semmcci: Monte Carlo Confidence Intervals

Ivan Jacob Agaloos Pesigan

Installation

You can install the CRAN release of `semmcci` with:

```r
install.packages("semmcci")
```

You can install the development version of `semmcci` from GitHub with:

```r
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        0.1984 0.0365 20000 0.1256 0.2697
#> b         0.4965 0.0327 20000 0.4324 0.5603
#> a         0.4876 0.0321 20000 0.4255 0.5503
#> Y~~Y      1.0526 0.0470 20000 0.9606 1.1451
#> M~~M      0.9926 0.0443 20000 0.9048 1.0781
```

```
#> X~~X     0.9622 0.0000 20000 0.9622 0.9622
#> indirect 0.2421 0.0226 20000 0.1994 0.2877
#> direct   0.1984 0.0365 20000 0.1256 0.2697
#> total    0.4405 0.0366 20000 0.3678 0.5116
```

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        1.345152e-03 -4.987621e-04  4.296464e-06  7.461789e-06  1.145404e-05
#> b        -4.987621e-04  1.062507e-03  6.897070e-06  1.931659e-05  9.118204e-06
#> a         4.296464e-06  6.897070e-06  1.040895e-03  6.827452e-06 -2.044276e-06
#> Y~~Y      7.461789e-06  1.931659e-05  6.827452e-06  2.194004e-03 -1.098407e-05
#> M~~M      1.145404e-05  9.118204e-06 -2.044276e-06 -1.098407e-05  1.962270e-03
#> X~~X     -6.530334e-07  3.485741e-06 -2.427045e-06 -9.146847e-06  1.164537e-05
#> indirect -2.411409e-04  5.216009e-04  5.197446e-04  1.264685e-05  3.238004e-06
#> direct    1.345152e-03 -4.987621e-04  4.296464e-06  7.461789e-06  1.145404e-05
#> total     1.104011e-03  2.283887e-05  5.240411e-04  2.010864e-05  1.469204e-05
#>                   X~~X      indirect        direct         total
#> cp       -6.530334e-07 -2.411409e-04  1.345152e-03  1.104011e-03
```

```
#> b         3.485741e-06  5.216009e-04 -4.987621e-04  2.283887e-05
#> a        -2.427045e-06  5.197446e-04  4.296464e-06  5.240411e-04
#> Y~~Y     -9.146847e-06  1.264685e-05  7.461789e-06  2.010864e-05
#> M~~M      1.164537e-05  3.238004e-06  1.145404e-05  1.469204e-05
#> X~~X      1.869306e-03 -2.544658e-07 -6.530334e-07 -9.074992e-07
#> indirect -2.544658e-07  5.133276e-04 -2.411409e-04  2.721868e-04
#> direct   -6.530334e-07 -2.411409e-04  1.345152e-03  1.104011e-03
#> total    -9.074992e-07  2.721868e-04  1.104011e-03  1.376197e-03
```

```
MCStd(unstd)
```

```
#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1597 0.0292 20000 0.0657 0.0827 0.1019 0.2163 0.2354 0.2561
#> b        0.4505 0.0270 20000 0.3592 0.3786 0.3961 0.5025 0.5171 0.5372
#> a        0.4328 0.0258 20000 0.3413 0.3646 0.3812 0.4829 0.4978 0.5170
#> Y~~Y     0.7093 0.0243 20000 0.6267 0.6450 0.6606 0.7560 0.7708 0.7853
#> M~~M     0.8127 0.0223 20000 0.7328 0.7522 0.7668 0.8547 0.8671 0.8835
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1950 0.0169 20000 0.1413 0.1522 0.1622 0.2284 0.2397 0.2520
#> direct   0.1597 0.0292 20000 0.0657 0.0827 0.1019 0.2163 0.2354 0.2561
#> total    0.3547 0.0278 20000 0.2578 0.2811 0.2989 0.4083 0.4243 0.4441
```

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