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       0.2362 0.0344 20000 0.1680 0.3030
#> b        0.5420 0.0328 20000 0.4776 0.6061
#> a        0.4683 0.0303 20000 0.4090 0.5273
#> Y~~Y     1.0381 0.0466 20000 0.9469 1.1297
#> M~~M     0.9749 0.0437 20000 0.8901 1.0605
```

```
#> X~~X     1.0764 0.0000 20000 1.0764 1.0764
#> indirect 0.2538 0.0224 20000 0.2104 0.2986
#> direct   0.2362 0.0344 20000 0.1680 0.3030
#> total    0.4900 0.0348 20000 0.4209 0.5576
```

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        1.190367e-03 -4.938526e-04 -6.386108e-06  9.354024e-06  1.006420e-05
#> b        -4.938526e-04  1.060278e-03  1.770546e-06  4.387645e-07 -1.962630e-06
#> a        -6.386108e-06  1.770546e-06  9.168113e-04  1.943664e-06 -1.128838e-05
#> Y~~Y      9.354024e-06  4.387645e-07  1.943664e-06  2.179413e-03 -2.506892e-05
#> M~~M      1.006420e-05 -1.962630e-06 -1.128838e-05 -2.506892e-05  1.937583e-03
#> X~~X     -1.248721e-05 -1.200834e-05  1.317887e-06  9.259408e-07 -8.140195e-07
#> indirect -2.350371e-04  4.973844e-04  4.976896e-04  1.729489e-06 -6.342601e-06
#> direct    1.190367e-03 -4.938526e-04 -6.386108e-06  9.354024e-06  1.006420e-05
#> total     9.553295e-04  3.531753e-06  4.913035e-04  1.108351e-05  3.721598e-06
#>                   X~~X      indirect        direct         total
#> cp       -1.248721e-05 -2.350371e-04  1.190367e-03  9.553295e-04
```

```
#> b        -1.200834e-05  4.973844e-04 -4.938526e-04  3.531753e-06
#> a         1.317887e-06  4.976896e-04 -6.386108e-06  4.913035e-04
#> Y~~Y      9.259408e-07  1.729489e-06  9.354024e-06  1.108351e-05
#> M~~M     -8.140195e-07 -6.342601e-06  1.006420e-05  3.721598e-06
#> X~~X      2.256622e-03 -5.195606e-06 -1.248721e-05 -1.768281e-05
#> indirect -5.195606e-06  5.035523e-04 -2.350371e-04  2.685153e-04
#> direct   -1.248721e-05 -2.350371e-04  1.190367e-03  9.553295e-04
#> total    -1.768281e-05  2.685153e-04  9.553295e-04  1.223845e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1948 0.0281 20000 0.1058 0.1224 0.1391 0.2492 0.2660 0.2857
#> b        0.4741 0.0259 20000 0.3867 0.4065 0.4233 0.5246 0.5402 0.5574
#> a        0.4415 0.0256 20000 0.3533 0.3734 0.3902 0.4906 0.5068 0.5236
#> Y~~Y     0.6558 0.0242 20000 0.5759 0.5916 0.6077 0.7026 0.7166 0.7303
#> M~~M     0.8051 0.0226 20000 0.7258 0.7432 0.7593 0.8478 0.8606 0.8752
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2093 0.0170 20000 0.1539 0.1666 0.1764 0.2427 0.2535 0.2651
#> direct   0.1948 0.0281 20000 0.1058 0.1224 0.1391 0.2492 0.2660 0.2857
#> total    0.4041 0.0262 20000 0.3148 0.3351 0.3516 0.4537 0.4697 0.4884
```

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*, *56*(3), 1678–1696. 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. (2024). *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