semmcci: Monte Carlo Confidence Intervals

Ivan Jacob Agaloos Pesigan

Installation

You can install the CRAN release of `semmcci` with:

```r
install.packages("semmcci")
```

You can install the development version of `semmcci` from GitHub with:

```r
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 = 0.05)

#> Monte Carlo Confidence Intervals
#>             est     se     R   2.5%  97.5%
#> cp       0.2979 0.0358 20000 0.2282 0.3682
#> b        0.4551 0.0319 20000 0.3928 0.5167
#> a        0.5032 0.0310 20000 0.4425 0.5643
#> Y~~Y     1.0175 0.0460 20000 0.9274 1.1075
#> M~~M     0.9818 0.0440 20000 0.8970 1.0682
```

```
#> X~~X     1.0082 0.0000 20000 1.0082 1.0082
#> indirect 0.2290 0.0215 20000 0.1886 0.2726
#> direct   0.2979 0.0358 20000 0.2282 0.3682
#> total    0.5269 0.0349 20000 0.4580 0.5950
```

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 = 0.05)
vcov(unstd)

#>                     cp             b             a          Y~~Y         M~~M
#> cp        1.262268e-03 -5.216038e-04 -1.116305e-06 -1.059054e-05 8.265170e-06
#> b        -5.216038e-04  1.031825e-03  6.225931e-06 -2.569003e-06 4.165162e-06
#> a        -1.116305e-06  6.225931e-06  9.874212e-04 -1.808795e-05 2.030282e-06
#> Y~~Y     -1.059054e-05 -2.569003e-06 -1.808795e-05  2.049632e-03 5.778843e-06
#> M~~M      8.265170e-06  4.165162e-06  2.030282e-06  5.778843e-06 1.917763e-03
#> X~~X      5.140403e-06  2.521090e-06  2.285907e-06  7.244459e-06 8.107378e-06
#> indirect -2.629486e-04  5.223456e-04  4.525191e-04 -9.121039e-06 2.739687e-06
#> direct    1.262268e-03 -5.216038e-04 -1.116305e-06 -1.059054e-05 8.265170e-06
#> total     9.993198e-04  7.418122e-07  4.514027e-04 -1.971158e-05 1.100486e-05
#>                  X~~X      indirect        direct         total
#> cp       5.140403e-06 -2.629486e-04  1.262268e-03  9.993198e-04
```

```
#> b        2.521090e-06  5.223456e-04 -5.216038e-04  7.418122e-07
#> a        2.285907e-06  4.525191e-04 -1.116305e-06  4.514027e-04
#> Y~~Y     7.244459e-06 -9.121039e-06 -1.059054e-05 -1.971158e-05
#> M~~M     8.107378e-06  2.739687e-06  8.265170e-06  1.100486e-05
#> X~~X     2.027925e-03  2.400183e-06  5.140403e-06  7.540586e-06
#> indirect 2.400183e-06  4.699470e-04 -2.629486e-04  2.069984e-04
#> direct   5.140403e-06 -2.629486e-04  1.262268e-03  9.993198e-04
#> total    7.540586e-06  2.069984e-04  9.993198e-04  1.206318e-03
```

```
MCStd(unstd)
```

```
#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2441 0.0286 20000 0.1459 0.1702 0.1878 0.2999 0.3198 0.3378
#> b        0.4132 0.0273 20000 0.3234 0.3413 0.3584 0.4653 0.4815 0.5006
#> a        0.4542 0.0252 20000 0.3663 0.3865 0.4041 0.5030 0.5176 0.5331
#> Y~~Y     0.6780 0.0244 20000 0.5987 0.6141 0.6294 0.7250 0.7387 0.7559
#> M~~M     0.7937 0.0229 20000 0.7158 0.7321 0.7470 0.8367 0.8506 0.8659
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1877 0.0166 20000 0.1367 0.1463 0.1560 0.2207 0.2310 0.2449
#> direct   0.2441 0.0286 20000 0.1459 0.1702 0.1878 0.2999 0.3198 0.3378
#> total    0.4318 0.0258 20000 0.3443 0.3630 0.3804 0.4809 0.4975 0.5137
```

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*, *56*(3), 1678–1696. 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. (2024). *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