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       0.2009 0.0370 20000 0.0787 0.1033 0.1288 0.2745 0.2962 0.3228
#> b        0.4490 0.0329 20000 0.3436 0.3646 0.3838 0.5130 0.5318 0.5569
#> a        0.5115 0.0320 20000 0.4057 0.4293 0.4487 0.5742 0.5931 0.6208
#> Y~~Y     1.0922 0.0493 20000 0.9392 0.9644 0.9945 1.1903 1.2213 1.2554
#> M~~M     1.0202 0.0454 20000 0.8737 0.9036 0.9321 1.1098 1.1371 1.1687
#> indirect 0.2297 0.0221 20000 0.1646 0.1750 0.1878 0.2747 0.2890 0.3072
#> direct   0.2009 0.0370 20000 0.0787 0.1033 0.1288 0.2745 0.2962 0.3228
```

```
#> total    0.4306 0.0360 20000 0.3088 0.3379 0.3605 0.5017 0.5240 0.5477
```

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        1.367363e-03 -5.556033e-04  7.610120e-06 8.451578e-09  4.163883e-06
#> b        -5.556033e-04  1.071442e-03 -2.838399e-06 1.789075e-05  3.605661e-06
#> a         7.610120e-06 -2.838399e-06  1.013103e-03 1.835957e-05 -1.158511e-05
#> Y~~Y      8.451578e-09  1.789075e-05  1.835957e-05 2.388033e-03  7.307806e-06
#> M~~M      4.163883e-06  3.605661e-06 -1.158511e-05 7.307806e-06  2.071762e-03
#> X~~X     -1.050948e-05  2.623606e-06 -3.555268e-06 4.337287e-07  7.047652e-06
#> indirect -2.809397e-04  5.465928e-04  4.538996e-04 1.751790e-05 -3.417055e-06
#> direct    1.367363e-03 -5.556033e-04  7.610120e-06 8.451578e-09  4.163883e-06
#> total     1.086423e-03 -9.010559e-06  4.615097e-04 1.752635e-05  7.468280e-07
#>                    X~~X      indirect        direct         total
#> cp       -1.050948e-05 -2.809397e-04  1.367363e-03  1.086423e-03
#> b         2.623606e-06  5.465928e-04 -5.556033e-04 -9.010559e-06
#> a        -3.555268e-06  4.538996e-04  7.610120e-06  4.615097e-04
#> Y~~Y      4.337287e-07  1.751790e-05  8.451578e-09  1.752635e-05
```

```
#> M~~M      7.047652e-06 -3.417055e-06  4.163883e-06  7.468280e-07
#> X~~X      2.003845e-03  3.373022e-07 -1.050948e-05 -1.017218e-05
#> indirect  3.373022e-07  4.845759e-04 -2.809397e-04  2.036361e-04
#> direct   -1.050948e-05 -2.809397e-04  1.367363e-03  1.086423e-03
#> total    -1.017218e-05  2.036361e-04  1.086423e-03  1.290059e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.1658 0.0302 20000 0.0646 0.0879 0.1059 0.2238 0.2423 0.2651
#> b         0.4176 0.0283 20000 0.3268 0.3433 0.3608 0.4719 0.4880 0.5071
#> a         0.4538 0.0251 20000 0.3699 0.3885 0.4038 0.5029 0.5181 0.5361
#> Y~~Y      0.7353 0.0237 20000 0.6527 0.6726 0.6873 0.7799 0.7940 0.8073
#> M~~M      0.7940 0.0228 20000 0.7126 0.7315 0.7471 0.8370 0.8490 0.8632
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1895 0.0170 20000 0.1399 0.1474 0.1569 0.2241 0.2341 0.2485
#> direct    0.1658 0.0302 20000 0.0646 0.0879 0.1059 0.2238 0.2423 0.2651
#> total     0.3553 0.0276 20000 0.2640 0.2828 0.2996 0.4077 0.4244 0.4473
```

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