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       0.2345 0.0333 20000 0.1234 0.1490 0.1698 0.3004 0.3191 0.3424
#> b        0.4514 0.0311 20000 0.3469 0.3709 0.3905 0.5122 0.5305 0.5548
#> a        0.4383 0.0312 20000 0.3386 0.3594 0.3780 0.4987 0.5176 0.5407
#> Y~~Y     0.9843 0.0442 20000 0.8424 0.8714 0.8974 1.0711 1.0980 1.1299
#> M~~M     1.0224 0.0460 20000 0.8726 0.9034 0.9325 1.1134 1.1404 1.1661
#> indirect 0.1978 0.0196 20000 0.1403 0.1507 0.1609 0.2376 0.2512 0.2666
#> direct   0.2345 0.0333 20000 0.1234 0.1490 0.1698 0.3004 0.3191 0.3424
```

```
#> total    0.4324 0.0335 20000 0.3285 0.3466 0.3665 0.4976 0.5182 0.5378
```

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        1.107108e-03 -4.058129e-04  6.344456e-06 -9.671897e-06  1.221672e-05
#> b        -4.058129e-04  9.636905e-04  3.907864e-06  6.583729e-06 -7.187945e-06
#> a         6.344456e-06  3.907864e-06  9.730455e-04 -6.173194e-06  1.021776e-05
#> Y~~Y     -9.671897e-06  6.583729e-06 -6.173194e-06  1.952227e-03 -2.152105e-05
#> M~~M      1.221672e-05 -7.187945e-06  1.021776e-05 -2.152105e-05  2.083450e-03
#> X~~X     -6.538738e-06 -6.387499e-06  1.005963e-05  1.899544e-05  8.213108e-06
#> indirect -1.745151e-04  4.241260e-04  4.409668e-04  5.790835e-07  1.396640e-06
#> direct    1.107108e-03 -4.058129e-04  6.344456e-06 -9.671897e-06  1.221672e-05
#> total     9.325928e-04  1.831310e-05  4.473113e-04 -9.092814e-06  1.361336e-05
#>                    X~~X      indirect        direct         total
#> cp       -6.538738e-06 -1.745151e-04  1.107108e-03  9.325928e-04
#> b        -6.387499e-06  4.241260e-04 -4.058129e-04  1.831310e-05
#> a         1.005963e-05  4.409668e-04  6.344456e-06  4.473113e-04
#> Y~~Y      1.899544e-05  5.790835e-07 -9.671897e-06 -9.092814e-06
```

```
#> M~~M      8.213108e-06  1.396640e-06  1.221672e-05  1.361336e-05
#> X~~X      2.200820e-03  1.618957e-06 -6.538738e-06 -4.919781e-06
#> indirect  1.618957e-06  3.858667e-04 -1.745151e-04  2.113516e-04
#> direct   -6.538738e-06 -1.745151e-04  1.107108e-03  9.325928e-04
#> total    -4.919781e-06  2.113516e-04  9.325928e-04  1.143944e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2045 0.0286 20000 0.1106 0.1290 0.1479 0.2602 0.2770 0.2964
#> b         0.4238 0.0269 20000 0.3351 0.3547 0.3705 0.4762 0.4926 0.5097
#> a         0.4070 0.0264 20000 0.3202 0.3361 0.3534 0.4573 0.4726 0.4888
#> Y~~Y      0.7080 0.0244 20000 0.6246 0.6417 0.6586 0.7541 0.7679 0.7844
#> M~~M      0.8343 0.0214 20000 0.7611 0.7766 0.7909 0.8751 0.8871 0.8975
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1725 0.0160 20000 0.1222 0.1327 0.1416 0.2042 0.2147 0.2261
#> direct    0.2045 0.0286 20000 0.1106 0.1290 0.1479 0.2602 0.2770 0.2964
#> total     0.3770 0.0272 20000 0.2834 0.3035 0.3224 0.4288 0.4458 0.4660
```

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