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       0.2668 0.0367 20000 0.1949 0.3387
#> b        0.5227 0.0326 20000 0.4585 0.5867
#> a        0.5264 0.0318 20000 0.4654 0.5887
#> Y~~Y     1.0653 0.0475 20000 0.9732 1.1585
#> M~~M     1.0122 0.0455 20000 0.9244 1.1018
```

```
#> X~~X     0.9952 0.0000 20000 0.9952 0.9952
#> indirect 0.2752 0.0239 20000 0.2299 0.3241
#> direct   0.2668 0.0367 20000 0.1949 0.3387
#> total    0.5420 0.0366 20000 0.4696 0.6143
```

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        1.357110e-03 -5.618857e-04 -6.919961e-07 -1.453077e-05 -2.444729e-05
#> b        -5.618857e-04  1.059243e-03 -6.539931e-06  9.224726e-06  8.692043e-06
#> a        -6.919961e-07 -6.539931e-06  1.007618e-03  1.732049e-05 -1.304118e-05
#> Y~~Y     -1.453077e-05  9.224726e-06  1.732049e-05  2.326754e-03 -7.172139e-06
#> M~~M     -2.444729e-05  8.692043e-06 -1.304118e-05 -7.172139e-06  2.062490e-03
#> X~~X      2.057378e-05 -4.763977e-06 -1.774932e-05  1.413271e-05 -3.394449e-06
#> indirect -2.962079e-04  5.543580e-04  5.233485e-04  1.469214e-05 -2.475263e-06
#> direct    1.357110e-03 -5.618857e-04 -6.919961e-07 -1.453077e-05 -2.444729e-05
#> total     1.060902e-03 -7.527665e-06  5.226565e-04  1.613713e-07 -2.692255e-05
#>                    X~~X      indirect        direct         total
#> cp        2.057378e-05 -2.962079e-04  1.357110e-03  1.060902e-03
```

```
#> b        -4.763977e-06  5.543580e-04 -5.618857e-04 -7.527665e-06
#> a        -1.774932e-05  5.233485e-04 -6.919961e-07  5.226565e-04
#> Y~~Y      1.413271e-05  1.469214e-05 -1.453077e-05  1.613713e-07
#> M~~M     -3.394449e-06 -2.475263e-06 -2.444729e-05 -2.692255e-05
#> X~~X      1.979062e-03 -1.195316e-05  2.057378e-05  8.620620e-06
#> indirect -1.195316e-05  5.665869e-04 -2.962079e-04  2.703790e-04
#> direct    2.057378e-05 -2.962079e-04  1.357110e-03  1.060902e-03
#> total     8.620620e-06  2.703790e-04  1.060902e-03  1.331281e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2082 0.0285 20000 0.1107 0.1340 0.1524 0.2639 0.2810 0.3017
#> b        0.4641 0.0265 20000 0.3772 0.3952 0.4112 0.5150 0.5307 0.5506
#> a        0.4627 0.0247 20000 0.3779 0.3985 0.4139 0.5106 0.5257 0.5428
#> Y~~Y     0.6519 0.0242 20000 0.5707 0.5886 0.6040 0.6980 0.7115 0.7275
#> M~~M     0.7859 0.0229 20000 0.7054 0.7236 0.7392 0.8287 0.8412 0.8572
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2147 0.0170 20000 0.1626 0.1725 0.1820 0.2485 0.2602 0.2760
#> direct   0.2082 0.0285 20000 0.1107 0.1340 0.1524 0.2639 0.2810 0.3017
#> total    0.4230 0.0259 20000 0.3322 0.3525 0.3711 0.4722 0.4884 0.5070
```

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