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:

```
if (!require("remotes")) 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. A description of the package and code examples are presented in Pesigan and Cheung (2023).

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       0.2300 0.0370 20000 0.1132 0.1344 0.1577 0.3040 0.3271 0.3530
#> b        0.5018 0.0331 20000 0.3947 0.4171 0.4373 0.5662 0.5885 0.6126
#> a        0.5766 0.0299 20000 0.4820 0.4984 0.5171 0.6350 0.6530 0.6740
#> Y~~Y     1.0230 0.0456 20000 0.8818 0.9086 0.9354 1.1143 1.1424 1.1796
#> M~~M     0.9195 0.0413 20000 0.7830 0.8143 0.8382 0.9992 1.0237 1.0562
```

```
#> X~~X     1.0283 0.0000 20000 1.0283 1.0283 1.0283 1.0283 1.0283 1.0283
#> indirect 0.2894 0.0243 20000 0.2138 0.2294 0.2431 0.3380 0.3553 0.3731
#> direct   0.2300 0.0370 20000 0.1132 0.1344 0.1577 0.3040 0.3271 0.3530
#> total    0.5194 0.0352 20000 0.3997 0.4284 0.4511 0.5888 0.6100 0.6337
```

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        1.368541e-03 -6.354869e-04  1.269295e-05 -7.712641e-06  5.577951e-06
#> b        -6.354869e-04  1.128475e-03 -5.444887e-07  1.084738e-05 -1.940790e-05
#> a         1.269295e-05 -5.444887e-07  8.979088e-04 -6.874994e-06  1.230908e-05
#> Y~~Y     -7.712641e-06  1.084738e-05 -6.874994e-06  2.112969e-03 -2.769883e-06
#> M~~M      5.577951e-06 -1.940790e-05  1.230908e-05 -2.769883e-06  1.719666e-03
#> X~~X     -8.034885e-06  1.320202e-06  1.239392e-05 -5.452529e-05  5.238062e-06
#> indirect -3.594856e-04  6.499550e-04  4.503427e-04  3.050987e-06 -4.866092e-06
#> direct    1.368541e-03 -6.354869e-04  1.269295e-05 -7.712641e-06  5.577951e-06
#> total     1.009056e-03  1.446808e-05  4.630356e-04 -4.661654e-06  7.118588e-07
#>                    X~~X      indirect        direct         total
#> cp       -8.034885e-06 -3.594856e-04  1.368541e-03  1.009056e-03
```

```
#> b          1.320202e-06  6.499550e-04 -6.354869e-04  1.446808e-05
#> a          1.239392e-05  4.503427e-04  1.269295e-05  4.630356e-04
#> Y~~Y      -5.452529e-05  3.050987e-06 -7.712641e-06 -4.661654e-06
#> M~~M       5.238062e-06 -4.866092e-06  5.577951e-06  7.118588e-07
#> X~~X       2.101493e-03  6.568853e-06 -8.034885e-06 -1.466032e-06
#> indirect   6.568853e-06  6.015529e-04 -3.594856e-04  2.420674e-04
#> direct    -8.034885e-06 -3.594856e-04  1.368541e-03  1.009056e-03
#> total     -1.466032e-06  2.420674e-04  1.009056e-03  1.251123e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.1885 0.0300 20000 0.0907 0.1107 0.1294 0.2465 0.2651 0.2853
#> b         0.4554 0.0280 20000 0.3604 0.3813 0.3997 0.5098 0.5253 0.5466
#> a         0.5206 0.0231 20000 0.4403 0.4596 0.4749 0.5644 0.5780 0.5909
#> Y~~Y      0.6678 0.0246 20000 0.5898 0.6025 0.6189 0.7161 0.7298 0.7440
#> M~~M      0.7290 0.0240 20000 0.6508 0.6659 0.6814 0.7745 0.7887 0.8061
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2371 0.0185 20000 0.1776 0.1897 0.2013 0.2738 0.2846 0.2977
#> direct    0.1885 0.0300 20000 0.0907 0.1107 0.1294 0.2465 0.2651 0.2853
#> total     0.4255 0.0262 20000 0.3379 0.3560 0.3723 0.4748 0.4907 0.5073
```

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

Pesigan, I. J. A., & Cheung, S. F. (2023). Monte Carlo confidence intervals for the indirect effect with missing data. *Behavior Research Methods*. https://doi.org/10.3758/s13428-023-02114-4

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. (2023). *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