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       0.2069 0.0371 20000 0.0859 0.1130 0.1349 0.2801 0.3036 0.3355
#> b        0.5095 0.0318 20000 0.4050 0.4294 0.4470 0.5721 0.5909 0.6141
#> a        0.5317 0.0324 20000 0.4236 0.4482 0.4683 0.5952 0.6145 0.6385
#> Y~~Y     0.9847 0.0442 20000 0.8469 0.8713 0.8981 1.0696 1.0955 1.1339
#> M~~M     0.9850 0.0442 20000 0.8396 0.8709 0.8980 1.0715 1.0979 1.1296
```

```
#> indirect 0.2709 0.0236 20000 0.1993 0.2130 0.2260 0.3187 0.3345 0.3549
#> direct   0.2069 0.0371 20000 0.0859 0.1130 0.1349 0.2801 0.3036 0.3355
#> total    0.4778 0.0369 20000 0.3543 0.3839 0.4058 0.5504 0.5754 0.6020
```

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        1.320807e-03 -5.101985e-04 -7.139319e-06 -1.907627e-05  1.148097e-05
#> b        -5.101985e-04  9.805853e-04  8.833207e-06  2.209975e-05  7.481908e-06
#> a        -7.139319e-06  8.833207e-06  1.058401e-03 -1.651300e-05  4.555547e-06
#> Y~~Y     -1.907627e-05  2.209975e-05 -1.651300e-05  1.931059e-03 -2.939528e-06
#> M~~M      1.148097e-05  7.481908e-06  4.555547e-06 -2.939528e-06  1.943139e-03
#> X~~X     -3.810892e-06 -3.220538e-06  7.961345e-06  6.171802e-06  1.168444e-05
#> indirect -2.749068e-04  5.258204e-04  5.440635e-04  3.686878e-06  6.686393e-06
#> direct    1.320807e-03 -5.101985e-04 -7.139319e-06 -1.907627e-05  1.148097e-05
#> total     1.045900e-03  1.562191e-05  5.369242e-04 -1.538939e-05  1.816736e-05
#>                    X~~X      indirect        direct         total
#> cp       -3.810892e-06 -2.749068e-04  1.320807e-03  1.045900e-03
#> b        -3.220538e-06  5.258204e-04 -5.101985e-04  1.562191e-05
```

```
#> a         7.961345e-06  5.440635e-04 -7.139319e-06  5.369242e-04
#> Y~~Y      6.171802e-06  3.686878e-06 -1.907627e-05 -1.538939e-05
#> M~~M      1.168444e-05  6.686393e-06  1.148097e-05  1.816736e-05
#> X~~X      1.729626e-03  1.691655e-06 -3.810892e-06 -2.119237e-06
#> indirect  1.691655e-06  5.578428e-04 -2.749068e-04  2.829360e-04
#> direct   -3.810892e-06 -2.749068e-04  1.320807e-03  1.045900e-03
#> total    -2.119237e-06  2.829360e-04  1.045900e-03  1.328836e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1652 0.0288 20000 0.0662 0.0913 0.1075 0.2206 0.2390 0.2633
#> b        0.4721 0.0264 20000 0.3851 0.4027 0.4195 0.5227 0.5390 0.5580
#> a        0.4582 0.0250 20000 0.3730 0.3912 0.4086 0.5058 0.5213 0.5371
#> Y~~Y     0.6783 0.0243 20000 0.5984 0.6151 0.6296 0.7246 0.7398 0.7563
#> M~~M     0.7901 0.0228 20000 0.7115 0.7283 0.7441 0.8331 0.8470 0.8609
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2163 0.0174 20000 0.1633 0.1727 0.1826 0.2508 0.2626 0.2751
#> direct   0.1652 0.0288 20000 0.0662 0.0913 0.1075 0.2206 0.2390 0.2633
#> total    0.3815 0.0269 20000 0.2862 0.3092 0.3279 0.4332 0.4486 0.4658
```

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