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       0.2690 0.0357 20000 0.1500 0.1755 0.1984 0.3390 0.3617 0.3799
#> b        0.5284 0.0316 20000 0.4280 0.4466 0.4664 0.5904 0.6090 0.6330
#> a        0.5183 0.0319 20000 0.4147 0.4359 0.4557 0.5809 0.6005 0.6223
#> Y~~Y     0.9578 0.0428 20000 0.8206 0.8479 0.8734 1.0415 1.0686 1.1008
#> M~~M     0.9577 0.0428 20000 0.8242 0.8477 0.8739 1.0418 1.0670 1.0911
#> indirect 0.2738 0.0235 20000 0.2026 0.2166 0.2296 0.3214 0.3368 0.3575
#> direct   0.2690 0.0357 20000 0.1500 0.1755 0.1984 0.3390 0.3617 0.3799
```

```
#> total    0.5428 0.0360 20000 0.4289 0.4512 0.4715 0.6124 0.6359 0.6545
```

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        1.277210e-03 -5.111609e-04  7.871041e-06  7.472683e-06 -8.292192e-06
#> b        -5.111609e-04  1.005297e-03 -3.136994e-06 -8.035740e-06  3.990959e-06
#> a         7.871041e-06 -3.136994e-06  1.028647e-03  7.762916e-06 -1.788864e-05
#> Y~~Y      7.472683e-06 -8.035740e-06  7.762916e-06  1.850647e-03 -1.914375e-05
#> M~~M     -8.292192e-06  3.990959e-06 -1.788864e-05 -1.914375e-05  1.854109e-03
#> X~~X     -1.266036e-05 -2.695049e-06 -6.572482e-06  1.545416e-05  1.823492e-06
#> indirect -2.602783e-04  5.193531e-04  5.419915e-04 -4.904122e-07 -7.299429e-06
#> direct    1.277210e-03 -5.111609e-04  7.871041e-06  7.472683e-06 -8.292192e-06
#> total     1.016931e-03  8.192170e-06  5.498625e-04  6.982271e-06 -1.559162e-05
#>                    X~~X      indirect        direct         total
#> cp       -1.266036e-05 -2.602783e-04  1.277210e-03  1.016931e-03
#> b        -2.695049e-06  5.193531e-04 -5.111609e-04  8.192170e-06
#> a        -6.572482e-06  5.419915e-04  7.871041e-06  5.498625e-04
#> Y~~Y      1.545416e-05 -4.904122e-07  7.472683e-06  6.982271e-06
```

```
#> M~~M      1.823492e-06 -7.299429e-06 -8.292192e-06 -1.559162e-05
#> X~~X      1.807505e-03 -5.255251e-06 -1.266036e-05 -1.791561e-05
#> indirect -5.255251e-06  5.566428e-04 -2.602783e-04  2.963646e-04
#> direct   -1.266036e-05 -2.602783e-04  1.277210e-03  1.016931e-03
#> total    -1.791561e-05  2.963646e-04  1.016931e-03  1.313296e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2134 0.0279 20000 0.1176 0.1391 0.1586 0.2674 0.2850 0.3033
#> b         0.4743 0.0260 20000 0.3898 0.4066 0.4226 0.5252 0.5395 0.5559
#> a         0.4581 0.0252 20000 0.3760 0.3920 0.4081 0.5070 0.5201 0.5355
#> Y~~Y      0.6367 0.0243 20000 0.5529 0.5735 0.5877 0.6832 0.6970 0.7142
#> M~~M      0.7901 0.0231 20000 0.7133 0.7295 0.7429 0.8334 0.8463 0.8587
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2173 0.0171 20000 0.1651 0.1750 0.1841 0.2515 0.2623 0.2723
#> direct   0.2134 0.0279 20000 0.1176 0.1391 0.1586 0.2674 0.2850 0.3033
#> total    0.4307 0.0258 20000 0.3441 0.3619 0.3786 0.4796 0.4939 0.5139
```

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