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       0.2505 0.0350 20000 0.1349 0.1609 0.1822 0.3194 0.3409 0.3726
#> b        0.4366 0.0311 20000 0.3395 0.3565 0.3756 0.4974 0.5178 0.5380
#> a        0.4668 0.0322 20000 0.3590 0.3836 0.4036 0.5298 0.5502 0.5727
#> Y~~Y     0.9870 0.0442 20000 0.8413 0.8741 0.8998 1.0734 1.0990 1.1306
#> M~~M     1.0147 0.0455 20000 0.8716 0.8971 0.9253 1.1039 1.1314 1.1602
#> indirect 0.2038 0.0201 20000 0.1406 0.1557 0.1661 0.2446 0.2583 0.2742
#> direct   0.2505 0.0350 20000 0.1349 0.1609 0.1822 0.3194 0.3409 0.3726
```

```
#> total    0.4543 0.0347 20000 0.3419 0.3657 0.3865 0.5226 0.5428 0.5681
```

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        1.219044e-03 -4.506375e-04 2.758696e-07 -2.738422e-06  3.286574e-06
#> b        -4.506375e-04  9.771294e-04 1.639338e-06 -9.609944e-07 -3.170759e-06
#> a         2.758696e-07  1.639338e-06 1.028202e-03  7.939729e-06  1.216798e-05
#> Y~~Y     -2.738422e-06 -9.609944e-07 7.939729e-06  1.915568e-03  2.229114e-05
#> M~~M      3.286574e-06 -3.170759e-06 1.216798e-05  2.229114e-05  2.031686e-03
#> X~~X      1.720681e-05 -2.083577e-05 5.169536e-06  3.866271e-06 -1.036414e-06
#> indirect -2.104903e-04  4.570715e-04 4.497368e-04  2.380664e-06  3.704809e-06
#> direct    1.219044e-03 -4.506375e-04 2.758696e-07 -2.738422e-06  3.286574e-06
#> total     1.008554e-03  6.434004e-06 4.500127e-04 -3.577573e-07  6.991383e-06
#>                    X~~X      indirect        direct         total
#> cp        1.720681e-05 -2.104903e-04  1.219044e-03  1.008554e-03
#> b        -2.083577e-05  4.570715e-04 -4.506375e-04  6.434004e-06
#> a         5.169536e-06  4.497368e-04  2.758696e-07  4.500127e-04
#> Y~~Y      3.866271e-06  2.380664e-06 -2.738422e-06 -3.577573e-07
```

```
#> M~~M     -1.036414e-06  3.704809e-06  3.286574e-06  6.991383e-06
#> X~~X      1.960047e-03 -7.187499e-06  1.720681e-05  1.001931e-05
#> indirect -7.187499e-06  4.108234e-04 -2.104903e-04  2.003331e-04
#> direct    1.720681e-05 -2.104903e-04  1.219044e-03  1.008554e-03
#> total     1.001931e-05  2.003331e-04  1.008554e-03  1.208887e-03
```

```
MCStd(unstd)
```

```
#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2109 0.0290 20000 0.1166 0.1355 0.1533 0.2679 0.2871 0.3114
#> b        0.4115 0.0273 20000 0.3198 0.3393 0.3574 0.4638 0.4802 0.5013
#> a        0.4169 0.0260 20000 0.3300 0.3477 0.3643 0.4668 0.4811 0.4993
#> Y~~Y     0.7138 0.0241 20000 0.6335 0.6500 0.6653 0.7600 0.7741 0.7903
#> M~~M     0.8262 0.0217 20000 0.7507 0.7685 0.7821 0.8673 0.8791 0.8911
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1716 0.0159 20000 0.1218 0.1323 0.1412 0.2034 0.2139 0.2258
#> direct   0.2109 0.0290 20000 0.1166 0.1355 0.1533 0.2679 0.2871 0.3114
#> total    0.3825 0.0271 20000 0.2877 0.3108 0.3284 0.4344 0.4510 0.4720
```

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