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       0.2623 0.0363 20000 0.1400 0.1677 0.1910 0.3327 0.3539 0.3786
#> b        0.4720 0.0323 20000 0.3717 0.3894 0.4089 0.5355 0.5556 0.5765
#> a        0.4942 0.0323 20000 0.3862 0.4109 0.4309 0.5566 0.5779 0.5987
#> Y~~Y     1.0574 0.0474 20000 0.9037 0.9359 0.9642 1.1506 1.1795 1.2096
#> M~~M     1.0202 0.0453 20000 0.8717 0.9036 0.9300 1.1085 1.1380 1.1746
```

```
#> indirect 0.2333 0.0221 20000 0.1684 0.1793 0.1914 0.2778 0.2932 0.3127
#> direct   0.2623 0.0363 20000 0.1400 0.1677 0.1910 0.3327 0.3539 0.3786
#> total    0.4955 0.0360 20000 0.3788 0.4009 0.4238 0.5650 0.5858 0.6093
```

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        1.318653e-03 -5.119521e-04 -1.107310e-05  1.794898e-07  5.885707e-06
#> b        -5.119521e-04  1.035579e-03  7.534770e-06 -6.309655e-06 -3.778587e-06
#> a        -1.107310e-05  7.534770e-06  1.044352e-03 -7.010224e-06  8.512614e-07
#> Y~~Y      1.794898e-07 -6.309655e-06 -7.010224e-06  2.254502e-03 -6.361850e-06
#> M~~M      5.885707e-06 -3.778587e-06  8.512614e-07 -6.361850e-06  2.074832e-03
#> X~~X      5.071303e-06 -1.894799e-05  1.745371e-06 -4.705401e-06 -3.782700e-06
#> indirect -2.581570e-04  5.154069e-04  4.970483e-04 -6.498307e-06 -1.800241e-06
#> direct    1.318653e-03 -5.119521e-04 -1.107310e-05  1.794898e-07  5.885707e-06
#> total     1.060496e-03  3.454797e-06  4.859752e-04 -6.318818e-06  4.085466e-06
#>                    X~~X      indirect        direct         total
#> cp        5.071303e-06 -2.581570e-04  1.318653e-03  1.060496e-03
#> b        -1.894799e-05  5.154069e-04 -5.119521e-04  3.454797e-06
```

```
#>          1.745371e-06  4.970483e-04 -1.107310e-05  4.859752e-04
#> Y~~Y     -4.705401e-06 -6.498307e-06  1.794898e-07 -6.318818e-06
#> M~~M     -3.782700e-06 -1.800241e-06  5.885707e-06  4.085466e-06
#> X~~X      1.945289e-03 -8.750248e-06  5.071303e-06 -3.678945e-06
#> indirect -8.750248e-06  4.906068e-04 -2.581570e-04  2.324498e-04
#> direct    5.071303e-06 -2.581570e-04  1.318653e-03  1.060496e-03
#> total    -3.678945e-06  2.324498e-04  1.060496e-03  1.292946e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2106 0.0288 20000 0.1121 0.1349 0.1537 0.2664 0.2827 0.3032
#> b        0.4290 0.0271 20000 0.3373 0.3588 0.3751 0.4809 0.4978 0.5180
#> a        0.4367 0.0257 20000 0.3487 0.3685 0.3862 0.4864 0.5031 0.5194
#> Y~~Y     0.6927 0.0242 20000 0.6083 0.6282 0.6441 0.7389 0.7522 0.7697
#> M~~M     0.8093 0.0224 20000 0.7303 0.7469 0.7634 0.8509 0.8642 0.8784
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1873 0.0165 20000 0.1363 0.1463 0.1556 0.2203 0.2315 0.2460
#> direct   0.2106 0.0288 20000 0.1121 0.1349 0.1537 0.2664 0.2827 0.3032
#> total    0.3979 0.0265 20000 0.3107 0.3294 0.3447 0.4486 0.4645 0.4827
```

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