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       0.2855 0.0359 20000 0.1696 0.1937 0.2157 0.3555 0.3769 0.4056
#> b        0.4890 0.0321 20000 0.3865 0.4057 0.4276 0.5530 0.5723 0.5945
#> a        0.4994 0.0314 20000 0.3959 0.4191 0.4386 0.5605 0.5805 0.6045
#> Y~~Y     1.0106 0.0451 20000 0.8592 0.8927 0.9221 1.0988 1.1280 1.1577
#> M~~M     0.9860 0.0439 20000 0.8470 0.8748 0.9005 1.0728 1.0991 1.1271
```

```
#> X~~X     1.0021 0.0000 20000 1.0021 1.0021 1.0021 1.0021 1.0021 1.0021
#> indirect 0.2442 0.0223 20000 0.1757 0.1896 0.2026 0.2900 0.3047 0.3283
#> direct   0.2855 0.0359 20000 0.1696 0.1937 0.2157 0.3555 0.3769 0.4056
#> total    0.5297 0.0357 20000 0.4123 0.4396 0.4601 0.5995 0.6193 0.6485
```

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        1.241551e-03 -5.001543e-04  3.284650e-06 -7.801464e-06 -1.821984e-05
#> b        -5.001543e-04  1.020973e-03 -1.692593e-06  4.491019e-06  7.210398e-07
#> a         3.284650e-06 -1.692593e-06  9.786142e-04  5.303121e-06 -1.827256e-05
#> Y~~Y     -7.801464e-06  4.491019e-06  5.303121e-06  2.053431e-03  4.588313e-06
#> M~~M     -1.821984e-05  7.210398e-07 -1.827256e-05  4.588313e-06  1.954772e-03
#> X~~X     -5.578129e-06  3.384436e-06 -2.929718e-05 -1.803008e-05 -6.216278e-06
#> indirect -2.486892e-04  5.092160e-04  4.773199e-04  5.009233e-06 -8.295345e-06
#> direct    1.241551e-03 -5.001543e-04  3.284650e-06 -7.801464e-06 -1.821984e-05
#> total     9.928615e-04  9.061663e-06  4.806046e-04 -2.792231e-06 -2.651518e-05
#>                    X~~X      indirect        direct         total
#> cp       -5.578129e-06 -2.486892e-04  1.241551e-03  9.928615e-04
```

```
#> b          3.384436e-06  5.092160e-04 -5.001543e-04  9.061663e-06
#> a         -2.929718e-05  4.773199e-04  3.284650e-06  4.806046e-04
#> Y~~Y      -1.803008e-05  5.009233e-06 -7.801464e-06 -2.792231e-06
#> M~~M      -6.216278e-06 -8.295345e-06 -1.821984e-05 -2.651518e-05
#> X~~X       2.018269e-03 -1.274144e-05 -5.578129e-06 -1.831957e-05
#> indirect -1.274144e-05  4.886052e-04 -2.486892e-04  2.399160e-04
#> direct   -5.578129e-06 -2.486892e-04  1.241551e-03  9.928615e-04
#> total    -1.831957e-05  2.399160e-04  9.928615e-04  1.232777e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2312 0.0281 20000 0.1408 0.1585 0.1760 0.2861 0.3036 0.3227
#> b        0.4398 0.0265 20000 0.3509 0.3701 0.3870 0.4910 0.5065 0.5223
#> a        0.4497 0.0251 20000 0.3650 0.3836 0.3999 0.4986 0.5127 0.5294
#> Y~~Y     0.6616 0.0243 20000 0.5799 0.5990 0.6135 0.7083 0.7240 0.7408
#> M~~M     0.7978 0.0226 20000 0.7198 0.7372 0.7514 0.8400 0.8529 0.8668
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1978 0.0165 20000 0.1441 0.1563 0.1661 0.2309 0.2409 0.2529
#> direct   0.2312 0.0281 20000 0.1408 0.1585 0.1760 0.2861 0.3036 0.3227
#> total    0.4290 0.0257 20000 0.3420 0.3600 0.3776 0.4775 0.4934 0.5111
```

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