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 = 0.05)

#> Monte Carlo Confidence Intervals
#>             est     se     R   2.5%  97.5%
#> cp       0.2150 0.0352 20000 0.1466 0.2844
#> b        0.5449 0.0324 20000 0.4813 0.6077
#> a        0.4873 0.0310 20000 0.4265 0.5477
#> Y~~Y     1.0066 0.0452 20000 0.9182 1.0949
#> M~~M     0.9647 0.0430 20000 0.8805 1.0491
```

```
#> X~~X     1.0210 0.0000 20000 1.0210 1.0210
#> indirect 0.2655 0.0230 20000 0.2221 0.3117
#> direct   0.2150 0.0352 20000 0.1466 0.2844
#> total    0.4806 0.0357 20000 0.4108 0.5505
```

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 = 0.05)
vcov(unstd)

#>                      cp             b             a          Y~~Y          M~~M
#> cp        1.221481e-03 -5.152732e-04  4.412774e-06 -3.872998e-06 -1.425833e-06
#> b        -5.152732e-04  1.048567e-03  1.370764e-06 -6.951904e-06 -3.449051e-06
#> a         4.412774e-06  1.370764e-06  9.455504e-04  1.824863e-06 -1.197386e-06
#> Y~~Y     -3.872998e-06 -6.951904e-06  1.824863e-06  2.043803e-03  1.006572e-05
#> M~~M     -1.425833e-06 -3.449051e-06 -1.197386e-06  1.006572e-05  1.861307e-03
#> X~~X      1.487876e-06  4.753877e-06 -1.269400e-05 -2.160775e-06 -2.135659e-05
#> indirect -2.482918e-04  5.113632e-04  5.158728e-04 -2.851905e-06 -2.234931e-06
#> direct    1.221481e-03 -5.152732e-04  4.412774e-06 -3.872998e-06 -1.425833e-06
#> total     9.731897e-04 -3.910022e-06  5.202856e-04 -6.724903e-06 -3.660765e-06
#>                    X~~X      indirect        direct         total
#> cp        1.487876e-06 -2.482918e-04  1.221481e-03  9.731897e-04
```

```
#> b         4.753877e-06  5.113632e-04 -5.152732e-04 -3.910022e-06
#> a        -1.269400e-05  5.158728e-04  4.412774e-06  5.202856e-04
#> Y~~Y     -2.160775e-06 -2.851905e-06 -3.872998e-06 -6.724903e-06
#> M~~M     -2.135659e-05 -2.234931e-06 -1.425833e-06 -3.660765e-06
#> X~~X      2.117213e-03 -4.914782e-06  1.487876e-06 -3.426906e-06
#> indirect -4.914782e-06  5.311055e-04 -2.482918e-04  2.828137e-04
#> direct    1.487876e-06 -2.482918e-04  1.221481e-03  9.731897e-04
#> total    -3.426906e-06  2.828137e-04  9.731897e-04  1.256003e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1757 0.0283 20000 0.0813 0.1016 0.1199 0.2313 0.2486 0.2676
#> b        0.4842 0.0261 20000 0.3925 0.4154 0.4332 0.5356 0.5513 0.5657
#> a        0.4482 0.0253 20000 0.3606 0.3820 0.3972 0.4964 0.5116 0.5295
#> Y~~Y     0.6584 0.0243 20000 0.5787 0.5948 0.6099 0.7043 0.7176 0.7359
#> M~~M     0.7992 0.0226 20000 0.7196 0.7383 0.7536 0.8422 0.8541 0.8700
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2170 0.0173 20000 0.1597 0.1732 0.1834 0.2515 0.2628 0.2762
#> direct   0.1757 0.0283 20000 0.0813 0.1016 0.1199 0.2313 0.2486 0.2676
#> total    0.3927 0.0267 20000 0.3054 0.3217 0.3394 0.4439 0.4597 0.4778
```

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*, *56*(3), 1678–1696. 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. (2024). *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