . 14
MCStd . . . 16
print.semmcci . . . 18
summary.semmcci . . . 20
vcov.semmcci . . . 21

Index **24**

`coef.semmcci` *Parameter Estimates*

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 = 5L # 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 = 5L # use a large value e.g., 20000L for actual research
)

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

---

confint.semmcci *Monte Carlo Confidence Intervals for the Parameter Estimates*

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 = 5L # 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

## MCMI() -----------------------------------------------------------------
unstd <- MCMI(
  fit,
  mi = mi,
  R = 5L # use a large value e.g., 20000L for actual research
)

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

Func *Monte Carlo Confidence Intervals (List)*

Description

Calculates Monte Carlo confidence intervals for defined parameters.

Usage

```
Func(coef, func, ..., est, alpha = c(0.001, 0.01, 0.05), ncores = NULL)
```

Arguments

coef	List. A list of parameters.
func	R function.

1. The first argument x is the argument coef.

2. The function algebraically manipulates `coef` to return at a new numeric vector. It is best to have a named vector as an output.
3. The function can take additional named arguments passed using `...`.

`...`	Additional arguments to pass to `func`.
`est`	Numeric vector. Vector of original parameter estimates.
`alpha`	Numeric vector. Significance level α.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use single core.

Details

The distribution of parameters is provided as a list (`params`) and the definition of the function of paremeters is provided by a function (`func`). Confidence intervals for defined parameters are generated using the generated 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 ("Func").

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. (2024). 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: `MC()`, `MCFunc()`, `MCGeneric()`, `MCMI()`, `MCStd()`

Examples

```
library(semmcci)

## Generate Parameters -----------------------------------------------------
coef <- lapply(
  X = 1:5,
  FUN = function(i) {
    rnorm(n = 1)
  }
)

## Func() ------------------------------------------------------------------
### Define func -----------------------------------------------------------
func <- function(x) {
  out <- exp(x)
  names(out) <- "exp"
  out
}
### Generate Confidence Intervals -----------------------------------------
Func(
  coef,
  func = func,
  est = 1,
  alpha = 0.05
)
```

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. (2024). 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: Func(), MCFunc(), MCGeneric(), MCMI(), 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 = 5L, # use a large value e.g., 20000L for actual research
  alpha = 0.05
)
```

MCFunc *Monte Carlo Confidence Intervals (Function)*

Description

Calculates Monte Carlo confidence intervals for defined parameters.

Usage

```
MCFunc(
  coef,
  vcov,
  func,
  ...,
  R = 20000L,
  alpha = c(0.001, 0.01, 0.05),
```

```
  decomposition = "eigen",
  pd = TRUE,
  tol = 1e-06,
  seed = NULL,
  ncores = NULL
)
```

Arguments

coef	Numeric vector. Vector of estimated parameters.
vcov	Numeric matrix. Sampling variance-covariance matrix of estimated parameters.
func	R function. 1. The first argument x is the argument coef. 2. The function algebraically manipulates coef to return at a new numeric vector. It is best to have a named vector as an output. 3. The function can take additional named arguments passed using ....
...	Additional arguments to pass to func.
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.
ncores	Positive integer. Number of cores to use. If ncores = NULL, use single core.

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 defined parameters are generated using the simulated sampling distribution. Parameters are defined using the func argument.

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 ("MCFunc").

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. (2024). 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: `Func()`, `MC()`, `MCGeneric()`, `MCMI()`, `MCStd()`

Examples

```
library(semmcci)

## MCFunc() ------------------------------------------------------------
### Define func -------------------------------------------------------
func <- function(x) {
  out <- exp(x)
  names(out) <- "exp"
  out
}
### Generate Confidence Intervals -------------------------------------
MCFunc(
  coef = 0,
  vcov = matrix(1),
  func = func,
  R = 5L, # use a large value e.g., 20000L for actual research
  alpha = 0.05
)
```

`MCGeneric` *Monte Carlo Confidence Intervals (Generic)*

Description

Calculates Monte Carlo confidence intervals for defined parameters for any fitted model object with `coef` and `vcov` methods.

Usage

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

Arguments

`object`	R object. Fitted model object with `coef` and `vcov` methods that return a named vector of estimated parameters and sampling variance-covariance matrix, respectively.
`def`	List of character strings. A list of defined functions of parameters. The string should be a valid R expression when parsed and should result a single value when evaluated.
`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 defined parameters are generated using the simulated sampling distribution. Parameters are defined using the `def` argument.

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 ("MCGeneric").

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. (2024). 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: `Func()`, `MC()`, `MCFunc()`, `MCMI()`, `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
"
fit <- sem(data = df, model = model, missing = "fiml")

## MCGeneric() -----------------------------------------------------------------
MCGeneric(
  fit,
  R = 5L, # use a large value e.g., 20000L for actual research
  alpha = 0.05,
  def = list(
    "a * b",
    "cp + (a * b)"
  )
)
```

MCMI *Monte Carlo Confidence Intervals (Multiple Imputation)*

Description

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

Usage

```
MCMI(
  lav,
  mi,
  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.
mi	Object of class mids (output of mice::mice()), object of class amelia (output of Amelia::amelia()), or a list of multiply imputed data sets.
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 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 `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 ("MCMI").

References

Pesigan, I. J. A., & Cheung, S. F. (2024). 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: `Func()`, `MC()`, `MCFunc()`, `MCGeneric()`, `MCStd()`

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 = 5L, # 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

mc	Output of the MC() or MCMI() function.
alpha	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. (2024). 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: `Func()`, `MC()`, `MCFunc()`, `MCGeneric()`, `MCMI()`

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 = 5L, # 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 = 5L, # 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

`x`	an object of class `semmcci`.
`alpha`	Numeric vector. Significance level α. If `alpha = NULL`, use the argument `alpha` used in `x`.
`digits`	Integer indicating the number of decimal places to display.
`...`	further arguments.

Value

Prints 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 = 5L # 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

## MCMI() ---------------------------------------------------------------
unstd <- MCMI(
  fit,
  mi = mi,
  R = 5L # 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 = 5L # 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 = 5L # 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 = 5L # 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 = 5L # 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**
  Func, 5
  MC, 7
  MCFunc, 9
  MCGeneric, 11
  MCMI, 14
  MCStd, 16
∗ **mc**
  Func, 5
  MC, 7
  MCFunc, 9
  MCGeneric, 11
  MCMI, 14
  MCStd, 16
∗ **methods**
  coef.semmcci, 2
  confint.semmcci, 3
  print.semmcci, 18
  summary.semmcci, 20
  vcov.semmcci, 21
∗ **semmcci**
  Func, 5
  MC, 7
  MCFunc, 9
  MCGeneric, 11
  MCMI, 14
  MCStd, 16

Amelia::amelia(), *14*

coef.semmcci, 2
confint.semmcci, 3

Func, 5
Func(), *9*, *11*, *13*, *15*, *17*

MC, 7
MC(), *6*, *11*, *13*, *15–17*
MCFunc, 9
MCFunc(), *6*, *9*, *13*, *15*, *17*
MCGeneric, 11
MCGeneric(), *6*, *9*, *11*, *15*, *17*
MCMI, 14
MCMI(), *6*, *9*, *11*, *13*, *16, 17*
MCStd, 16
MCStd(), *6*, *9*, *11*, *13*, *15*
mice::mice(), *14*

print.semmcci, 18

summary.semmcci, 20

vcov.semmcci, 21