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       0.3180 0.0344 20000 0.2016 0.2287 0.2514 0.3853 0.4062 0.4317
#> b        0.4900 0.0306 20000 0.3931 0.4123 0.4304 0.5496 0.5675 0.5895
#> a        0.5449 0.0314 20000 0.4436 0.4636 0.4833 0.6064 0.6267 0.6479
#> Y~~Y     0.9314 0.0416 20000 0.8002 0.8247 0.8494 1.0121 1.0370 1.0673
#> M~~M     1.0080 0.0449 20000 0.8696 0.8947 0.9194 1.0960 1.1215 1.1550
#> indirect 0.2670 0.0227 20000 0.1994 0.2114 0.2243 0.3127 0.3272 0.3437
#> direct   0.3180 0.0344 20000 0.2016 0.2287 0.2514 0.3853 0.4062 0.4317
```

```
#> total    0.5849 0.0338 20000 0.4725 0.4972 0.5183 0.6513 0.6701 0.6940
```

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         1.185902e-03 -4.883727e-04  9.130200e-06  1.100685e-06 -2.500174e-05
#> b         -4.883727e-04  9.198831e-04 -2.607586e-06  7.210834e-06  2.305123e-05
#> a          9.130200e-06 -2.607586e-06  9.853857e-04  1.006566e-05 -1.966465e-05
#> Y~~Y       1.100685e-06  7.210834e-06  1.006566e-05  1.735754e-03  3.072704e-06
#> M~~M      -2.500174e-05  2.305123e-05 -1.966465e-05  3.072704e-06  2.006063e-03
#> X~~X      -1.613580e-05  9.669462e-06 -6.053670e-06 -7.113514e-06 -1.024505e-05
#> indirect -2.618288e-04  5.003836e-04  4.814110e-04  8.775120e-06  3.128114e-06
#> direct     1.185902e-03 -4.883727e-04  9.130200e-06  1.100685e-06 -2.500174e-05
#> total      9.240734e-04  1.201084e-05  4.905412e-04  9.875805e-06 -2.187363e-05
#>                    X~~X      indirect        direct         total
#> cp        -1.613580e-05 -2.618288e-04  1.185902e-03  9.240734e-04
#> b          9.669462e-06  5.003836e-04 -4.883727e-04  1.201084e-05
#> a         -6.053670e-06  4.814110e-04  9.130200e-06  4.905412e-04
#> Y~~Y      -7.113514e-06  8.775120e-06  1.100685e-06  9.875805e-06
```

```
#> M~~M     -1.024505e-05  3.128114e-06 -2.500174e-05 -2.187363e-05
#> X~~X      2.096836e-03  2.485766e-06 -1.613580e-05 -1.365003e-05
#> indirect  2.485766e-06  5.096667e-04 -2.618288e-04  2.478379e-04
#> direct   -1.613580e-05 -2.618288e-04  1.185902e-03  9.240734e-04
#> total    -1.365003e-05  2.478379e-04  9.240734e-04  1.171911e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2601 0.0275 20000 0.1685 0.1884 0.2061 0.3137 0.3317 0.3537
#> b         0.4546 0.0260 20000 0.3679 0.3855 0.4032 0.5050 0.5201 0.5390
#> a         0.4804 0.0244 20000 0.4003 0.4157 0.4314 0.5272 0.5405 0.5544
#> Y~~Y      0.6120 0.0241 20000 0.5320 0.5492 0.5649 0.6593 0.6726 0.6909
#> M~~M      0.7692 0.0234 20000 0.6926 0.7078 0.7221 0.8139 0.8272 0.8398
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2184 0.0169 20000 0.1682 0.1773 0.1857 0.2519 0.2625 0.2752
#> direct    0.2601 0.0275 20000 0.1685 0.1884 0.2061 0.3137 0.3317 0.3537
#> total     0.4785 0.0244 20000 0.3963 0.4127 0.4290 0.5247 0.5394 0.5557
```

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