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 = 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       0.2628 0.0360 20000 0.1449 0.1700 0.1915 0.3333 0.3555 0.3806
#> b        0.4809 0.0327 20000 0.3766 0.3940 0.4155 0.5443 0.5635 0.5845
#> a        0.5097 0.0308 20000 0.4094 0.4305 0.4491 0.5704 0.5899 0.6082
#> Y~~Y     1.0796 0.0486 20000 0.9197 0.9540 0.9848 1.1754 1.2038 1.2376
#> M~~M     1.0107 0.0454 20000 0.8630 0.8944 0.9222 1.0983 1.1277 1.1605
```

```
#> X~~X     1.0707 0.0000 20000 1.0707 1.0707 1.0707 1.0707 1.0707 1.0707
#> indirect 0.2451 0.0224 20000 0.1776 0.1904 0.2026 0.2903 0.3068 0.3241
#> direct   0.2628 0.0360 20000 0.1449 0.1700 0.1915 0.3333 0.3555 0.3806
#> total    0.5079 0.0352 20000 0.3953 0.4171 0.4386 0.5757 0.5984 0.6207
```

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        1.281722e-03 -5.427671e-04 -4.189556e-06  1.414101e-05  2.376254e-05
#> b        -5.427671e-04  1.062472e-03  2.466302e-06  8.394431e-07 -2.932467e-06
#> a        -4.189556e-06  2.466302e-06  9.366328e-04  2.882179e-07 -1.474542e-05
#> Y~~Y      1.414101e-05  8.394431e-07  2.882179e-07  2.374478e-03  5.267115e-06
#> M~~M      2.376254e-05 -2.932467e-06 -1.474542e-05  5.267115e-06  2.029594e-03
#> X~~X     -4.800553e-06  7.586405e-06 -1.756537e-06 -5.137239e-06  6.575730e-06
#> indirect -2.778824e-04  5.419231e-04  4.520773e-04  9.043220e-07 -8.418901e-06
#> direct    1.281722e-03 -5.427671e-04 -4.189556e-06  1.414101e-05  2.376254e-05
#> total     1.003840e-03 -8.440530e-07  4.478878e-04  1.504533e-05  1.534364e-05
#>                    X~~X      indirect        direct         total
#> cp       -4.800553e-06 -2.778824e-04  1.281722e-03  1.003840e-03
```

```
#> b         7.586405e-06  5.419231e-04 -5.427671e-04 -8.440530e-07
#> a        -1.756537e-06  4.520773e-04 -4.189556e-06  4.478878e-04
#> Y~~Y     -5.137239e-06  9.043220e-07  1.414101e-05  1.504533e-05
#> M~~M      6.575730e-06 -8.418901e-06  2.376254e-05  1.534364e-05
#> X~~X      2.271395e-03  2.860358e-06 -4.800553e-06 -1.940195e-06
#> indirect  2.860358e-06  4.944213e-04 -2.778824e-04  2.165389e-04
#> direct   -4.800553e-06 -2.778824e-04  1.281722e-03  1.003840e-03
#> total    -1.940195e-06  2.165389e-04  1.003840e-03  1.220379e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2157 0.0289 20000 0.1258 0.1418 0.1591 0.2728 0.2913 0.3110
#> b        0.4330 0.0272 20000 0.3452 0.3620 0.3783 0.4850 0.4996 0.5176
#> a        0.4646 0.0248 20000 0.3799 0.4002 0.4153 0.5119 0.5253 0.5414
#> Y~~Y     0.6792 0.0242 20000 0.5970 0.6154 0.6295 0.7250 0.7395 0.7549
#> M~~M     0.7842 0.0230 20000 0.7069 0.7240 0.7380 0.8275 0.8398 0.8557
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2012 0.0169 20000 0.1490 0.1587 0.1681 0.2346 0.2460 0.2587
#> direct   0.2157 0.0289 20000 0.1258 0.1418 0.1591 0.2728 0.2913 0.3110
#> total    0.4168 0.0260 20000 0.3318 0.3482 0.3651 0.4667 0.4829 0.4998
```

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