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 = 0.05)

#> Monte Carlo Confidence Intervals
#>             est     se     R   2.5%  97.5%
#> cp       0.1966 0.0354 20000 0.1282 0.2657
#> b        0.5004 0.0298 20000 0.4427 0.5594
#> a        0.5024 0.0344 20000 0.4346 0.5693
#> Y~~Y     0.9802 0.0436 20000 0.8952 1.0656
#> M~~M     1.1037 0.0495 20000 1.0066 1.2019
```

```
#> X~~X     0.9483 0.0000 20000 0.9483 0.9483
#> indirect 0.2514 0.0228 20000 0.2079 0.2973
#> direct   0.1966 0.0354 20000 0.1282 0.2657
#> total    0.4480 0.0364 20000 0.3764 0.5199
```

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 = 0.05)
vcov(unstd)

#>                      cp             b             a          Y~~Y          M~~M
#> cp        1.262064e-03 -4.534674e-04  4.883476e-06 -1.421767e-05 -1.614755e-05
#> b        -4.534674e-04  8.896284e-04 -3.365735e-06 -1.676273e-05  1.441149e-05
#> a         4.883476e-06 -3.365735e-06  1.128167e-03  1.250150e-05 -6.398011e-06
#> Y~~Y     -1.421767e-05 -1.676273e-05  1.250150e-05  1.877585e-03 -2.831493e-05
#> M~~M     -1.614755e-05  1.441149e-05 -6.398011e-06 -2.831493e-05  2.446544e-03
#> X~~X      6.801422e-06 -7.946383e-06  2.998430e-06  6.498936e-06 -1.650297e-05
#> indirect -2.253588e-04  4.456670e-04  5.626991e-04 -2.182637e-06  4.013043e-06
#> direct    1.262064e-03 -4.534674e-04  4.883476e-06 -1.421767e-05 -1.614755e-05
#> total     1.036705e-03 -7.800380e-06  5.675826e-04 -1.640031e-05 -1.213451e-05
#>                    X~~X      indirect        direct         total
#> cp        6.801422e-06 -2.253588e-04  1.262064e-03  1.036705e-03
```

```
#> b        -7.946383e-06  4.456670e-04 -4.534674e-04 -7.800380e-06
#> a         2.998430e-06  5.626991e-04  4.883476e-06  5.675826e-04
#> Y~~Y      6.498936e-06 -2.182637e-06 -1.421767e-05 -1.640031e-05
#> M~~M     -1.650297e-05  4.013043e-06 -1.614755e-05 -1.213451e-05
#> X~~X      1.781688e-03 -2.558877e-06  6.801422e-06  4.242544e-06
#> indirect -2.558877e-06  5.066092e-04 -2.253588e-04  2.812504e-04
#> direct    6.801422e-06 -2.253588e-04  1.262064e-03  1.036705e-03
#> total     4.242544e-06  2.812504e-04  1.036705e-03  1.317956e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1592 0.0286 20000 0.0633 0.0844 0.1024 0.2148 0.2321 0.2555
#> b        0.4821 0.0261 20000 0.3946 0.4129 0.4295 0.5327 0.5476 0.5672
#> a        0.4221 0.0257 20000 0.3344 0.3546 0.3706 0.4715 0.4871 0.5073
#> Y~~Y     0.6774 0.0243 20000 0.5949 0.6132 0.6293 0.7242 0.7380 0.7535
#> M~~M     0.8218 0.0217 20000 0.7427 0.7628 0.7777 0.8627 0.8742 0.8882
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2035 0.0168 20000 0.1513 0.1613 0.1710 0.2368 0.2486 0.2599
#> direct   0.1592 0.0286 20000 0.0633 0.0844 0.1024 0.2148 0.2321 0.2555
#> total    0.3627 0.0275 20000 0.2644 0.2875 0.3075 0.4151 0.4303 0.4488
```

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*, *56*(3), 1678–1696. 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. (2024). *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