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       0.2318 0.0354 20000 0.1096 0.1414 0.1620 0.3009 0.3229 0.3496
#> b        0.5155 0.0318 20000 0.4076 0.4336 0.4534 0.5775 0.5982 0.6171
#> a        0.5137 0.0314 20000 0.4115 0.4333 0.4524 0.5754 0.5924 0.6125
#> Y~~Y     0.9740 0.0433 20000 0.8330 0.8613 0.8889 1.0584 1.0849 1.1114
#> M~~M     0.9528 0.0428 20000 0.8132 0.8420 0.8691 1.0356 1.0638 1.0915
```

```
#> X~~X     0.9755 0.0000 20000 0.9755 0.9755 0.9755 0.9755 0.9755 0.9755
#> indirect 0.2648 0.0230 20000 0.1946 0.2088 0.2209 0.3111 0.3263 0.3418
#> direct   0.2318 0.0354 20000 0.1096 0.1414 0.1620 0.3009 0.3229 0.3496
#> total    0.4966 0.0357 20000 0.3759 0.4055 0.4270 0.5665 0.5873 0.6082
```

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        1.280308e-03 -5.316301e-04  1.085021e-05 -4.719043e-07 -1.619165e-06
#> b        -5.316301e-04  1.033432e-03 -8.425513e-06 -1.631105e-05  7.483306e-07
#> a         1.085021e-05 -8.425513e-06  9.861986e-04  2.264940e-05 -1.108467e-05
#> Y~~Y     -4.719043e-07 -1.631105e-05  2.264940e-05  1.898432e-03  8.977205e-06
#> M~~M     -1.619165e-06  7.483306e-07 -1.108467e-05  8.977205e-06  1.800880e-03
#> X~~X     -5.472468e-06  1.088494e-06 -8.196668e-06  7.382674e-06  6.159015e-06
#> indirect -2.669788e-04  5.261132e-04  5.033358e-04  3.764989e-06 -5.809226e-06
#> direct    1.280308e-03 -5.316301e-04  1.085021e-05 -4.719043e-07 -1.619165e-06
#> total     1.013329e-03 -5.516927e-06  5.141861e-04  3.293085e-06 -7.428392e-06
#>                    X~~X      indirect        direct         total
#> cp       -5.472468e-06 -2.669788e-04  1.280308e-03  1.013329e-03
```

```
#> b         1.088494e-06  5.261132e-04 -5.316301e-04 -5.516927e-06
#> a        -8.196668e-06  5.033358e-04  1.085021e-05  5.141861e-04
#> Y~~Y      7.382674e-06  3.764989e-06 -4.719043e-07  3.293085e-06
#> M~~M      6.159015e-06 -5.809226e-06 -1.619165e-06 -7.428392e-06
#> X~~X      1.893493e-03 -4.361743e-06 -5.472468e-06 -9.834211e-06
#> indirect -4.361743e-06  5.301280e-04 -2.669788e-04  2.631492e-04
#> direct   -5.472468e-06 -2.669788e-04  1.280308e-03  1.013329e-03
#> total    -9.834211e-06  2.631492e-04  1.013329e-03  1.276478e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.1890 0.0288 20000 0.0967 0.1145 0.1319 0.2446 0.2623 0.2828
#> b        0.4681 0.0266 20000 0.3819 0.3980 0.4151 0.5191 0.5354 0.5534
#> a        0.4612 0.0250 20000 0.3762 0.3937 0.4110 0.5083 0.5226 0.5412
#> Y~~Y     0.6636 0.0244 20000 0.5826 0.5996 0.6145 0.7102 0.7249 0.7402
#> M~~M     0.7873 0.0230 20000 0.7071 0.7269 0.7416 0.8311 0.8450 0.8584
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.2159 0.0172 20000 0.1607 0.1724 0.1820 0.2493 0.2600 0.2752
#> direct   0.1890 0.0288 20000 0.0967 0.1145 0.1319 0.2446 0.2623 0.2828
#> total    0.4049 0.0265 20000 0.3158 0.3342 0.3515 0.4556 0.4715 0.4923
```

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