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.

```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       0.2182 0.0340 20000 0.1056 0.1301 0.1512 0.2842 0.3055 0.3273
#> b        0.4769 0.0314 20000 0.3723 0.3965 0.4157 0.5385 0.5579 0.5824
#> a        0.4530 0.0314 20000 0.3506 0.3720 0.3912 0.5142 0.5350 0.5563
#> Y~~Y     1.0119 0.0447 20000 0.8692 0.8981 0.9244 1.0997 1.1275 1.1597
#> M~~M     1.0405 0.0468 20000 0.8852 0.9173 0.9480 1.1311 1.1592 1.1944
#> indirect 0.2160 0.0208 20000 0.1542 0.1655 0.1767 0.2579 0.2714 0.2902
#> direct   0.2182 0.0340 20000 0.1056 0.1301 0.1512 0.2842 0.3055 0.3273
```

```
#> total    0.4342 0.0343 20000 0.3212 0.3442 0.3661 0.5011 0.5203 0.5413
```

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        1.158039e-03 -4.390144e-04 -1.019308e-05 -1.209944e-07 -1.013576e-06
#> b        -4.390144e-04  9.716156e-04 -4.423029e-06  1.935143e-07  1.608158e-05
#> a        -1.019308e-05 -4.423029e-06  9.877173e-04  1.361234e-06  7.478744e-06
#> Y~~Y     -1.209944e-07  1.935143e-07  1.361234e-06  2.047838e-03 -6.300256e-06
#> M~~M     -1.013576e-06  1.608158e-05  7.478744e-06 -6.300256e-06  2.170319e-03
#> X~~X      1.073894e-06  2.646837e-06  1.371974e-05 -2.137914e-05 -6.149838e-06
#> indirect -2.037934e-04  4.385663e-04  4.692516e-04  4.281744e-07  1.066678e-05
#> direct    1.158039e-03 -4.390144e-04 -1.019308e-05 -1.209944e-07 -1.013576e-06
#> total     9.542461e-04 -4.480815e-07  4.590586e-04  3.071800e-07  9.653202e-06
#>                   X~~X      indirect        direct         total
#> cp        1.073894e-06 -2.037934e-04  1.158039e-03  9.542461e-04
#> b         2.646837e-06  4.385663e-04 -4.390144e-04 -4.480815e-07
#> a         1.371974e-05  4.692516e-04 -1.019308e-05  4.590586e-04
#> Y~~Y     -2.137914e-05  4.281744e-07 -1.209944e-07  3.071800e-07
```

```
#> M~~M     -6.149838e-06  1.066678e-05 -1.013576e-06  9.653202e-06
#> X~~X      2.218100e-03  7.649252e-06  1.073894e-06  8.723146e-06
#> indirect  7.649252e-06  4.237421e-04 -2.037934e-04  2.199487e-04
#> direct    1.073894e-06 -2.037934e-04  1.158039e-03  9.542461e-04
#> total     8.723146e-06  2.199487e-04  9.542461e-04  1.174195e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1863 0.0287 20000 0.0899 0.1125 0.1292 0.2423 0.2595 0.2790
#> b        0.4444 0.0267 20000 0.3565 0.3738 0.3910 0.4958 0.5118 0.5268
#> a        0.4150 0.0262 20000 0.3257 0.3458 0.3626 0.4652 0.4811 0.4965
#> Y~~Y     0.6991 0.0243 20000 0.6179 0.6341 0.6504 0.7454 0.7597 0.7744
#> M~~M     0.8278 0.0218 20000 0.7535 0.7685 0.7836 0.8685 0.8804 0.8939
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1844 0.0164 20000 0.1352 0.1440 0.1527 0.2171 0.2280 0.2425
#> direct   0.1863 0.0287 20000 0.0899 0.1125 0.1292 0.2423 0.2595 0.2790
#> total    0.3707 0.0272 20000 0.2802 0.2994 0.3168 0.4231 0.4392 0.4572
```

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