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 (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       0.2486 0.0349 20000 0.1794 0.3167
#> b        0.5366 0.0318 20000 0.4746 0.5993
#> a        0.5238 0.0307 20000 0.4632 0.5837
#> Y~~Y     0.9530 0.0428 20000 0.8689 1.0369
#> M~~M     0.9509 0.0427 20000 0.8667 1.0345
```

```
#> X~~X     1.0232 0.0000 20000 1.0232 1.0232
#> indirect 0.2811 0.0234 20000 0.2363 0.3283
#> direct   0.2486 0.0349 20000 0.1794 0.3167
#> total    0.5296 0.0348 20000 0.4609 0.5972
```

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        1.206753e-03 -5.246122e-04 -9.777599e-06  3.977539e-06 -8.648381e-06
#> b        -5.246122e-04  9.990624e-04 -4.418747e-06 -6.819281e-06  6.979718e-06
#> a        -9.777599e-06 -4.418747e-06  9.392479e-04 -9.920553e-06 -5.149393e-06
#> Y~~Y      3.977539e-06 -6.819281e-06 -9.920553e-06  1.813424e-03 -5.324116e-06
#> M~~M     -8.648381e-06  6.979718e-06 -5.149393e-06 -5.324116e-06  1.811818e-03
#> X~~X      8.400330e-07  1.076077e-05  6.875755e-06 -8.802182e-07 -1.394091e-05
#> indirect -2.800202e-04  5.205892e-04  5.016319e-04 -8.974258e-06  7.608427e-07
#> direct    1.206753e-03 -5.246122e-04 -9.777599e-06  3.977539e-06 -8.648381e-06
#> total     9.267329e-04 -4.023020e-06  4.918543e-04 -4.996719e-06 -7.887538e-06
#>                    X~~X      indirect        direct         total
#> cp        8.400330e-07 -2.800202e-04  1.206753e-03  9.267329e-04
```

```
#> b         1.076077e-05  5.205892e-04 -5.246122e-04 -4.023020e-06
#> a         6.875755e-06  5.016319e-04 -9.777599e-06  4.918543e-04
#> Y~~Y     -8.802182e-07 -8.974258e-06  3.977539e-06 -4.996719e-06
#> M~~M     -1.394091e-05  7.608427e-07 -8.648381e-06 -7.887538e-06
#> X~~X      2.092627e-03  9.327583e-06  8.400330e-07  1.016762e-05
#> indirect  9.327583e-06  5.425692e-04 -2.800202e-04  2.625491e-04
#> direct    8.400330e-07 -2.800202e-04  1.206753e-03  9.267329e-04
#> total     1.016762e-05  2.625491e-04  9.267329e-04  1.189282e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2044 0.0282 20000 0.1114 0.1303 0.1483 0.2588 0.2764 0.2995
#> b        0.4840 0.0260 20000 0.3980 0.4155 0.4314 0.5345 0.5498 0.5678
#> a        0.4774 0.0246 20000 0.3932 0.4131 0.4281 0.5244 0.5380 0.5549
#> Y~~Y     0.6296 0.0242 20000 0.5518 0.5661 0.5812 0.6766 0.6925 0.7085
#> M~~M     0.7721 0.0234 20000 0.6921 0.7105 0.7250 0.8168 0.8294 0.8454
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2311 0.0176 20000 0.1774 0.1874 0.1970 0.2661 0.2766 0.2890
#> direct   0.2044 0.0282 20000 0.1114 0.1303 0.1483 0.2588 0.2764 0.2995
#> total    0.4354 0.0256 20000 0.3490 0.3662 0.3835 0.4843 0.5000 0.5156
```

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