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       0.2791 0.0346 20000 0.1686 0.1906 0.2112 0.3465 0.3693 0.3911
#> b        0.4699 0.0324 20000 0.3601 0.3862 0.4062 0.5333 0.5521 0.5767
#> a        0.4460 0.0310 20000 0.3452 0.3660 0.3846 0.5066 0.5249 0.5482
#> Y~~Y     1.0303 0.0459 20000 0.8799 0.9156 0.9403 1.1207 1.1460 1.1800
#> M~~M     0.9803 0.0440 20000 0.8380 0.8694 0.8941 1.0659 1.0972 1.1286
```

```
#> X~~X     1.0233 0.0000 20000 1.0233 1.0233 1.0233 1.0233 1.0233 1.0233
#> indirect 0.2096 0.0207 20000 0.1484 0.1596 0.1705 0.2510 0.2646 0.2801
#> direct   0.2791 0.0346 20000 0.1686 0.1906 0.2112 0.3465 0.3693 0.3911
#> total    0.4887 0.0349 20000 0.3777 0.3989 0.4210 0.5566 0.5798 0.6075
```

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        1.203402e-03 -4.536934e-04 -3.594844e-07  3.150057e-05 -1.086221e-05
#> b        -4.536934e-04  1.036303e-03 -5.975551e-06 -1.563021e-05  1.754877e-05
#> a        -3.594844e-07 -5.975551e-06  9.643442e-04  1.152135e-05  1.382227e-05
#> Y~~Y      3.150057e-05 -1.563021e-05  1.152135e-05  2.121793e-03 -1.359568e-05
#> M~~M     -1.086221e-05  1.754877e-05  1.382227e-05 -1.359568e-05  1.928527e-03
#> X~~X      1.537991e-05  5.468066e-06 -4.551201e-06  4.253013e-06 -1.177012e-05
#> indirect -2.025580e-04  4.593973e-04  4.508056e-04 -1.577140e-06  1.395944e-05
#> direct    1.203402e-03 -4.536934e-04 -3.594844e-07  3.150057e-05 -1.086221e-05
#> total     1.000843e-03  5.703959e-06  4.504461e-04  2.992344e-05  3.097230e-06
#>                    X~~X      indirect        direct        total
#> cp        1.537991e-05 -2.025580e-04  1.203402e-03 1.000843e-03
```

```
#> b         5.468066e-06  4.593973e-04 -4.536934e-04 5.703959e-06
#> a        -4.551201e-06  4.508056e-04 -3.594844e-07 4.504461e-04
#> Y~~Y      4.253013e-06 -1.577140e-06  3.150057e-05 2.992344e-05
#> M~~M     -1.177012e-05  1.395944e-05 -1.086221e-05 3.097230e-06
#> X~~X      2.074346e-03  4.134066e-08  1.537991e-05 1.542125e-05
#> indirect  4.134066e-08  4.179067e-04 -2.025580e-04 2.153487e-04
#> direct    1.537991e-05 -2.025580e-04  1.203402e-03 1.000843e-03
#> total     1.542125e-05  2.153487e-04  1.000843e-03 1.216192e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2312 0.0282 20000 0.1398 0.1593 0.1763 0.2863 0.3041 0.3228
#> b        0.4187 0.0268 20000 0.3294 0.3493 0.3658 0.4704 0.4872 0.5077
#> a        0.4147 0.0261 20000 0.3270 0.3463 0.3626 0.4651 0.4802 0.4964
#> Y~~Y     0.6909 0.0243 20000 0.6117 0.6261 0.6422 0.7369 0.7511 0.7691
#> M~~M     0.8281 0.0217 20000 0.7536 0.7694 0.7837 0.8685 0.8801 0.8931
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1736 0.0158 20000 0.1237 0.1347 0.1435 0.2052 0.2155 0.2304
#> direct   0.2312 0.0282 20000 0.1398 0.1593 0.1763 0.2863 0.3041 0.3228
#> total    0.4048 0.0263 20000 0.3202 0.3350 0.3518 0.4557 0.4721 0.4866
```

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