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        0.1852 0.0362 20000 0.1138 0.2564
#> b         0.5124 0.0332 20000 0.4471 0.5776
#> a         0.4564 0.0313 20000 0.3957 0.5177
#> Y~~Y      1.0947 0.0486 20000 0.9987 1.1893
#> M~~M      0.9770 0.0435 20000 0.8909 1.0622
```

```
#> X~~X     0.9991 0.0000 20000 0.9991 0.9991
#> indirect 0.2339 0.0222 20000 0.1922 0.2790
#> direct   0.1852 0.0362 20000 0.1138 0.2564
#> total    0.4191 0.0366 20000 0.3471 0.4903
```

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        1.318540e-03 -5.021682e-04  2.381084e-06 -5.985933e-06 1.589859e-06
#> b        -5.021682e-04  1.124038e-03  4.454665e-06  3.135361e-07 6.248108e-06
#> a         2.381084e-06  4.454665e-06  9.734419e-04 -1.979667e-05 5.819165e-06
#> Y~~Y     -5.985933e-06  3.135361e-07 -1.979667e-05  2.437676e-03 3.233185e-06
#> M~~M      1.589859e-06  6.248108e-06  5.819165e-06  3.233185e-06 1.909134e-03
#> X~~X      1.606620e-05 -1.650912e-05 -6.752608e-06  1.764834e-05 1.811406e-05
#> indirect -2.278638e-04  5.152778e-04  5.009029e-04 -9.875977e-06 6.046237e-06
#> direct    1.318540e-03 -5.021682e-04  2.381084e-06 -5.985933e-06 1.589859e-06
#> total     1.090676e-03  1.310966e-05  5.032840e-04 -1.586191e-05 7.636096e-06
#>                   X~~X      indirect        direct         total
#> cp        1.606620e-05 -2.278638e-04  1.318540e-03  1.090676e-03
```

```
#> b        -1.650912e-05  5.152778e-04 -5.021682e-04  1.310966e-05
#> a        -6.752608e-06  5.009029e-04  2.381084e-06  5.032840e-04
#> Y~~Y      1.764834e-05 -9.875977e-06 -5.985933e-06 -1.586191e-05
#> M~~M      1.811406e-05  6.046237e-06  1.589859e-06  7.636096e-06
#> X~~X      2.000230e-03 -1.100884e-05  1.606620e-05  5.057363e-06
#> indirect -1.100884e-05  4.929802e-04 -2.278638e-04  2.651163e-04
#> direct    1.606620e-05 -2.278638e-04  1.318540e-03  1.090676e-03
#> total     5.057363e-06  2.651163e-04  1.090676e-03  1.355793e-03
```

```r
MCStd(unstd)
```

```
#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1498 0.0292 20000 0.0536 0.0737 0.0928 0.2070 0.2250 0.2482
#> b        0.4515 0.0270 20000 0.3612 0.3799 0.3981 0.5041 0.5198 0.5386
#> a        0.4191 0.0259 20000 0.3317 0.3513 0.3676 0.4690 0.4848 0.5000
#> Y~~Y     0.7170 0.0243 20000 0.6348 0.6506 0.6676 0.7626 0.7775 0.7903
#> M~~M     0.8244 0.0217 20000 0.7500 0.7650 0.7800 0.8649 0.8766 0.8899
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1892 0.0167 20000 0.1364 0.1477 0.1572 0.2228 0.2336 0.2483
#> direct   0.1498 0.0292 20000 0.0536 0.0737 0.0928 0.2070 0.2250 0.2482
#> total    0.3390 0.0280 20000 0.2473 0.2667 0.2838 0.3931 0.4117 0.4328
```

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