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:

```
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.

```
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       0.2808 0.0358 20000 0.1656 0.1893 0.2103 0.3513 0.3743 0.3982
#> b        0.4511 0.0324 20000 0.3465 0.3684 0.3877 0.5149 0.5350 0.5603
#> a        0.4648 0.0321 20000 0.3581 0.3813 0.4022 0.5273 0.5451 0.5674
#> Y~~Y     0.9657 0.0431 20000 0.8220 0.8571 0.8801 1.0503 1.0751 1.1052
#> M~~M     0.9294 0.0417 20000 0.7902 0.8187 0.8471 1.0107 1.0359 1.0579
#> indirect 0.2097 0.0208 20000 0.1462 0.1587 0.1705 0.2515 0.2662 0.2831
#> direct   0.2808 0.0358 20000 0.1656 0.1893 0.2103 0.3513 0.3743 0.3982
```

```
#> total    0.4904 0.0355 20000 0.3806 0.4007 0.4204 0.5600 0.5833 0.6086
```

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        1.280324e-03 -4.894659e-04  7.005191e-06 4.458855e-06 -2.307513e-06
#> b        -4.894659e-04  1.035452e-03 -6.492774e-06 9.237067e-06  2.201555e-05
#> a         7.005191e-06 -6.492774e-06  1.024427e-03 2.573513e-06  9.599587e-06
#> Y~~Y      4.458855e-06  9.237067e-06  2.573513e-06 1.861069e-03  4.577268e-06
#> M~~M     -2.307513e-06  2.201555e-05  9.599587e-06 4.577268e-06  1.753997e-03
#> X~~X      1.695991e-05 -3.754059e-06 -1.468657e-05 3.350966e-06 -2.189612e-06
#> indirect -2.245014e-04  4.780734e-04  4.581719e-04 5.665633e-06  1.483301e-05
#> direct    1.280324e-03 -4.894659e-04  7.005191e-06 4.458855e-06 -2.307513e-06
#> total     1.055823e-03 -1.139255e-05  4.651771e-04 1.012449e-05  1.252550e-05
#>                    X~~X      indirect        direct         total
#> cp        1.695991e-05 -2.245014e-04  1.280324e-03  1.055823e-03
#> b        -3.754059e-06  4.780734e-04 -4.894659e-04 -1.139255e-05
#> a        -1.468657e-05  4.581719e-04  7.005191e-06  4.651771e-04
#> Y~~Y      3.350966e-06  5.665633e-06  4.458855e-06  1.012449e-05
```

```
#> M~~M     -2.189612e-06  1.483301e-05 -2.307513e-06  1.252550e-05
#> X~~X      1.657720e-03 -8.837264e-06  1.695991e-05  8.122648e-06
#> indirect -8.837264e-06  4.293942e-04 -2.245014e-04  2.048928e-04
#> direct    1.695991e-05 -2.245014e-04  1.280324e-03  1.055823e-03
#> total     8.122648e-06  2.048928e-04  1.055823e-03  1.260715e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2291 0.0287 20000 0.1343 0.1556 0.1728 0.2853 0.3038 0.3223
#> b         0.4084 0.0273 20000 0.3112 0.3373 0.3538 0.4600 0.4773 0.4953
#> a         0.4188 0.0261 20000 0.3316 0.3519 0.3666 0.4687 0.4841 0.4996
#> Y~~Y      0.7024 0.0241 20000 0.6195 0.6375 0.6541 0.7490 0.7624 0.7778
#> M~~M      0.8246 0.0218 20000 0.7504 0.7656 0.7803 0.8656 0.8762 0.8900
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1711 0.0158 20000 0.1211 0.1320 0.1404 0.2023 0.2119 0.2252
#> direct   0.2291 0.0287 20000 0.1343 0.1556 0.1728 0.2853 0.3038 0.3223
#> total    0.4001 0.0266 20000 0.3135 0.3291 0.3467 0.4511 0.4677 0.4837
```

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