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       0.1828 0.0359 20000 0.0647 0.0911 0.1123 0.2531 0.2759 0.3033
#> b        0.5612 0.0296 20000 0.4678 0.4854 0.5036 0.6188 0.6368 0.6589
#> a        0.5774 0.0339 20000 0.4606 0.4897 0.5117 0.6437 0.6658 0.6872
#> Y~~Y     0.9206 0.0411 20000 0.7837 0.8127 0.8395 1.0020 1.0279 1.0537
#> M~~M     1.0559 0.0471 20000 0.8995 0.9356 0.9640 1.1481 1.1763 1.2104
#> indirect 0.3241 0.0257 20000 0.2468 0.2611 0.2756 0.3758 0.3929 0.4158
#> direct   0.1828 0.0359 20000 0.0647 0.0911 0.1123 0.2531 0.2759 0.3033
```

```
#> total    0.5069 0.0369 20000 0.3887 0.4127 0.4348 0.5803 0.6040 0.6258
```

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        1.290407e-03 -5.019136e-04 -9.201936e-06  1.071185e-05 -1.688887e-05
#> b        -5.019136e-04  8.639965e-04  5.896644e-06 -9.585003e-06  1.956851e-06
#> a        -9.201936e-06  5.896644e-06  1.134582e-03 -8.303259e-06 -1.449211e-05
#> Y~~Y      1.071185e-05 -9.585003e-06 -8.303259e-06  1.686406e-03  1.751157e-05
#> M~~M     -1.688887e-05  1.956851e-06 -1.449211e-05  1.751157e-05  2.210445e-03
#> X~~X      1.118711e-05 -4.263693e-06  1.344503e-05 -4.181401e-06  6.057402e-06
#> indirect -2.949320e-04  5.024850e-04  6.406449e-04 -1.031006e-05 -6.720787e-06
#> direct    1.290407e-03 -5.019136e-04 -9.201936e-06  1.071185e-05 -1.688887e-05
#> total     9.954746e-04  5.714006e-07  6.314429e-04  4.017943e-07 -2.360966e-05
#>                    X~~X      indirect        direct         total
#> cp        1.118711e-05 -2.949320e-04  1.290407e-03  9.954746e-04
#> b        -4.263693e-06  5.024850e-04 -5.019136e-04  5.714006e-07
#> a         1.344503e-05  6.406449e-04 -9.201936e-06  6.314429e-04
#> Y~~Y     -4.181401e-06 -1.031006e-05  1.071185e-05  4.017943e-07
```

```
#> M~~M      6.057402e-06 -6.720787e-06 -1.688887e-05 -2.360966e-05
#> X~~X      1.734811e-03  4.979151e-06  1.118711e-05  1.616627e-05
#> indirect  4.979151e-06  6.511106e-04 -2.949320e-04  3.561786e-04
#> direct    1.118711e-05 -2.949320e-04  1.290407e-03  9.954746e-04
#> total     1.616627e-05  3.561786e-04  9.954746e-04  1.351653e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.1441 0.0282 20000 0.0532 0.0714 0.0893 0.1993 0.2167 0.2361
#> b         0.5369 0.0250 20000 0.4565 0.4715 0.4872 0.5847 0.5990 0.6188
#> a         0.4758 0.0246 20000 0.3970 0.4115 0.4270 0.5229 0.5370 0.5537
#> Y~~Y      0.6173 0.0241 20000 0.5319 0.5542 0.5696 0.6637 0.6765 0.6911
#> M~~M      0.7737 0.0233 20000 0.6934 0.7116 0.7266 0.8177 0.8307 0.8424
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2554 0.0183 20000 0.1989 0.2096 0.2195 0.2919 0.3025 0.3180
#> direct    0.1441 0.0282 20000 0.0532 0.0714 0.0893 0.1993 0.2167 0.2361
#> total     0.3996 0.0266 20000 0.3089 0.3297 0.3467 0.4515 0.4663 0.4824
```

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