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       0.1842 0.0348 20000 0.0722 0.0950 0.1156 0.2518 0.2747 0.2962
#> b        0.4871 0.0295 20000 0.3881 0.4118 0.4287 0.5445 0.5607 0.5803
#> a        0.5004 0.0338 20000 0.3904 0.4140 0.4337 0.5660 0.5897 0.6115
#> Y~~Y     0.9431 0.0423 20000 0.8039 0.8326 0.8601 1.0261 1.0556 1.0884
#> M~~M     1.0816 0.0482 20000 0.9208 0.9584 0.9868 1.1759 1.2061 1.2486
#> indirect 0.2437 0.0221 20000 0.1733 0.1900 0.2017 0.2885 0.3043 0.3228
#> direct   0.1842 0.0348 20000 0.0722 0.0950 0.1156 0.2518 0.2747 0.2962
```

```
#> total    0.4279 0.0356 20000 0.3164 0.3376 0.3583 0.4976 0.5197 0.5448
```

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         1.201901e-03 -4.269227e-04  9.726280e-06 9.693810e-07 -7.205429e-08
#> b         -4.269227e-04  8.621611e-04 -5.628447e-06 1.675493e-06  3.214888e-06
#> a          9.726280e-06 -5.628447e-06  1.105607e-03 1.718632e-05 -4.180848e-06
#> Y~~Y       9.693810e-07  1.675493e-06  1.718632e-05 1.789577e-03  7.530713e-07
#> M~~M      -7.205429e-08  3.214888e-06 -4.180848e-06 7.530713e-07  2.336722e-03
#> X~~X      -4.417043e-06  2.413896e-07  1.753522e-05 1.802068e-05  7.876700e-07
#> indirect -2.087449e-04  4.279064e-04  5.359708e-04 9.231590e-06 -2.325121e-07
#> direct     1.201901e-03 -4.269227e-04  9.726280e-06 9.693810e-07 -7.205429e-08
#> total      9.931563e-04  9.836506e-07  5.456971e-04 1.020097e-05 -3.045664e-07
#>                    X~~X      indirect        direct         total
#> cp        -4.417043e-06 -2.087449e-04  1.201901e-03  9.931563e-04
#> b          2.413896e-07  4.279064e-04 -4.269227e-04  9.836506e-07
#> a          1.753522e-05  5.359708e-04  9.726280e-06  5.456971e-04
#> Y~~Y       1.802068e-05  9.231590e-06  9.693810e-07  1.020097e-05
```

```
#> M~~M      7.876700e-07 -2.325121e-07 -7.205429e-08 -3.045664e-07
#> X~~X      1.829748e-03  8.738122e-06 -4.417043e-06  4.321079e-06
#> indirect  8.738122e-06  4.758983e-04 -2.087449e-04  2.671535e-04
#> direct   -4.417043e-06 -2.087449e-04  1.201901e-03  9.931563e-04
#> total     4.321079e-06  2.671535e-04  9.931563e-04  1.260310e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.1537 0.0287 20000 0.0587 0.0794 0.0960 0.2088 0.2254 0.2450
#> b         0.4775 0.0260 20000 0.3849 0.4090 0.4262 0.5276 0.5425 0.5614
#> a         0.4260 0.0258 20000 0.3368 0.3569 0.3748 0.4751 0.4902 0.5067
#> Y~~Y      0.6859 0.0242 20000 0.6034 0.6215 0.6376 0.7322 0.7472 0.7627
#> M~~M      0.8186 0.0219 20000 0.7433 0.7597 0.7743 0.8595 0.8726 0.8866
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2034 0.0169 20000 0.1482 0.1606 0.1705 0.2365 0.2474 0.2602
#> direct    0.1537 0.0287 20000 0.0587 0.0794 0.0960 0.2088 0.2254 0.2450
#> total     0.3571 0.0276 20000 0.2658 0.2849 0.3013 0.4098 0.4252 0.4435
```

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