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       0.2521 0.0365 20000 0.1361 0.1561 0.1797 0.3236 0.3468 0.3704
#> b        0.4749 0.0333 20000 0.3687 0.3896 0.4097 0.5408 0.5604 0.5861
#> a        0.4926 0.0304 20000 0.3850 0.4140 0.4333 0.5520 0.5713 0.5940
#> Y~~Y     1.0137 0.0452 20000 0.8597 0.8973 0.9253 1.1018 1.1289 1.1662
#> M~~M     0.9042 0.0405 20000 0.7699 0.7995 0.8244 0.9837 1.0084 1.0372
#> indirect 0.2339 0.0219 20000 0.1652 0.1804 0.1924 0.2781 0.2934 0.3124
#> direct   0.2521 0.0365 20000 0.1361 0.1561 0.1797 0.3236 0.3468 0.3704
```

```
#> total    0.4860 0.0358 20000 0.3722 0.3939 0.4157 0.5561 0.5783 0.6040
```

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        1.319015e-03 -5.467230e-04 -1.788766e-06  1.483090e-05  9.306492e-06
#> b        -5.467230e-04  1.123073e-03  8.291350e-06 -1.146019e-05 -6.827371e-06
#> a        -1.788766e-06  8.291350e-06  9.340422e-04 -5.924467e-07  6.599526e-07
#> Y~~Y      1.483090e-05 -1.146019e-05 -5.924467e-07  2.037307e-03 -1.896644e-05
#> M~~M      9.306492e-06 -6.827371e-06  6.599526e-07 -1.896644e-05  1.648975e-03
#> X~~X     -2.592720e-06 -8.874841e-06 -1.451977e-05 -6.207313e-06  4.519854e-06
#> indirect -2.701212e-04  5.573479e-04  4.477744e-04 -6.224271e-06 -3.109518e-06
#> direct    1.319015e-03 -5.467230e-04 -1.788766e-06  1.483090e-05  9.306492e-06
#> total     1.048893e-03  1.062481e-05  4.459857e-04  8.606632e-06  6.196974e-06
#>                    X~~X      indirect        direct         total
#> cp       -2.592720e-06 -2.701212e-04  1.319015e-03  1.048893e-03
#> b        -8.874841e-06  5.573479e-04 -5.467230e-04  1.062481e-05
#> a        -1.451977e-05  4.477744e-04 -1.788766e-06  4.459857e-04
#> Y~~Y     -6.207313e-06 -6.224271e-06  1.483090e-05  8.606632e-06
```

```
#> M~~M      4.519854e-06 -3.109518e-06  9.306492e-06  6.196974e-06
#> X~~X      1.826251e-03 -1.114833e-05 -2.592720e-06 -1.374105e-05
#> indirect -1.114833e-05  4.884114e-04 -2.701212e-04  2.182902e-04
#> direct   -2.592720e-06 -2.701212e-04  1.319015e-03  1.048893e-03
#> total    -1.374105e-05  2.182902e-04  1.048893e-03  1.267183e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2054 0.0291 20000 0.1098 0.1285 0.1477 0.2618 0.2782 0.2989
#> b         0.4213 0.0277 20000 0.3304 0.3500 0.3667 0.4743 0.4900 0.5056
#> a         0.4523 0.0250 20000 0.3675 0.3864 0.4027 0.5001 0.5155 0.5320
#> Y~~Y      0.7020 0.0242 20000 0.6212 0.6369 0.6530 0.7483 0.7622 0.7775
#> M~~M      0.7954 0.0226 20000 0.7170 0.7343 0.7499 0.8378 0.8507 0.8649
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1906 0.0167 20000 0.1394 0.1492 0.1585 0.2239 0.2348 0.2482
#> direct   0.2054 0.0291 20000 0.1098 0.1285 0.1477 0.2618 0.2782 0.2989
#> total    0.3959 0.0265 20000 0.3049 0.3249 0.3429 0.4465 0.4611 0.4815
```

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