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       0.2839 0.0359 20000 0.1632 0.1904 0.2134 0.3539 0.3765 0.4065
#> b        0.4274 0.0333 20000 0.3174 0.3425 0.3619 0.4915 0.5126 0.5417
#> a        0.5207 0.0297 20000 0.4253 0.4448 0.4619 0.5791 0.5974 0.6131
#> Y~~Y     1.0368 0.0466 20000 0.8872 0.9156 0.9458 1.1286 1.1605 1.1973
#> M~~M     0.9320 0.0420 20000 0.7986 0.8253 0.8504 1.0142 1.0401 1.0766
#> indirect 0.2225 0.0216 20000 0.1567 0.1705 0.1820 0.2658 0.2796 0.3014
#> direct   0.2839 0.0359 20000 0.1632 0.1904 0.2134 0.3539 0.3765 0.4065
```

```
#> total    0.5065 0.0340 20000 0.3981 0.4184 0.4398 0.5726 0.5958 0.6204
```

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        1.264324e-03 -5.743968e-04  8.450549e-06  4.771145e-06  1.671122e-06
#> b        -5.743968e-04  1.116459e-03 -4.999059e-06 -5.488294e-06 -1.581165e-06
#> a         8.450549e-06 -4.999059e-06  8.881188e-04  7.253602e-06 -2.189288e-06
#> Y~~Y      4.771145e-06 -5.488294e-06  7.253602e-06  2.148333e-03 -8.453300e-06
#> M~~M      1.671122e-06 -1.581165e-06 -2.189288e-06 -8.453300e-06  1.733167e-03
#> X~~X      4.436152e-06 -5.257534e-06  1.131799e-05  1.497794e-05  8.985521e-06
#> indirect -2.950870e-04  5.789447e-04  3.778020e-04  7.643079e-08 -1.738015e-06
#> direct    1.264324e-03 -5.743968e-04  8.450549e-06  4.771145e-06  1.671122e-06
#> total     9.692372e-04  4.547892e-06  3.862525e-04  4.847575e-06 -6.689348e-08
#>                    X~~X      indirect        direct         total
#> cp        4.436152e-06 -2.950870e-04  1.264324e-03  9.692372e-04
#> b        -5.257534e-06  5.789447e-04 -5.743968e-04  4.547892e-06
#> a         1.131799e-05  3.778020e-04  8.450549e-06  3.862525e-04
#> Y~~Y      1.497794e-05  7.643079e-08  4.771145e-06  4.847575e-06
```

```
#> M~~M      8.985521e-06 -1.738015e-06  1.671122e-06 -6.689348e-08
#> X~~X      2.215292e-03  2.132212e-06  4.436152e-06  6.568364e-06
#> indirect  2.132212e-06  4.641386e-04 -2.950870e-04  1.690516e-04
#> direct    4.436152e-06 -2.950870e-04  1.264324e-03  9.692372e-04
#> total     6.568364e-06  1.690516e-04  9.692372e-04  1.138289e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2397 0.0295 20000 0.1439 0.1632 0.1826 0.2975 0.3144 0.3386
#> b         0.3880 0.0286 20000 0.2918 0.3120 0.3311 0.4425 0.4595 0.4787
#> a         0.4841 0.0243 20000 0.4058 0.4198 0.4348 0.5308 0.5444 0.5595
#> Y~~Y      0.7020 0.0242 20000 0.6216 0.6371 0.6521 0.7469 0.7627 0.7787
#> M~~M      0.7656 0.0235 20000 0.6870 0.7036 0.7182 0.8110 0.8238 0.8353
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1878 0.0170 20000 0.1350 0.1460 0.1552 0.2219 0.2334 0.2466
#> direct    0.2397 0.0295 20000 0.1439 0.1632 0.1826 0.2975 0.3144 0.3386
#> total     0.4275 0.0257 20000 0.3404 0.3608 0.3766 0.4774 0.4922 0.5084
```

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