# semmcci: Monte Carlo Confidence Intervals

Ivan Jacob Agaloos Pesigan

## Installation

You can install the CRAN release of `semmcci` with:

```
install.packages("semmcci")
```

You can install the development version of `semmcci` from GitHub with:

```
if (!require("remotes")) install.packages("remotes")
remotes::install_github("jeksterslab/semmcci")
```

## Documentation

See GitHub Pages for package documentation.

## Description

In the Monte Carlo method, 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 by obtaining percentiles corresponding to $100(1 - \alpha)\%$ from the generated sampling distribution, where $\alpha$ is the significance level.

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. The package 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. A description of the package and code examples are presented in Pesigan and Cheung (2023).

# Example

A common application of the Monte Carlo method is to generate confidence intervals for the indirect effect. In the simple mediation model, variable `X` has an effect on variable `Y`, through a mediating variable `M`. This mediating or indirect effect is a product of path coefficients from the fitted model.

```
library(semmcci)
library(lavaan)
```

## 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)
```

## Model Specification

The indirect effect is defined by the product of the slopes of paths `X` to `M` labeled as `a` and `M` to `Y` labeled as `b`. In this example, we are interested in the confidence intervals of `indirect` defined as

the product of `a` and `b` using the `:=` operator in the `lavaan` model syntax.

```
model <- "
  Y ~ cp * X + b * M
  M ~ a * X
  indirect := a * b
  direct := cp
  total := cp + (a * b)
"
```

## Model Fitting

We can now fit the model using the `sem()` function from `lavaan`.

```
fit <- sem(data = data, model = model)
```

## Monte Carlo Confidence Intervals

The `fit` `lavaan` object can then be passed to the `MC()` function to generate Monte Carlo confidence intervals.

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

#> Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2182 0.0361 20000 0.1013 0.1261 0.1479 0.2887 0.3096 0.3442
#> b        0.5467 0.0331 20000 0.4379 0.4619 0.4817 0.6114 0.6320 0.6523
#> a        0.4849 0.0311 20000 0.3815 0.4047 0.4245 0.5456 0.5656 0.5832
#> Y~~Y     1.0179 0.0451 20000 0.8698 0.9006 0.9301 1.1070 1.1345 1.1630
#> M~~M     0.9531 0.0425 20000 0.8103 0.8449 0.8699 1.0353 1.0631 1.0918
```

```
#> X~~X     0.9965 0.0000 20000 0.9965 0.9965 0.9965 0.9965 0.9965 0.9965
#> indirect 0.2651 0.0234 20000 0.1922 0.2073 0.2209 0.3119 0.3287 0.3466
#> direct   0.2182 0.0361 20000 0.1013 0.1261 0.1479 0.2887 0.3096 0.3442
#> total    0.4832 0.0362 20000 0.3664 0.3903 0.4127 0.5547 0.5793 0.6030
```

## Standardized Monte Carlo Confidence Intervals

Standardized Monte Carlo Confidence intervals can be generated by passing the result of the `MC()` function to `MCStd()`.

**Note:** We recommend setting `fixed.x = FALSE` when generating standardized estimates and confidence intervals to model the variances and covariances of the predictors if they are assumed to be random.

```
fit <- sem(data = data, model = model, fixed.x = FALSE)
unstd <- MC(fit, R = 20000L, alpha = c(0.001, 0.01, 0.05))
vcov(unstd)

#>                      cp             b             a          Y~~Y          M~~M
#> cp        1.283507e-03 -5.186935e-04  6.545678e-06  1.205327e-05 -1.054391e-05
#> b        -5.186935e-04  1.066224e-03 -4.467962e-06 -8.704210e-06  2.218909e-06
#> a         6.545678e-06 -4.467962e-06  9.465010e-04  7.646218e-06 -1.791959e-05
#> Y~~Y      1.205327e-05 -8.704210e-06  7.646218e-06  2.121056e-03  2.273621e-05
#> M~~M     -1.054391e-05  2.218909e-06 -1.791959e-05  2.273621e-05  1.828964e-03
#> X~~X     -7.706578e-06 -2.158284e-07 -1.374491e-05 -5.845108e-06  1.464856e-06
#> indirect -2.476897e-04  5.143014e-04  5.159773e-04  1.174013e-07 -8.623563e-06
#> direct    1.283507e-03 -5.186935e-04  6.545678e-06  1.205327e-05 -1.054391e-05
#> total     1.035817e-03 -4.392160e-06  5.225230e-04  1.217067e-05 -1.916747e-05
#>                    X~~X      indirect        direct         total
#> cp       -7.706578e-06 -2.476897e-04  1.283507e-03  1.035817e-03
```

```
#> b         -2.158284e-07  5.143014e-04 -5.186935e-04 -4.392160e-06
#> a         -1.374491e-05  5.159773e-04  6.545678e-06  5.225230e-04
#> Y~~Y      -5.845108e-06  1.174013e-07  1.205327e-05  1.217067e-05
#> M~~M       1.464856e-06 -8.623563e-06 -1.054391e-05 -1.916747e-05
#> X~~X       1.964623e-03 -7.192095e-06 -7.706578e-06 -1.489867e-05
#> indirect -7.192095e-06  5.327170e-04 -2.476897e-04  2.850273e-04
#> direct   -7.706578e-06 -2.476897e-04  1.283507e-03  1.035817e-03
#> total    -1.489867e-05  2.850273e-04  1.035817e-03  1.320844e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1757 0.0285 20000 0.0780 0.1014 0.1196 0.2312 0.2483 0.2703
#> b        0.4807 0.0261 20000 0.3943 0.4119 0.4290 0.5309 0.5460 0.5619
#> a        0.4442 0.0253 20000 0.3609 0.3791 0.3941 0.4931 0.5084 0.5255
#> Y~~Y     0.6630 0.0243 20000 0.5843 0.5996 0.6143 0.7093 0.7231 0.7412
#> M~~M     0.8027 0.0225 20000 0.7239 0.7416 0.7568 0.8447 0.8563 0.8697
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2135 0.0171 20000 0.1613 0.1708 0.1806 0.2477 0.2592 0.2717
#> direct   0.1757 0.0285 20000 0.0780 0.1014 0.1196 0.2312 0.2483 0.2703
#> total    0.3893 0.0269 20000 0.2985 0.3187 0.3349 0.4404 0.4573 0.4721
```

## 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. https://doi.org/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*. https://doi.org/10.3758/s13428-023-02114-4

Preacher, K. J., & Selig, J. P. (2012). Advantages of Monte Carlo confidence intervals for indirect effects. *Communication Methods and Measures*, *6*(2), 77–98. https://doi.org/10.1080/19312458.2012.679848

R Core Team. (2023). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/

Tofighi, D., & Kelley, K. (2019). Indirect effects in sequential mediation models: Evaluating methods for hypothesis testing and confidence interval formation. *Multivariate Behavioral Research*, *55*(2), 188–210. https://doi.org/10.1080/00273171.2019.1618545

Tofighi, D., & MacKinnon, D. P. (2015). Monte Carlo confidence intervals for complex functions of indirect effects. *Structural Equation Modeling: A Multidisciplinary Journal*, *23*(2), 194–205. https://doi.org/10.1080/10705511.2015.1057284