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       0.2211 0.0360 20000 0.1017 0.1273 0.1491 0.2906 0.3123 0.3317
#> b        0.5061 0.0312 20000 0.4065 0.4266 0.4445 0.5671 0.5850 0.6082
#> a        0.5059 0.0324 20000 0.3998 0.4230 0.4427 0.5693 0.5895 0.6139
#> Y~~Y     1.0499 0.0466 20000 0.9005 0.9328 0.9591 1.1426 1.1716 1.1951
#> M~~M     1.0734 0.0483 20000 0.9180 0.9496 0.9782 1.1679 1.1974 1.2334
#> indirect 0.2560 0.0227 20000 0.1880 0.2005 0.2127 0.3018 0.3164 0.3353
#> direct   0.2211 0.0360 20000 0.1017 0.1273 0.1491 0.2906 0.3123 0.3317
```

```
#> total    0.4771 0.0361 20000 0.3631 0.3833 0.4053 0.5469 0.5684 0.5939
```

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        1.274676e-03 -4.910299e-04  7.220688e-06  2.295727e-07  1.644746e-05
#> b        -4.910299e-04  9.653467e-04  2.981119e-06 -7.729841e-06 -9.684831e-06
#> a         7.220688e-06  2.981119e-06  1.054041e-03 -1.513652e-06 -6.698961e-07
#> Y~~Y      2.295727e-07 -7.729841e-06 -1.513652e-06  2.229978e-03  2.826300e-05
#> M~~M      1.644746e-05 -9.684831e-06 -6.698961e-07  2.826300e-05  2.324120e-03
#> X~~X      1.814411e-05  4.561913e-06  5.231554e-06  2.969471e-05 -8.980702e-06
#> indirect -2.445934e-04  4.898889e-04  5.353537e-04 -3.969030e-06 -4.502918e-06
#> direct    1.274676e-03 -4.910299e-04  7.220688e-06  2.295727e-07  1.644746e-05
#> total     1.030083e-03 -1.140988e-06  5.425744e-04 -3.739457e-06  1.194454e-05
#>                    X~~X      indirect        direct         total
#> cp        1.814411e-05 -2.445934e-04  1.274676e-03  1.030083e-03
#> b         4.561913e-06  4.898889e-04 -4.910299e-04 -1.140988e-06
#> a         5.231554e-06  5.353537e-04  7.220688e-06  5.425744e-04
#> Y~~Y      2.969471e-05 -3.969030e-06  2.295727e-07 -3.739457e-06
```

```
#> M~~M     -8.980702e-06 -4.502918e-06  1.644746e-05  1.194454e-05
#> X~~X      2.072063e-03  5.936795e-06  1.814411e-05  2.408090e-05
#> indirect  5.936795e-06  5.199982e-04 -2.445934e-04  2.754048e-04
#> direct    1.814411e-05 -2.445934e-04  1.274676e-03  1.030083e-03
#> total     2.408090e-05  2.754048e-04  1.030083e-03  1.305488e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.1786 0.0286 20000 0.0890 0.1043 0.1223 0.2345 0.2517 0.2731
#> b         0.4685 0.0262 20000 0.3824 0.3999 0.4170 0.5192 0.5341 0.5494
#> a         0.4416 0.0255 20000 0.3546 0.3740 0.3903 0.4903 0.5045 0.5217
#> Y~~Y      0.6747 0.0243 20000 0.5914 0.6102 0.6253 0.7206 0.7358 0.7518
#> M~~M      0.8050 0.0225 20000 0.7278 0.7455 0.7596 0.8477 0.8601 0.8743
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2069 0.0170 20000 0.1561 0.1646 0.1741 0.2406 0.2525 0.2635
#> direct    0.1786 0.0286 20000 0.0890 0.1043 0.1223 0.2345 0.2517 0.2731
#> total     0.3855 0.0270 20000 0.2973 0.3122 0.3313 0.4366 0.4527 0.4713
```

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