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
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.

```r
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`.

```r
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.

```r
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       0.2099 0.0362 20000 0.0924 0.1158 0.1393 0.2814 0.3032 0.3357
#> b        0.4985 0.0333 20000 0.3884 0.4119 0.4328 0.5637 0.5839 0.6051
#> a        0.4553 0.0304 20000 0.3594 0.3779 0.3951 0.5147 0.5336 0.5539
#> Y~~Y     1.0366 0.0461 20000 0.8877 0.9176 0.9456 1.1263 1.1569 1.1929
#> M~~M     0.9205 0.0411 20000 0.7782 0.8158 0.8400 1.0017 1.0274 1.0547
#> indirect 0.2270 0.0214 20000 0.1613 0.1738 0.1862 0.2700 0.2829 0.3031
#> direct   0.2099 0.0362 20000 0.0924 0.1158 0.1393 0.2814 0.3032 0.3357
```

```
#> total    0.4369 0.0360 20000 0.3218 0.3430 0.3659 0.5079 0.5317 0.5583
```

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        1.277585e-03 -5.092131e-04  5.232929e-06 -2.973406e-06 4.651350e-06
#> b        -5.092131e-04  1.127454e-03  2.415179e-06  3.666688e-06 4.650087e-06
#> a         5.232929e-06  2.415179e-06  9.212307e-04  1.191007e-05 6.567312e-06
#> Y~~Y     -2.973406e-06  3.666688e-06  1.191007e-05  2.140650e-03 1.102296e-05
#> M~~M      4.651350e-06  4.650087e-06  6.567312e-06  1.102296e-05 1.671072e-03
#> X~~X      1.721305e-05 -4.069073e-06 -3.580027e-06  5.079307e-06 9.788150e-06
#> indirect -2.294499e-04  5.146601e-04  4.593136e-04  7.751633e-06 5.388449e-06
#> direct    1.277585e-03 -5.092131e-04  5.232929e-06 -2.973406e-06 4.651350e-06
#> total     1.048136e-03  5.446997e-06  4.645465e-04  4.778227e-06 1.003980e-05
#>                   X~~X      indirect        direct        total
#> cp        1.721305e-05 -2.294499e-04  1.277585e-03 1.048136e-03
#> b        -4.069073e-06  5.146601e-04 -5.092131e-04 5.446997e-06
#> a        -3.580027e-06  4.593136e-04  5.232929e-06 4.645465e-04
#> Y~~Y      5.079307e-06  7.751633e-06 -2.973406e-06 4.778227e-06
```

```
#> M~~M      9.788150e-06  5.388449e-06  4.651350e-06 1.003980e-05
#> X~~X      1.954233e-03 -3.387192e-06  1.721305e-05 1.382585e-05
#> indirect -3.387192e-06  4.639064e-04 -2.294499e-04 2.344565e-04
#> direct    1.721305e-05 -2.294499e-04  1.277585e-03 1.048136e-03
#> total     1.382585e-05  2.344565e-04  1.048136e-03 1.282592e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1728 0.0292 20000 0.0733 0.0977 0.1152 0.2299 0.2471 0.2695
#> b        0.4385 0.0272 20000 0.3520 0.3681 0.3840 0.4905 0.5080 0.5259
#> a        0.4259 0.0256 20000 0.3401 0.3582 0.3750 0.4759 0.4912 0.5067
#> Y~~Y     0.7134 0.0242 20000 0.6311 0.6487 0.6647 0.7597 0.7738 0.7899
#> M~~M     0.8186 0.0218 20000 0.7433 0.7587 0.7735 0.8594 0.8717 0.8843
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1868 0.0165 20000 0.1346 0.1454 0.1549 0.2195 0.2303 0.2452
#> direct   0.1728 0.0292 20000 0.0733 0.0977 0.1152 0.2299 0.2471 0.2695
#> total    0.3595 0.0275 20000 0.2597 0.2867 0.3053 0.4128 0.4296 0.4492
```

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