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:

```
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.

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       0.2130 0.0347 20000 0.1006 0.1247 0.1443 0.2804 0.3006 0.3240
#> b        0.5285 0.0320 20000 0.4248 0.4469 0.4656 0.5912 0.6110 0.6341
#> a        0.5063 0.0304 20000 0.4072 0.4279 0.4448 0.5647 0.5823 0.6041
#> Y~~Y     0.9439 0.0427 20000 0.8080 0.8363 0.8611 1.0280 1.0571 1.0840
#> M~~M     0.9310 0.0413 20000 0.7982 0.8233 0.8498 1.0116 1.0380 1.0672
#> indirect 0.2675 0.0229 20000 0.1957 0.2103 0.2232 0.3130 0.3295 0.3483
#> direct   0.2130 0.0347 20000 0.1006 0.1247 0.1443 0.2804 0.3006 0.3240
```

```
#> total    0.4805 0.0347 20000 0.3624 0.3923 0.4115 0.5470 0.5680 0.5941
```

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        1.207978e-03 -5.269591e-04 -3.330876e-06 -6.498779e-07 -2.136559e-05
#> b        -5.269591e-04  1.027699e-03 -2.245083e-06  2.388242e-06 -3.971980e-06
#> a        -3.330876e-06 -2.245083e-06  9.263742e-04 -1.086050e-05 -1.893709e-05
#> Y~~Y     -6.498779e-07  2.388242e-06 -1.086050e-05  1.815576e-03 -7.010814e-07
#> M~~M     -2.136559e-05 -3.971980e-06 -1.893709e-05 -7.010814e-07  1.719520e-03
#> X~~X      1.591856e-06  9.135584e-06  1.074797e-05  3.913913e-05 -2.019402e-05
#> indirect -2.688977e-04  5.197272e-04  4.881452e-04 -4.181016e-06 -1.186838e-05
#> direct    1.207978e-03 -5.269591e-04 -3.330876e-06 -6.498779e-07 -2.136559e-05
#> total     9.390798e-04 -7.231865e-06  4.848143e-04 -4.830894e-06 -3.323397e-05
#>                    X~~X      indirect        direct         total
#> cp        1.591856e-06 -2.688977e-04  1.207978e-03  9.390798e-04
#> b         9.135584e-06  5.197272e-04 -5.269591e-04 -7.231865e-06
#> a         1.074797e-05  4.881452e-04 -3.330876e-06  4.848143e-04
#> Y~~Y      3.913913e-05 -4.181016e-06 -6.498779e-07 -4.830894e-06
```

```
#> M~~M     -2.019402e-05 -1.186838e-05 -2.136559e-05 -3.323397e-05
#> X~~X      2.032656e-03  1.022811e-05  1.591856e-06  1.181997e-05
#> indirect  1.022811e-05  5.221965e-04 -2.688977e-04  2.532988e-04
#> direct    1.591856e-06 -2.688977e-04  1.207978e-03  9.390798e-04
#> total     1.181997e-05  2.532988e-04  9.390798e-04  1.192379e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.1783 0.0288 20000 0.0824 0.1033 0.1211 0.2346 0.2516 0.2690
#> b         0.4808 0.0265 20000 0.3946 0.4121 0.4283 0.5317 0.5469 0.5643
#> a         0.4658 0.0250 20000 0.3852 0.4008 0.4165 0.5140 0.5284 0.5449
#> Y~~Y      0.6572 0.0243 20000 0.5773 0.5935 0.6088 0.7035 0.7175 0.7319
#> M~~M      0.7830 0.0232 20000 0.7031 0.7208 0.7358 0.8265 0.8394 0.8516
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2240 0.0176 20000 0.1697 0.1795 0.1901 0.2594 0.2704 0.2843
#> direct    0.1783 0.0288 20000 0.0824 0.1033 0.1211 0.2346 0.2516 0.2690
#> total     0.4022 0.0266 20000 0.3146 0.3318 0.3487 0.4533 0.4695 0.4880
```

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

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. (2022). *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