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       0.2328 0.0346 20000 0.1241 0.1435 0.1651 0.3002 0.3221 0.3442
#> b        0.4976 0.0314 20000 0.3940 0.4172 0.4353 0.5593 0.5790 0.6026
#> a        0.4512 0.0320 20000 0.3461 0.3696 0.3887 0.5143 0.5330 0.5559
#> Y~~Y     0.9890 0.0437 20000 0.8451 0.8764 0.9038 1.0753 1.1018 1.1356
#> M~~M     1.0079 0.0452 20000 0.8608 0.8944 0.9196 1.0965 1.1265 1.1601
#> indirect 0.2245 0.0213 20000 0.1567 0.1718 0.1838 0.2678 0.2814 0.2974
#> direct   0.2328 0.0346 20000 0.1241 0.1435 0.1651 0.3002 0.3221 0.3442
```

```
#> total    0.4574 0.0354 20000 0.3437 0.3666 0.3877 0.5263 0.5481 0.5739
```

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         1.179006e-03 -4.250903e-04  3.044901e-06  1.546942e-05  1.248754e-05
#> b         -4.250903e-04  9.570226e-04  6.568770e-06  4.656337e-06 -2.169673e-05
#> a          3.044901e-06  6.568770e-06  1.029195e-03 -5.006434e-06  1.447601e-05
#> Y~~Y       1.546942e-05  4.656337e-06 -5.006434e-06  1.933169e-03 -7.015512e-06
#> M~~M       1.248754e-05 -2.169673e-05  1.447601e-05 -7.015512e-06  2.021175e-03
#> X~~X       5.973487e-06 -7.019130e-06 -1.426803e-05 -1.314930e-05  1.204911e-05
#> indirect -1.904827e-04  4.350323e-04  5.146820e-04 -1.874537e-07 -3.050474e-06
#> direct     1.179006e-03 -4.250903e-04  3.044901e-06  1.546942e-05  1.248754e-05
#> total      9.885229e-04  9.942017e-06  5.177269e-04  1.528196e-05  9.437065e-06
#>                    X~~X      indirect        direct         total
#> cp         5.973487e-06 -1.904827e-04  1.179006e-03  9.885229e-04
#> b         -7.019130e-06  4.350323e-04 -4.250903e-04  9.942017e-06
#> a         -1.426803e-05  5.146820e-04  3.044901e-06  5.177269e-04
#> Y~~Y      -1.314930e-05 -1.874537e-07  1.546942e-05  1.528196e-05
```

```
#> M~~M      1.204911e-05 -3.050474e-06  1.248754e-05  9.437065e-06
#> X~~X      1.983508e-03 -1.025454e-05  5.973487e-06 -4.281053e-06
#> indirect -1.025454e-05  4.531611e-04 -1.904827e-04  2.626785e-04
#> direct    5.973487e-06 -1.904827e-04  1.179006e-03  9.885229e-04
#> total    -4.281053e-06  2.626785e-04  9.885229e-04  1.251201e-03
```

```
MCStd(unstd)
```

```
#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1931 0.0281 20000 0.1055 0.1215 0.1383 0.2479 0.2670 0.2842
#> b        0.4552 0.0258 20000 0.3703 0.3891 0.4039 0.5049 0.5208 0.5367
#> a        0.4090 0.0264 20000 0.3198 0.3402 0.3556 0.4589 0.4738 0.4943
#> Y~~Y     0.6836 0.0241 20000 0.6070 0.6208 0.6354 0.7299 0.7441 0.7596
#> M~~M     0.8327 0.0215 20000 0.7557 0.7755 0.7894 0.8736 0.8843 0.8977
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1862 0.0164 20000 0.1350 0.1461 0.1544 0.2183 0.2289 0.2407
#> direct   0.1931 0.0281 20000 0.1055 0.1215 0.1383 0.2479 0.2670 0.2842
#> total    0.3792 0.0270 20000 0.2894 0.3090 0.3247 0.4317 0.4477 0.4642
```

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