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       0.2333 0.0356 20000 0.1636 0.3033
#> b        0.5135 0.0318 20000 0.4511 0.5762
#> a        0.5013 0.0318 20000 0.4380 0.5623
#> Y~~Y     0.9814 0.0438 20000 0.8968 1.0677
#> M~~M     0.9723 0.0436 20000 0.8871 1.0571
```

```
#> X~~X     0.9685 0.0000 20000 0.9685 0.9685
#> indirect 0.2574 0.0227 20000 0.2142 0.3030
#> direct   0.2333 0.0356 20000 0.1636 0.3033
#> total    0.4907 0.0358 20000 0.4205 0.5604
```

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        1.275496e-03 -5.094345e-04 -8.011727e-06 -9.161171e-07 -1.419400e-06
#> b        -5.094345e-04  1.017331e-03  9.816864e-06  9.015130e-06 -5.726813e-06
#> a        -8.011727e-06  9.816864e-06  1.006460e-03  1.152598e-06  1.057794e-05
#> Y~~Y     -9.161171e-07  9.015130e-06  1.152598e-06  1.926949e-03 -1.636743e-06
#> M~~M     -1.419400e-06 -5.726813e-06  1.057794e-05 -1.636743e-06  1.904090e-03
#> X~~X     -6.225897e-06  4.955365e-06  1.236015e-07 -6.609187e-06 -5.341083e-05
#> indirect -2.590208e-04  5.144343e-04  5.216260e-04  5.238145e-06  2.791805e-06
#> direct    1.275496e-03 -5.094345e-04 -8.011727e-06 -9.161171e-07 -1.419400e-06
#> total     1.016476e-03  4.999722e-06  5.136143e-04  4.322028e-06  1.372405e-06
#>                    X~~X      indirect        direct         total
#> cp       -6.225897e-06 -2.590208e-04  1.275496e-03  1.016476e-03
```

```
#> b         4.955365e-06  5.144343e-04 -5.094345e-04  4.999722e-06
#> a         1.236015e-07  5.216260e-04 -8.011727e-06  5.136143e-04
#> Y~~Y     -6.609187e-06  5.238145e-06 -9.161171e-07  4.322028e-06
#> M~~M     -5.341083e-05  2.791805e-06 -1.419400e-06  1.372405e-06
#> X~~X      1.865274e-03  2.725185e-06 -6.225897e-06 -3.500712e-06
#> indirect  2.725185e-06  5.264130e-04 -2.590208e-04  2.673921e-04
#> direct   -6.225897e-06 -2.590208e-04  1.275496e-03  1.016476e-03
#> total    -3.500712e-06  2.673921e-04  1.016476e-03  1.283868e-03
```

```
MCStd(unstd)
```

```
#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1893 0.0286 20000 0.0931 0.1146 0.1335 0.2451 0.2629 0.2807
#> b        0.4668 0.0264 20000 0.3791 0.3982 0.4145 0.5180 0.5341 0.5497
#> a        0.4474 0.0253 20000 0.3626 0.3801 0.3964 0.4961 0.5111 0.5259
#> Y~~Y     0.6672 0.0243 20000 0.5851 0.6028 0.6179 0.7140 0.7273 0.7427
#> M~~M     0.7998 0.0226 20000 0.7234 0.7387 0.7538 0.8429 0.8555 0.8685
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2089 0.0172 20000 0.1571 0.1660 0.1755 0.2429 0.2544 0.2659
#> direct   0.1893 0.0286 20000 0.0931 0.1146 0.1335 0.2451 0.2629 0.2807
#> total    0.3982 0.0266 20000 0.3078 0.3258 0.3455 0.4486 0.4655 0.4821
```

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