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       0.2410 0.0351 20000 0.1246 0.1539 0.1728 0.3102 0.3330 0.3576
#> b        0.4950 0.0337 20000 0.3827 0.4069 0.4288 0.5609 0.5801 0.6061
#> a        0.4842 0.0289 20000 0.3878 0.4094 0.4281 0.5403 0.5575 0.5796
#> Y~~Y     1.0290 0.0459 20000 0.8794 0.9079 0.9383 1.1183 1.1478 1.1799
#> M~~M     0.9071 0.0404 20000 0.7753 0.8023 0.8277 0.9862 1.0111 1.0412
```

```
#> X~~X     1.0922 0.0000 20000 1.0922 1.0922 1.0922 1.0922 1.0922 1.0922
#> indirect 0.2397 0.0217 20000 0.1731 0.1860 0.1985 0.2837 0.2983 0.3164
#> direct   0.2410 0.0351 20000 0.1246 0.1539 0.1728 0.3102 0.3330 0.3576
#> total    0.4807 0.0341 20000 0.3706 0.3931 0.4137 0.5476 0.5711 0.5955
```

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        1.205369e-03 -5.513440e-04 -1.773341e-06  7.346263e-06 -1.661458e-05
#> b        -5.513440e-04  1.125990e-03 -1.834131e-06 -9.188814e-06  1.115866e-05
#> a        -1.773341e-06 -1.834131e-06  8.418177e-04 -4.212888e-06  7.120267e-06
#> Y~~Y      7.346263e-06 -9.188814e-06 -4.212888e-06  2.136802e-03  1.494283e-05
#> M~~M     -1.661458e-05  1.115866e-05  7.120267e-06  1.494283e-05  1.661214e-03
#> X~~X      1.752276e-05 -2.980629e-06  1.668780e-06  3.441732e-06  2.724278e-06
#> indirect -2.673255e-04  5.440834e-04  4.157872e-04 -6.563684e-06  9.075874e-06
#> direct    1.205369e-03 -5.513440e-04 -1.773341e-06  7.346263e-06 -1.661458e-05
#> total     9.380432e-04 -7.260615e-06  4.140139e-04  7.825788e-07 -7.538704e-06
#>                    X~~X      indirect        direct         total
#> cp        1.752276e-05 -2.673255e-04  1.205369e-03  9.380432e-04
```

```
#> b         -2.980629e-06  5.440834e-04 -5.513440e-04 -7.260615e-06
#> a          1.668780e-06  4.157872e-04 -1.773341e-06  4.140139e-04
#> Y~~Y       3.441732e-06 -6.563684e-06  7.346263e-06  7.825788e-07
#> M~~M       2.724278e-06  9.075874e-06 -1.661458e-05 -7.538704e-06
#> X~~X       2.313048e-03 -4.055811e-07  1.752276e-05  1.711718e-05
#> indirect -4.055811e-07  4.700787e-04 -2.673255e-04  2.027532e-04
#> direct     1.752276e-05 -2.673255e-04  1.205369e-03  9.380432e-04
#> total      1.711718e-05  2.027532e-04  9.380432e-04  1.140796e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2054 0.0292 20000 0.1100 0.1298 0.1483 0.2622 0.2796 0.3062
#> b         0.4354 0.0275 20000 0.3435 0.3620 0.3809 0.4886 0.5053 0.5225
#> a         0.4692 0.0247 20000 0.3861 0.4055 0.4203 0.5167 0.5309 0.5462
#> Y~~Y      0.6844 0.0242 20000 0.6040 0.6204 0.6350 0.7303 0.7439 0.7593
#> M~~M      0.7798 0.0232 20000 0.7016 0.7182 0.7330 0.8233 0.8355 0.8509
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2043 0.0172 20000 0.1513 0.1615 0.1711 0.2384 0.2496 0.2622
#> direct   0.2054 0.0292 20000 0.1100 0.1298 0.1483 0.2622 0.2796 0.3062
#> total    0.4097 0.0263 20000 0.3219 0.3416 0.3566 0.4600 0.4755 0.4918
```

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