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       0.2282 0.0343 20000 0.1178 0.1389 0.1608 0.2947 0.3164 0.3439
#> b        0.5078 0.0302 20000 0.4105 0.4306 0.4490 0.5673 0.5859 0.6052
#> a        0.5674 0.0309 20000 0.4658 0.4877 0.5070 0.6278 0.6464 0.6712
#> Y~~Y     0.9247 0.0410 20000 0.7965 0.8175 0.8449 1.0052 1.0319 1.0600
#> M~~M     1.0191 0.0457 20000 0.8656 0.9007 0.9297 1.1090 1.1393 1.1697
```

```
#> X~~X     1.0622 0.0000 20000 1.0622 1.0622 1.0622 1.0622 1.0622 1.0622
#> indirect 0.2881 0.0232 20000 0.2167 0.2309 0.2436 0.3344 0.3517 0.3677
#> direct   0.2282 0.0343 20000 0.1178 0.1389 0.1608 0.2947 0.3164 0.3439
#> total    0.5164 0.0336 20000 0.4047 0.4289 0.4507 0.5827 0.6034 0.6273
```

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        1.176793e-03 -5.295995e-04  9.833350e-06  1.788752e-05  2.263142e-06
#> b        -5.295995e-04  9.163862e-04 -7.397603e-06 -1.323987e-05  1.591455e-05
#> a         9.833350e-06 -7.397603e-06  9.685769e-04 -1.319374e-05  2.079680e-06
#> Y~~Y      1.788752e-05 -1.323987e-05 -1.319374e-05  1.710997e-03 -1.920676e-05
#> M~~M      2.263142e-06  1.591455e-05  2.079680e-06 -1.920676e-05  2.056723e-03
#> X~~X      7.563015e-07  1.232451e-05  6.347935e-06 -1.503951e-05  2.392128e-05
#> indirect -2.953131e-04  5.163581e-04  4.874052e-04 -1.429658e-05  9.982457e-06
#> direct    1.176793e-03 -5.295995e-04  9.833350e-06  1.788752e-05  2.263142e-06
#> total     8.814801e-04 -1.324140e-05  4.972385e-04  3.590939e-06  1.224560e-05
#>                    X~~X      indirect        direct         total
#> cp        7.563015e-07 -2.953131e-04  1.176793e-03  8.814801e-04
```

```
#> b         1.232451e-05  5.163581e-04 -5.295995e-04 -1.324140e-05
#> a         6.347935e-06  4.874052e-04  9.833350e-06  4.972385e-04
#> Y~~Y     -1.503951e-05 -1.429658e-05  1.788752e-05  3.590939e-06
#> M~~M      2.392128e-05  9.982457e-06  2.263142e-06  1.224560e-05
#> X~~X      2.257625e-03  1.030608e-05  7.563015e-07  1.106238e-05
#> indirect  1.030608e-05  5.413359e-04 -2.953131e-04  2.460228e-04
#> direct    7.563015e-07 -2.953131e-04  1.176793e-03  8.814801e-04
#> total     1.106238e-05  2.460228e-04  8.814801e-04  1.127503e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1940 0.0288 20000 0.1003 0.1198 0.1379 0.2494 0.2679 0.2876
#> b        0.4885 0.0267 20000 0.3999 0.4195 0.4349 0.5401 0.5566 0.5735
#> a        0.5012 0.0237 20000 0.4219 0.4391 0.4534 0.5470 0.5606 0.5753
#> Y~~Y     0.6288 0.0244 20000 0.5483 0.5651 0.5807 0.6756 0.6910 0.7063
#> M~~M     0.7488 0.0238 20000 0.6690 0.6857 0.7008 0.7944 0.8072 0.8220
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2449 0.0181 20000 0.1882 0.1997 0.2099 0.2806 0.2932 0.3069
#> direct   0.1940 0.0288 20000 0.1003 0.1198 0.1379 0.2494 0.2679 0.2876
#> total    0.4388 0.0257 20000 0.3511 0.3711 0.3880 0.4880 0.5038 0.5177
```

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