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.

```r
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`.

```r
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.

```r
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       0.2061 0.0358 20000 0.0943 0.1129 0.1350 0.2761 0.2972 0.3253
#> b        0.4967 0.0320 20000 0.3905 0.4144 0.4342 0.5604 0.5786 0.6022
#> a        0.5328 0.0308 20000 0.4288 0.4526 0.4733 0.5938 0.6127 0.6403
#> Y~~Y     1.0615 0.0478 20000 0.9114 0.9382 0.9672 1.1548 1.1841 1.2206
#> M~~M     1.0155 0.0452 20000 0.8622 0.8976 0.9256 1.1029 1.1316 1.1608
#> indirect 0.2646 0.0229 20000 0.1947 0.2089 0.2218 0.3109 0.3271 0.3457
#> direct   0.2061 0.0358 20000 0.0943 0.1129 0.1350 0.2761 0.2972 0.3253
```

```
#> total    0.4708 0.0351 20000 0.3555 0.3806 0.4020 0.5390 0.5600 0.5862
```

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        1.312268e-03 -5.650973e-04 -1.424410e-05  1.864110e-06 -5.983865e-06
#> b        -5.650973e-04  1.053034e-03  7.971301e-06  8.632495e-06  1.845429e-06
#> a        -1.424410e-05  7.971301e-06  9.484151e-04 -3.507583e-06 -4.259452e-06
#> Y~~Y      1.864110e-06  8.632495e-06 -3.507583e-06  2.268012e-03  9.472876e-06
#> M~~M     -5.983865e-06  1.845429e-06 -4.259452e-06  9.472876e-06  2.045050e-03
#> X~~X      2.456940e-05 -1.900428e-05  5.066493e-06 -5.862455e-06 -1.252983e-06
#> indirect -3.082891e-04  5.649850e-04  4.747212e-04  2.704147e-06 -1.090698e-06
#> direct    1.312268e-03 -5.650973e-04 -1.424410e-05  1.864110e-06 -5.983865e-06
#> total     1.003979e-03 -1.123689e-07  4.604771e-04  4.568257e-06 -7.074563e-06
#>                   X~~X      indirect        direct         total
#> cp        2.456940e-05 -3.082891e-04  1.312268e-03  1.003979e-03
#> b        -1.900428e-05  5.649850e-04 -5.650973e-04 -1.123689e-07
#> a         5.066493e-06  4.747212e-04 -1.424410e-05  4.604771e-04
#> Y~~Y     -5.862455e-06  2.704147e-06  1.864110e-06  4.568257e-06
```

```
#> M~~M     -1.252983e-06 -1.090698e-06 -5.983865e-06 -7.074563e-06
#> X~~X      2.313314e-03 -7.366499e-06  2.456940e-05  1.720291e-05
#> indirect -7.366499e-06  5.374982e-04 -3.082891e-04  2.292091e-04
#> direct    2.456940e-05 -3.082891e-04  1.312268e-03  1.003979e-03
#> total     1.720291e-05  2.292091e-04  1.003979e-03  1.233188e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1712 0.0299 20000 0.0696 0.0955 0.1134 0.2307 0.2485 0.2700
#> b        0.4583 0.0275 20000 0.3603 0.3855 0.4035 0.5107 0.5274 0.5450
#> a        0.4797 0.0244 20000 0.3945 0.4163 0.4310 0.5264 0.5418 0.5570
#> Y~~Y     0.6853 0.0243 20000 0.6033 0.6209 0.6365 0.7315 0.7444 0.7610
#> M~~M     0.7699 0.0234 20000 0.6898 0.7065 0.7229 0.8142 0.8267 0.8444
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2199 0.0178 20000 0.1643 0.1747 0.1852 0.2547 0.2673 0.2807
#> direct   0.1712 0.0299 20000 0.0696 0.0955 0.1134 0.2307 0.2485 0.2700
#> total    0.3911 0.0269 20000 0.2996 0.3210 0.3370 0.4430 0.4589 0.4768
```

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