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       0.2719 0.03572 20000 0.1481 0.1797 0.2011 0.3416 0.3639 0.3877
#> b        0.5100 0.03174 20000 0.4048 0.4278 0.4474 0.5722 0.5918 0.6115
#> a        0.5198 0.03179 20000 0.4155 0.4380 0.4576 0.5818 0.6005 0.6255
#> Y~~Y     1.0313 0.04611 20000 0.8797 0.9159 0.9409 1.1216 1.1507 1.1798
#> M~~M     1.0287 0.04582 20000 0.8811 0.9106 0.9388 1.1180 1.1451 1.1765
#> indirect 0.2651 0.02322 20000 0.1950 0.2076 0.2210 0.3121 0.3267 0.3424
#> direct   0.2719 0.03572 20000 0.1481 0.1797 0.2011 0.3416 0.3639 0.3877
```

```
#> total    0.5370 0.03573 20000 0.4208 0.4433 0.4673 0.6070 0.6281 0.6565
```

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))
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2175 0.0283 20000 0.1218 0.1421 0.1607 0.2719 0.2892 0.3120
#> b        0.4615 0.0260 20000 0.3770 0.3945 0.4100 0.5121 0.5270 0.5430
#> a        0.4595 0.0250 20000 0.3776 0.3947 0.4095 0.5071 0.5219 0.5359
#> Y~~Y     0.6475 0.0241 20000 0.5686 0.5851 0.5994 0.6945 0.7081 0.7229
#> M~~M     0.7888 0.0230 20000 0.7128 0.7276 0.7428 0.8323 0.8442 0.8575
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2121 0.0170 20000 0.1596 0.1702 0.1794 0.2460 0.2570 0.2688
#> direct   0.2175 0.0283 20000 0.1218 0.1421 0.1607 0.2719 0.2892 0.3120
#> total    0.4295 0.0260 20000 0.3390 0.3601 0.3765 0.4789 0.4936 0.5094
```

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