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:

```
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.

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       0.1894 0.0357 20000 0.0761 0.0987 0.1188 0.2593 0.2809 0.3001
#> b        0.5606 0.0319 20000 0.4603 0.4791 0.4985 0.6233 0.6418 0.6664
#> a        0.4672 0.0323 20000 0.3652 0.3857 0.4048 0.5309 0.5508 0.5741
#> Y~~Y     1.0368 0.0463 20000 0.8839 0.9162 0.9467 1.1281 1.1565 1.1905
#> M~~M     1.0216 0.0455 20000 0.8698 0.9055 0.9321 1.1104 1.1383 1.1716
#> indirect 0.2619 0.0234 20000 0.1914 0.2054 0.2182 0.3095 0.3255 0.3422
#> direct   0.1894 0.0357 20000 0.0761 0.0987 0.1188 0.2593 0.2809 0.3001
```

```
#> total    0.4513 0.0373 20000 0.3313 0.3548 0.3784 0.5252 0.5495 0.5713
```

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        1.266216e-03 -4.666719e-04  1.535392e-05 1.381602e-05 -2.481601e-06
#> b        -4.666719e-04  1.011975e-03 -1.848828e-06 7.454959e-07  8.337651e-06
#> a         1.535392e-05 -1.848828e-06  1.030451e-03 8.753073e-06 -1.458478e-06
#> Y~~Y      1.381602e-05  7.454959e-07  8.753073e-06 2.134122e-03  1.390519e-05
#> M~~M     -2.481601e-06  8.337651e-06 -1.458478e-06 1.390519e-05  2.116573e-03
#> X~~X      6.008656e-07  7.268163e-06 -1.533502e-05 3.506146e-06  5.084594e-06
#> indirect -2.094483e-04  4.720841e-04  5.769649e-04 5.174608e-06  3.398816e-06
#> direct    1.266216e-03 -4.666719e-04  1.535392e-05 1.381602e-05 -2.481601e-06
#> total     1.056768e-03  5.412189e-06  5.923188e-04 1.899063e-05  9.172150e-07
#>                    X~~X      indirect        direct         total
#> cp        6.008656e-07 -2.094483e-04  1.266216e-03  1.056768e-03
#> b         7.268163e-06  4.720841e-04 -4.666719e-04  5.412189e-06
#> a        -1.533502e-05  5.769649e-04  1.535392e-05  5.923188e-04
#> Y~~Y      3.506146e-06  5.174608e-06  1.381602e-05  1.899063e-05
```

```
#> M~~M      5.084594e-06  3.398816e-06 -2.481601e-06  9.172150e-07
#> X~~X      1.967215e-03 -5.271359e-06  6.008656e-07 -4.670494e-06
#> indirect -5.271359e-06  5.452643e-04 -2.094483e-04  3.358160e-04
#> direct    6.008656e-07 -2.094483e-04  1.266216e-03  1.056768e-03
#> total    -4.670494e-06  3.358160e-04  1.056768e-03  1.392584e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.1511 0.0281 20000 0.0602 0.0791 0.0952 0.2051 0.2231 0.2424
#> b         0.4995 0.0253 20000 0.4179 0.4338 0.4500 0.5488 0.5635 0.5807
#> a         0.4183 0.0260 20000 0.3282 0.3494 0.3661 0.4677 0.4831 0.5049
#> Y~~Y      0.6646 0.0243 20000 0.5796 0.6000 0.6157 0.7103 0.7248 0.7395
#> M~~M      0.8251 0.0217 20000 0.7451 0.7666 0.7813 0.8660 0.8779 0.8923
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2089 0.0171 20000 0.1576 0.1666 0.1759 0.2430 0.2540 0.2661
#> direct   0.1511 0.0281 20000 0.0602 0.0791 0.0952 0.2051 0.2231 0.2424
#> total    0.3600 0.0276 20000 0.2685 0.2863 0.3053 0.4129 0.4292 0.4464
```

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

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. (2022). *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