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:

```
if (!require("remotes")) 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. A description of the package and code examples are presented in Pesigan and Cheung (2023).

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       0.2528 0.0332 20000 0.1438 0.1675 0.1881 0.3182 0.3384 0.3620
#> b        0.4463 0.0304 20000 0.3487 0.3665 0.3868 0.5067 0.5244 0.5501
#> a        0.4663 0.0313 20000 0.3674 0.3864 0.4048 0.5274 0.5469 0.5683
#> Y~~Y     0.9665 0.0433 20000 0.8226 0.8539 0.8817 1.0512 1.0781 1.1072
#> M~~M     1.0591 0.0472 20000 0.9067 0.9376 0.9662 1.1514 1.1781 1.2088
```

```
#> indirect 0.2081 0.0200 20000 0.1477 0.1598 0.1704 0.2486 0.2629 0.2767
#> direct   0.2528 0.0332 20000 0.1438 0.1675 0.1881 0.3182 0.3384 0.3620
#> total    0.4609 0.0332 20000 0.3527 0.3760 0.3957 0.5261 0.5467 0.5686
```

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        1.087802e-03 -4.240069e-04 -1.345322e-06  4.975948e-06 -1.107599e-05
#> b        -4.240069e-04  9.169547e-04 -9.313440e-06  1.374765e-05  6.011219e-06
#> a        -1.345322e-06 -9.313440e-06  9.896524e-04 -1.468060e-05  1.679025e-05
#> Y~~Y      4.975948e-06  1.374765e-05 -1.468060e-05  1.852219e-03 -2.746276e-07
#> M~~M     -1.107599e-05  6.011219e-06  1.679025e-05 -2.746276e-07  2.256246e-03
#> X~~X     -1.329441e-06 -4.141096e-06  1.103868e-05  4.774324e-06  2.717746e-05
#> indirect -1.984074e-04  4.233594e-04  4.372316e-04  1.730420e-10  1.029041e-05
#> direct    1.087802e-03 -4.240069e-04 -1.345322e-06  4.975948e-06 -1.107599e-05
#> total     8.893950e-04 -6.475379e-07  4.358862e-04  4.976122e-06 -7.855797e-07
#>                    X~~X      indirect        direct         total
#> cp       -1.329441e-06 -1.984074e-04  1.087802e-03  8.893950e-04
#> b        -4.141096e-06  4.233594e-04 -4.240069e-04 -6.475379e-07
```

```
#> a         1.103868e-05  4.372316e-04 -1.345322e-06  4.358862e-04
#> Y~~Y      4.774324e-06  1.730420e-10  4.975948e-06  4.976122e-06
#> M~~M      2.717746e-05  1.029041e-05 -1.107599e-05 -7.855797e-07
#> X~~X      2.327770e-03  2.841314e-06 -1.329441e-06  1.511872e-06
#> indirect  2.841314e-06  3.933819e-04 -1.984074e-04  1.949745e-04
#> direct   -1.329441e-06 -1.984074e-04  1.087802e-03  8.893950e-04
#> total     1.511872e-06  1.949745e-04  8.893950e-04  1.084369e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2208 0.0284 20000 0.1298 0.1475 0.1641 0.2756 0.2926 0.3121
#> b        0.4280 0.0268 20000 0.3386 0.3600 0.3748 0.4801 0.4963 0.5133
#> a        0.4248 0.0259 20000 0.3381 0.3553 0.3728 0.4742 0.4891 0.5062
#> Y~~Y     0.6878 0.0241 20000 0.6100 0.6257 0.6394 0.7339 0.7474 0.7632
#> M~~M     0.8196 0.0220 20000 0.7438 0.7607 0.7752 0.8610 0.8737 0.8857
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1818 0.0161 20000 0.1320 0.1413 0.1506 0.2143 0.2243 0.2371
#> direct   0.2208 0.0284 20000 0.1298 0.1475 0.1641 0.2756 0.2926 0.3121
#> total    0.4026 0.0264 20000 0.3159 0.3332 0.3494 0.4526 0.4661 0.4850
```

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

Pesigan, I. J. A., & Cheung, S. F. (2023). Monte Carlo confidence intervals for the indirect effect with missing data. *Behavior Research Methods*. https://doi.org/10.3758/s13428-023-02114-4

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. (2023). *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