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       0.2590 0.0344 20000 0.1532 0.1733 0.1923 0.3269 0.3486 0.3676
#> b        0.4928 0.0319 20000 0.3903 0.4105 0.4308 0.5555 0.5753 0.6001
#> a        0.4937 0.0304 20000 0.3962 0.4174 0.4346 0.5539 0.5742 0.5986
#> Y~~Y     1.0082 0.0449 20000 0.8605 0.8939 0.9214 1.0967 1.1233 1.1543
#> M~~M     0.9838 0.0438 20000 0.8433 0.8720 0.8981 1.0696 1.0959 1.1241
```

```
#> X~~X     1.0837 0.0000 20000 1.0837 1.0837 1.0837 1.0837 1.0837 1.0837
#> indirect 0.2433 0.0217 20000 0.1808 0.1915 0.2028 0.2873 0.3018 0.3234
#> direct   0.2590 0.0344 20000 0.1532 0.1733 0.1923 0.3269 0.3486 0.3676
#> total    0.5023 0.0341 20000 0.3913 0.4155 0.4355 0.5694 0.5902 0.6174
```

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        1.198379e-03 -5.300056e-04 -1.788801e-05  3.863624e-06 -3.387637e-06
#> b        -5.300056e-04  1.034532e-03  7.141936e-06  6.766577e-06 -5.696204e-06
#> a        -1.788801e-05  7.141936e-06  8.893659e-04 -5.458532e-06 -5.639534e-06
#> Y~~Y      3.863624e-06  6.766577e-06 -5.458532e-06  2.034761e-03  2.605208e-05
#> M~~M     -3.387637e-06 -5.696204e-06 -5.639534e-06  2.605208e-05  1.956784e-03
#> X~~X     -9.899336e-07 -1.828176e-05 -1.046700e-06 -6.742779e-06  1.574282e-06
#> indirect -2.705441e-04  5.144656e-04  4.415846e-04  4.526365e-07 -5.357530e-06
#> direct    1.198379e-03 -5.300056e-04 -1.788801e-05  3.863624e-06 -3.387637e-06
#> total     9.278344e-04 -1.554007e-05  4.236966e-04  4.316260e-06 -8.745167e-06
#>                    X~~X      indirect        direct         total
#> cp       -9.899336e-07 -2.705441e-04  1.198379e-03  9.278344e-04
```

```
#> b         -1.828176e-05  5.144656e-04 -5.300056e-04 -1.554007e-05
#> a         -1.046700e-06  4.415846e-04 -1.788801e-05  4.236966e-04
#> Y~~Y      -6.742779e-06  4.526365e-07  3.863624e-06  4.316260e-06
#> M~~M       1.574282e-06 -5.357530e-06 -3.387637e-06 -8.745167e-06
#> X~~X       2.344370e-03 -9.519600e-06 -9.899336e-07 -1.050953e-05
#> indirect -9.519600e-06  4.725287e-04 -2.705441e-04  2.019846e-04
#> direct   -9.899336e-07 -2.705441e-04  1.198379e-03  9.278344e-04
#> total    -1.050953e-05  2.019846e-04  9.278344e-04  1.129819e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2186 0.0288 20000 0.1204 0.1439 0.1617 0.2747 0.2934 0.3127
#> b         0.4464 0.0269 20000 0.3537 0.3748 0.3925 0.4984 0.5147 0.5314
#> a         0.4601 0.0248 20000 0.3739 0.3953 0.4111 0.5078 0.5230 0.5374
#> Y~~Y      0.6631 0.0240 20000 0.5843 0.6006 0.6151 0.7091 0.7239 0.7382
#> M~~M      0.7883 0.0228 20000 0.7112 0.7265 0.7421 0.8310 0.8437 0.8602
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2054 0.0170 20000 0.1512 0.1628 0.1729 0.2396 0.2500 0.2635
#> direct   0.2186 0.0288 20000 0.1204 0.1439 0.1617 0.2747 0.2934 0.3127
#> total    0.4240 0.0258 20000 0.3358 0.3561 0.3719 0.4739 0.4880 0.5052
```

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