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       0.2243 0.0355 20000 0.1074 0.1332 0.1551 0.2937 0.3131 0.3415
#> b        0.5518 0.0319 20000 0.4524 0.4714 0.4902 0.6144 0.6335 0.6534
#> a        0.5014 0.0319 20000 0.4004 0.4190 0.4393 0.5639 0.5823 0.6107
#> Y~~Y     1.0270 0.0459 20000 0.8760 0.9106 0.9378 1.1183 1.1446 1.1770
#> M~~M     1.0019 0.0450 20000 0.8576 0.8860 0.9129 1.0909 1.1174 1.1440
```

```
#> X~~X     1.0080 0.0000 20000 1.0080 1.0080 1.0080 1.0080 1.0080 1.0080
#> indirect 0.2767 0.0239 20000 0.2055 0.2173 0.2316 0.3255 0.3416 0.3595
#> direct   0.2243 0.0355 20000 0.1074 0.1332 0.1551 0.2937 0.3131 0.3415
#> total    0.5010 0.0362 20000 0.3759 0.4087 0.4290 0.5714 0.5929 0.6185
```

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        1.272891e-03 -5.093678e-04  1.078949e-05  3.449957e-06  1.064877e-05
#> b        -5.093678e-04  1.013500e-03 -3.144595e-06 -1.170779e-05 -1.062281e-05
#> a         1.078949e-05 -3.144595e-06  1.003824e-03 -5.750704e-06 -2.201589e-06
#> Y~~Y      3.449957e-06 -1.170779e-05 -5.750704e-06  2.122383e-03 -6.461760e-06
#> M~~M      1.064877e-05 -1.062281e-05 -2.201589e-06 -6.461760e-06  2.017430e-03
#> X~~X     -2.323838e-05 -3.060102e-06  1.340446e-05 -5.271277e-06  1.154761e-05
#> indirect -2.498292e-04  5.068748e-04  5.522391e-04 -8.757883e-06 -6.591427e-06
#> direct    1.272891e-03 -5.093678e-04  1.078949e-05  3.449957e-06  1.064877e-05
#> total     1.023061e-03 -2.493000e-06  5.630286e-04 -5.307925e-06  4.057340e-06
#>                    X~~X      indirect        direct         total
#> cp       -2.323838e-05 -2.498292e-04  1.272891e-03  1.023061e-03
```

```
#> b         -3.060102e-06  5.068748e-04 -5.093678e-04 -2.493000e-06
#> a          1.340446e-05  5.522391e-04  1.078949e-05  5.630286e-04
#> Y~~Y      -5.271277e-06 -8.757883e-06  3.449957e-06 -5.307925e-06
#> M~~M       1.154761e-05 -6.591427e-06  1.064877e-05  4.057340e-06
#> X~~X       2.013415e-03  5.922915e-06 -2.323838e-05 -1.731547e-05
#> indirect   5.922915e-06  5.600562e-04 -2.498292e-04  3.102269e-04
#> direct    -2.323838e-05 -2.498292e-04  1.272891e-03  1.023061e-03
#> total     -1.731547e-05  3.102269e-04  1.023061e-03  1.333288e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.1789 0.0281 20000 0.0861 0.1063 0.1236 0.2333 0.2518 0.2705
#> b         0.4911 0.0256 20000 0.4062 0.4228 0.4397 0.5402 0.5558 0.5714
#> a         0.4493 0.0254 20000 0.3586 0.3833 0.3988 0.4987 0.5144 0.5342
#> Y~~Y      0.6479 0.0243 20000 0.5676 0.5835 0.5992 0.6944 0.7087 0.7232
#> M~~M      0.7981 0.0228 20000 0.7147 0.7354 0.7513 0.8410 0.8531 0.8714
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect  0.2206 0.0174 20000 0.1646 0.1769 0.1875 0.2554 0.2667 0.2794
#> direct    0.1789 0.0281 20000 0.0861 0.1063 0.1236 0.2333 0.2518 0.2705
#> total     0.3995 0.0266 20000 0.3101 0.3293 0.3466 0.4508 0.4655 0.4840
```

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