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 α 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 (2024).

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 = 0.05)

#> Monte Carlo Confidence Intervals
#>             est     se     R   2.5%  97.5%
#> cp       0.2630 0.0361 20000 0.1927 0.3351
#> b        0.5352 0.0334 20000 0.4700 0.6011
#> a        0.5176 0.0300 20000 0.4587 0.5769
#> Y~~Y     1.0166 0.0452 20000 0.9273 1.1050
#> M~~M     0.8987 0.0401 20000 0.8196 0.9762
```

```
#> X~~X     1.0023 0.0000 20000 1.0023 1.0023
#> indirect 0.2770 0.0237 20000 0.2320 0.3251
#> direct   0.2630 0.0361 20000 0.1927 0.3351
#> total    0.5400 0.0353 20000 0.4714 0.6094
```

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 = 0.05)
vcov(unstd)

#>                      cp             b             a          Y~~Y          M~~M
#> cp        1.322445e-03 -5.750927e-04 -8.913794e-06 -7.643317e-07  9.742360e-06
#> b        -5.750927e-04  1.117835e-03 -5.319823e-06 -1.402039e-05  3.579394e-06
#> a        -8.913794e-06 -5.319823e-06  8.746306e-04 -7.536438e-06  3.176189e-06
#> Y~~Y     -7.643317e-07 -1.402039e-05 -7.536438e-06  2.041021e-03  5.970824e-06
#> M~~M      9.742360e-06  3.579394e-06  3.176189e-06  5.970824e-06  1.593781e-03
#> X~~X     -1.376833e-05  2.320713e-05 -3.135028e-06 -1.079666e-05 -1.099491e-05
#> indirect -3.028692e-04  5.760076e-04  4.652447e-04 -1.095921e-05  3.239886e-06
#> direct    1.322445e-03 -5.750927e-04 -8.913794e-06 -7.643317e-07  9.742360e-06
#> total     1.019576e-03  9.148498e-07  4.563310e-04 -1.172354e-05  1.298225e-05
#>                    X~~X      indirect        direct         total
#> cp       -1.376833e-05 -3.028692e-04  1.322445e-03  1.019576e-03
```

```
#> b          2.320713e-05  5.760076e-04 -5.750927e-04  9.148498e-07
#> a         -3.135028e-06  4.652447e-04 -8.913794e-06  4.563310e-04
#> Y~~Y      -1.079666e-05 -1.095921e-05 -7.643317e-07 -1.172354e-05
#> M~~M      -1.099491e-05  3.239886e-06  9.742360e-06  1.298225e-05
#> X~~X       2.009007e-03  1.024192e-05 -1.376833e-05 -3.526419e-06
#> indirect   1.024192e-05  5.481927e-04 -3.028692e-04  2.453235e-04
#> direct    -1.376833e-05 -3.028692e-04  1.322445e-03  1.019576e-03
#> total     -3.526419e-06  2.453235e-04  1.019576e-03  1.264899e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2103 0.0286 20000 0.1165 0.1371 0.1542 0.2658 0.2830 0.3007
#> b         0.4620 0.0266 20000 0.3757 0.3914 0.4089 0.5130 0.5279 0.5481
#> a         0.4796 0.0241 20000 0.3964 0.4158 0.4312 0.5258 0.5406 0.5582
#> Y~~Y      0.6490 0.0244 20000 0.5699 0.5855 0.6005 0.6958 0.7122 0.7297
#> M~~M      0.7700 0.0231 20000 0.6885 0.7077 0.7235 0.8141 0.8271 0.8429
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect  0.2216 0.0173 20000 0.1662 0.1779 0.1880 0.2557 0.2664 0.2809
#> direct    0.2103 0.0286 20000 0.1165 0.1371 0.1542 0.2658 0.2830 0.3007
#> total     0.4320 0.0256 20000 0.3418 0.3630 0.3807 0.4810 0.4956 0.5131
```

References

Pesigan, I. J. A., & Cheung, S. F. (2024). Monte Carlo confidence intervals for the indirect effect with missing data. *Behavior Research Methods*, *56*(3), 1678–1696. https://doi.org/10.3758/s13428-023-02114-4

R Core Team. (2025). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/