semmcci: Monte Carlo Confidence Intervals

Ivan Jacob Agaloos Pesigan

Installation

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

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

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

```r
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 (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 = 0.05)

#> Monte Carlo Confidence Intervals
#>             est     se     R   2.5%  97.5%
#> cp       0.2227 0.0347 20000 0.1546 0.2898
#> b        0.5359 0.0318 20000 0.4741 0.5984
#> a        0.5089 0.0301 20000 0.4498 0.5669
#> Y~~Y     1.0162 0.0456 20000 0.9283 1.1056
#> M~~M     0.9907 0.0441 20000 0.9042 1.0769
```

```
#> X~~X     1.0870 0.0000 20000 1.0870 1.0870
#> indirect 0.2727 0.0229 20000 0.2288 0.3182
#> direct   0.2227 0.0347 20000 0.1546 0.2898
#> total    0.4954 0.0345 20000 0.4271 0.5625
```

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        1.213258e-03 -5.246586e-04 -5.762148e-06  3.482434e-06 2.777293e-06
#> b        -5.246586e-04  1.020861e-03  3.463212e-06  3.046951e-06 3.675673e-06
#> a        -5.762148e-06  3.463212e-06  9.063006e-04 -3.886751e-06 3.209967e-06
#> Y~~Y      3.482434e-06  3.046951e-06 -3.886751e-06  2.034561e-03 1.478206e-05
#> M~~M      2.777293e-06  3.675673e-06  3.209967e-06  1.478206e-05 1.986442e-03
#> X~~X      6.318465e-06 -1.203962e-06  1.064207e-08 -3.127643e-05 1.083881e-05
#> indirect -2.704308e-04  5.219255e-04  4.876330e-04 -9.175646e-07 3.670238e-06
#> direct    1.213258e-03 -5.246586e-04 -5.762148e-06  3.482434e-06 2.777293e-06
#> total     9.428274e-04 -2.733068e-06  4.818709e-04  2.564869e-06 6.447531e-06
#>                   X~~X      indirect        direct         total
#> cp        6.318465e-06 -2.704308e-04  1.213258e-03  9.428274e-04
```

```
#> b        -1.203962e-06  5.219255e-04 -5.246586e-04 -2.733068e-06
#> a         1.064207e-08  4.876330e-04 -5.762148e-06  4.818709e-04
#> Y~~Y     -3.127643e-05 -9.175646e-07  3.482434e-06  2.564869e-06
#> M~~M      1.083881e-05  3.670238e-06  2.777293e-06  6.447531e-06
#> X~~X      2.354875e-03 -1.103950e-06  6.318465e-06  5.214515e-06
#> indirect -1.103950e-06  5.282104e-04 -2.704308e-04  2.577796e-04
#> direct    6.318465e-06 -2.704308e-04  1.213258e-03  9.428274e-04
#> total     5.214515e-06  2.577796e-04  9.428274e-04  1.200607e-03
```

```
MCStd(unstd)
```

```
#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1855 0.0287 20000 0.0952 0.1115 0.1286 0.2405 0.2583 0.2799
#> b        0.4828 0.0262 20000 0.3972 0.4160 0.4316 0.5336 0.5504 0.5648
#> a        0.4704 0.0245 20000 0.3856 0.4065 0.4217 0.5173 0.5308 0.5479
#> Y~~Y     0.6483 0.0243 20000 0.5666 0.5847 0.5992 0.6947 0.7075 0.7224
#> M~~M     0.7787 0.0231 20000 0.6998 0.7182 0.7324 0.8222 0.8347 0.8513
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2271 0.0175 20000 0.1709 0.1835 0.1939 0.2620 0.2729 0.2853
#> direct   0.1855 0.0287 20000 0.0952 0.1115 0.1286 0.2405 0.2583 0.2799
#> total    0.4125 0.0263 20000 0.3275 0.3430 0.3598 0.4628 0.4775 0.4975
```

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