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       0.2504 0.0347 20000 0.1409 0.1605 0.1828 0.3189 0.3412 0.3659
#> b        0.5113 0.0314 20000 0.4102 0.4300 0.4496 0.5727 0.5906 0.6127
#> a        0.4296 0.0328 20000 0.3214 0.3456 0.3660 0.4948 0.5132 0.5334
#> Y~~Y     1.0041 0.0447 20000 0.8581 0.8890 0.9169 1.0926 1.1197 1.1500
#> M~~M     1.0279 0.0459 20000 0.8821 0.9121 0.9394 1.1194 1.1483 1.1881
#> indirect 0.2197 0.0215 20000 0.1534 0.1669 0.1788 0.2635 0.2779 0.2980
#> direct   0.2504 0.0347 20000 0.1409 0.1605 0.1828 0.3189 0.3412 0.3659
```

```
#> total    0.4701 0.0362 20000 0.3516 0.3784 0.3987 0.5418 0.5636 0.5910
```

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        1.219953e-03 -4.356365e-04 -4.780540e-06  1.128329e-05  1.490284e-05
#> b        -4.356365e-04  9.810192e-04  8.509137e-06  7.631810e-06 -9.260649e-06
#> a        -4.780540e-06  8.509137e-06  1.052212e-03 -7.514231e-06  1.052093e-05
#> Y~~Y      1.128329e-05  7.631810e-06 -7.514231e-06  1.988945e-03  4.637121e-06
#> M~~M      1.490284e-05 -9.260649e-06  1.052093e-05  4.637121e-06  2.140591e-03
#> X~~X      2.906520e-06 -1.423420e-05  1.271657e-05 -1.516877e-05  3.558264e-06
#> indirect -1.893278e-04  4.258541e-04  5.413546e-04 -1.200858e-06  1.898231e-06
#> direct    1.219953e-03 -4.356365e-04 -4.780540e-06  1.128329e-05  1.490284e-05
#> total     1.030626e-03 -9.782399e-06  5.365741e-04  1.008244e-05  1.680107e-05
#>                    X~~X      indirect        direct         total
#> cp        2.906520e-06 -1.893278e-04  1.219953e-03  1.030626e-03
#> b        -1.423420e-05  4.258541e-04 -4.356365e-04 -9.782399e-06
#> a         1.271657e-05  5.413546e-04 -4.780540e-06  5.365741e-04
#> Y~~Y     -1.516877e-05 -1.200858e-06  1.128329e-05  1.008244e-05
```

```
#> M~~M      3.558264e-06  1.898231e-06  1.490284e-05  1.680107e-05
#> X~~X      1.894221e-03  6.807565e-08  2.906520e-06  2.974596e-06
#> indirect  6.807565e-08  4.606656e-04 -1.893278e-04  2.713379e-04
#> direct    2.906520e-06 -1.893278e-04  1.219953e-03  1.030626e-03
#> total     2.974596e-06  2.713379e-04  1.030626e-03  1.301963e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2030 0.0279 20000 0.1078 0.1305 0.1488 0.2575 0.2740 0.2990
#> b         0.4606 0.0257 20000 0.3685 0.3928 0.4095 0.5103 0.5259 0.5434
#> a         0.3866 0.0269 20000 0.2962 0.3163 0.3326 0.4384 0.4536 0.4757
#> Y~~Y      0.6743 0.0241 20000 0.5936 0.6107 0.6255 0.7203 0.7341 0.7502
#> M~~M      0.8505 0.0208 20000 0.7737 0.7942 0.8078 0.8894 0.8999 0.9122
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1781 0.0162 20000 0.1268 0.1365 0.1469 0.2098 0.2208 0.2332
#> direct    0.2030 0.0279 20000 0.1078 0.1305 0.1488 0.2575 0.2740 0.2990
#> total     0.3811 0.0270 20000 0.2867 0.3082 0.3275 0.4325 0.4489 0.4691
```

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