semmcci: Monte Carlo Confidence Intervals

Ivan Jacob Agaloos Pesigan

Installation

You can install the CRAN release of `semmcci` with:

```r
install.packages("semmcci")
```

You can install the development version of `semmcci` from GitHub with:

```r
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        0.3235 0.0374 20000 0.2508 0.3963
#> b         0.3846 0.0327 20000 0.3200 0.4483
#> a         0.5118 0.0326 20000 0.4486 0.5768
#> Y~~Y      1.0351 0.0464 20000 0.9445 1.1261
#> M~~M      0.9683 0.0430 20000 0.8840 1.0531
```

```
#> X~~X     0.9340 0.0000 20000 0.9340 0.9340
#> indirect 0.1968 0.0211 20000 0.1574 0.2399
#> direct   0.3235 0.0374 20000 0.2508 0.3963
#> total    0.5204 0.0355 20000 0.4507 0.5899
```

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         1.390816e-03 -5.567506e-04  4.164149e-06 -2.115286e-06  1.986470e-05
#> b         -5.567506e-04  1.081295e-03  6.332381e-06 -4.851144e-06 -7.146828e-06
#> a          4.164149e-06  6.332381e-06  1.013326e-03 -1.483769e-05  1.055823e-05
#> Y~~Y      -2.115286e-06 -4.851144e-06 -1.483769e-05  2.144732e-03  1.340938e-05
#> M~~M       1.986470e-05 -7.146828e-06  1.055823e-05  1.340938e-05  1.864611e-03
#> X~~X       7.286538e-06 -3.740567e-06 -6.756040e-06 -5.778517e-06 -2.914840e-05
#> indirect -2.831418e-04  5.557648e-04  3.931286e-04 -8.287820e-06  4.188829e-07
#> direct     1.390816e-03 -5.567506e-04  4.164149e-06 -2.115286e-06  1.986470e-05
#> total      1.107675e-03 -9.857655e-07  3.972928e-04 -1.040311e-05  2.028358e-05
#>                    X~~X      indirect        direct         total
#> cp         7.286538e-06 -2.831418e-04  1.390816e-03  1.107675e-03
```

```
#> b        -3.740567e-06  5.557648e-04 -5.567506e-04 -9.857655e-07
#> a        -6.756040e-06  3.931286e-04  4.164149e-06  3.972928e-04
#> Y~~Y     -5.778517e-06 -8.287820e-06 -2.115286e-06 -1.040311e-05
#> M~~M     -2.914840e-05  4.188829e-07  1.986470e-05  2.028358e-05
#> X~~X      1.755086e-03 -4.612335e-06  7.286538e-06  2.674203e-06
#> indirect -4.612335e-06  4.367614e-04 -2.831418e-04  1.536196e-04
#> direct    7.286538e-06 -2.831418e-04  1.390816e-03  1.107675e-03
#> total     2.674203e-06  1.536196e-04  1.107675e-03  1.261294e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2614 0.0294 20000 0.1618 0.1856 0.2028 0.3180 0.3347 0.3548
#> b        0.3540 0.0288 20000 0.2593 0.2784 0.2967 0.4098 0.4262 0.4468
#> a        0.4491 0.0250 20000 0.3674 0.3844 0.3996 0.4973 0.5115 0.5310
#> Y~~Y     0.7232 0.0241 20000 0.6371 0.6585 0.6742 0.7688 0.7822 0.7970
#> M~~M     0.7983 0.0224 20000 0.7180 0.7384 0.7527 0.8403 0.8522 0.8650
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1590 0.0160 20000 0.1100 0.1191 0.1284 0.1911 0.2015 0.2151
#> direct   0.2614 0.0294 20000 0.1618 0.1856 0.2028 0.3180 0.3347 0.3548
#> total    0.4204 0.0261 20000 0.3341 0.3511 0.3680 0.4697 0.4854 0.5020
```

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