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       0.2440 0.0343 20000 0.1756 0.3115
#> b        0.5197 0.0297 20000 0.4613 0.5782
#> a        0.4766 0.0332 20000 0.4105 0.5415
#> Y~~Y     0.9202 0.0410 20000 0.8398 1.0001
#> M~~M     1.0252 0.0459 20000 0.9355 1.1153
```

```
#> X~~X     0.9397 0.0000 20000 0.9397 0.9397
#> indirect 0.2477 0.0222 20000 0.2053 0.2923
#> direct   0.2440 0.0343 20000 0.1756 0.3115
#> total    0.4917 0.0358 20000 0.4217 0.5626
```

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        1.167307e-03 -4.262839e-04 -9.643228e-06 -6.108110e-06 -1.187058e-06
#> b        -4.262839e-04  9.019869e-04 -3.095467e-06  1.228048e-06  1.291867e-05
#> a        -9.643228e-06 -3.095467e-06  1.099163e-03 -7.339422e-06  1.361389e-05
#> Y~~Y     -6.108110e-06  1.228048e-06 -7.339422e-06  1.674265e-03 -7.727285e-06
#> M~~M     -1.187058e-06  1.291867e-05  1.361389e-05 -7.727285e-06  2.140556e-03
#> X~~X     -2.871632e-06  1.260015e-05  1.941325e-06  2.014967e-05  5.972703e-06
#> indirect -2.078629e-04  4.278436e-04  5.694891e-04 -3.249375e-06  1.300235e-05
#> direct    1.167307e-03 -4.262839e-04 -9.643228e-06 -6.108110e-06 -1.187058e-06
#> total     9.594440e-04  1.559711e-06  5.598459e-04 -9.357485e-06  1.181530e-05
#>                    X~~X      indirect        direct         total
#> cp       -2.871632e-06 -2.078629e-04  1.167307e-03  9.594440e-04
```

```
#> b         1.260015e-05  4.278436e-04 -4.262839e-04  1.559711e-06
#> a         1.941325e-06  5.694891e-04 -9.643228e-06  5.598459e-04
#> Y~~Y      2.014967e-05 -3.249375e-06 -6.108110e-06 -9.357485e-06
#> M~~M      5.972703e-06  1.300235e-05 -1.187058e-06  1.181530e-05
#> X~~X      1.742832e-03  6.810391e-06 -2.871632e-06  3.938759e-06
#> indirect  6.810391e-06  5.005000e-04 -2.078629e-04  2.926371e-04
#> direct   -2.871632e-06 -2.078629e-04  1.167307e-03  9.594440e-04
#> total     3.938759e-06  2.926371e-04  9.594440e-04  1.252081e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1982 0.0274 20000 0.1072 0.1269 0.1438 0.2514 0.2686 0.2877
#> b        0.4846 0.0253 20000 0.4017 0.4181 0.4338 0.5333 0.5479 0.5699
#> a        0.4152 0.0262 20000 0.3282 0.3460 0.3625 0.4650 0.4812 0.4972
#> Y~~Y     0.6461 0.0243 20000 0.5607 0.5810 0.5971 0.6926 0.7080 0.7290
#> M~~M     0.8276 0.0217 20000 0.7528 0.7685 0.7837 0.8686 0.8803 0.8923
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2012 0.0168 20000 0.1475 0.1577 0.1687 0.2342 0.2453 0.2558
#> direct   0.1982 0.0274 20000 0.1072 0.1269 0.1438 0.2514 0.2686 0.2877
#> total    0.3994 0.0264 20000 0.3075 0.3288 0.3461 0.4503 0.4655 0.4832
```

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*, *56*(3), 1678–1696. 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. (2024). *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