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       0.2486 0.0355 20000 0.1342 0.1588 0.1794 0.3182 0.3419 0.3678
#> b        0.4923 0.0315 20000 0.3940 0.4108 0.4300 0.5537 0.5728 0.5948
#> a        0.4829 0.0320 20000 0.3792 0.4006 0.4205 0.5461 0.5655 0.5891
#> Y~~Y     1.0300 0.0463 20000 0.8676 0.9086 0.9397 1.1202 1.1480 1.1786
#> M~~M     1.0453 0.0468 20000 0.8876 0.9256 0.9537 1.1374 1.1682 1.1975
#> indirect 0.2377 0.0220 20000 0.1726 0.1837 0.1956 0.2821 0.2962 0.3134
#> direct   0.2486 0.0355 20000 0.1342 0.1588 0.1794 0.3182 0.3419 0.3678
```

```
#> total    0.4863 0.0357 20000 0.3646 0.3935 0.4171 0.5563 0.5803 0.6080
```

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        1.239065e-03 -4.673775e-04 -1.617551e-05 -6.923801e-06 -1.457474e-05
#> b        -4.673775e-04  9.753463e-04  2.231078e-05  4.405501e-06  1.862304e-05
#> a        -1.617551e-05  2.231078e-05  1.040852e-03 -1.277064e-05  5.230698e-06
#> Y~~Y     -6.923801e-06  4.405501e-06 -1.277064e-05  2.073652e-03 -2.037696e-05
#> M~~M     -1.457474e-05  1.862304e-05  5.230698e-06 -2.037696e-05  2.178554e-03
#> X~~X     -1.737344e-06 -3.525531e-07 -2.068483e-05 -5.217676e-06  2.141152e-05
#> indirect -2.335173e-04  4.821035e-04  5.226422e-04 -4.488176e-06  1.176954e-05
#> direct    1.239065e-03 -4.673775e-04 -1.617551e-05 -6.923801e-06 -1.457474e-05
#> total     1.005547e-03  1.472595e-05  5.064667e-04 -1.141198e-05 -2.805201e-06
#>                    X~~X      indirect        direct         total
#> cp       -1.737344e-06 -2.335173e-04  1.239065e-03  1.005547e-03
#> b        -3.525531e-07  4.821035e-04 -4.673775e-04  1.472595e-05
#> a        -2.068483e-05  5.226422e-04 -1.617551e-05  5.064667e-04
#> Y~~Y     -5.217676e-06 -4.488176e-06 -6.923801e-06 -1.141198e-05
```

```
#> M~~M      2.141152e-05  1.176954e-05 -1.457474e-05 -2.805201e-06
#> X~~X      2.028125e-03 -1.015144e-05 -1.737344e-06 -1.188878e-05
#> indirect -1.015144e-05  4.908954e-04 -2.335173e-04  2.573781e-04
#> direct   -1.737344e-06 -2.335173e-04  1.239065e-03  1.005547e-03
#> total    -1.188878e-05  2.573781e-04  1.005547e-03  1.262925e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2027 0.0283 20000 0.1101 0.1289 0.1472 0.2581 0.2751 0.2960
#> b         0.4516 0.0264 20000 0.3600 0.3820 0.3978 0.5020 0.5174 0.5327
#> a         0.4293 0.0257 20000 0.3444 0.3602 0.3777 0.4787 0.4929 0.5107
#> Y~~Y      0.6764 0.0244 20000 0.5959 0.6120 0.6275 0.7230 0.7383 0.7565
#> M~~M      0.8157 0.0220 20000 0.7392 0.7571 0.7709 0.8574 0.8703 0.8814
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1939 0.0167 20000 0.1420 0.1519 0.1615 0.2265 0.2373 0.2486
#> direct   0.2027 0.0283 20000 0.1101 0.1289 0.1472 0.2581 0.2751 0.2960
#> total    0.3966 0.0265 20000 0.3076 0.3274 0.3435 0.4477 0.4615 0.4813
```

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