semmcci: Monte Carlo Confidence Intervals

Ivan Jacob Agaloos Pesigan

Installation

You can install the CRAN release of `semmcci` with:

```r
install.packages("semmcci")
```

You can install the development version of `semmcci` from GitHub with:

```r
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       0.2444 0.0346 20000 0.1340 0.1565 0.1761 0.3121 0.3335 0.3578
#> b        0.4670 0.0302 20000 0.3669 0.3899 0.4070 0.5262 0.5448 0.5679
#> a        0.5042 0.0323 20000 0.3922 0.4221 0.4415 0.5680 0.5872 0.6047
#> Y~~Y     0.9785 0.0437 20000 0.8337 0.8648 0.8926 1.0648 1.0908 1.1193
#> M~~M     1.0611 0.0472 20000 0.9103 0.9413 0.9688 1.1541 1.1804 1.2171
#> indirect 0.2355 0.0214 20000 0.1713 0.1829 0.1945 0.2794 0.2946 0.3104
#> direct   0.2444 0.0346 20000 0.1340 0.1565 0.1761 0.3121 0.3335 0.3578
```

```
#> total    0.4798 0.0346 20000 0.3675 0.3920 0.4126 0.5476 0.5701 0.5946
```

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        1.180089e-03 -4.717698e-04  1.358915e-05 -1.169551e-05 -4.953331e-06
#> b        -4.717698e-04  9.322452e-04  1.148299e-06  8.700851e-06  1.019982e-05
#> a         1.358915e-05  1.148299e-06  1.041469e-03 -5.633326e-06 -8.453892e-06
#> Y~~Y     -1.169551e-05  8.700851e-06 -5.633326e-06  1.899731e-03  2.878240e-05
#> M~~M     -4.953331e-06  1.019982e-05 -8.453892e-06  2.878240e-05  2.228661e-03
#> X~~X      3.077006e-06 -5.106295e-06 -1.194074e-07 -2.868000e-05 -2.452718e-05
#> indirect -2.313455e-04  4.705112e-04  4.866968e-04  1.633348e-06  2.246936e-06
#> direct    1.180089e-03 -4.717698e-04  1.358915e-05 -1.169551e-05 -4.953331e-06
#> total     9.487435e-04 -1.258598e-06  5.002859e-04 -1.006216e-05 -2.706395e-06
#>                   X~~X      indirect        direct         total
#> cp        3.077006e-06 -2.313455e-04  1.180089e-03  9.487435e-04
#> b        -5.106295e-06  4.705112e-04 -4.717698e-04 -1.258598e-06
#> a        -1.194074e-07  4.866968e-04  1.358915e-05  5.002859e-04
#> Y~~Y     -2.868000e-05  1.633348e-06 -1.169551e-05 -1.006216e-05
```

```
#> M~~M     -2.452718e-05  2.246936e-06 -4.953331e-06 -2.706395e-06
#> X~~X      2.059571e-03 -2.376234e-06  3.077006e-06  7.007724e-07
#> indirect -2.376234e-06  4.653354e-04 -2.313455e-04  2.339899e-04
#> direct    3.077006e-06 -2.313455e-04  1.180089e-03  9.487435e-04
#> total     7.007724e-07  2.339899e-04  9.487435e-04  1.182733e-03
```

```
MCStd(unstd)
```

```
#> Standardized Monte Carlo Confidence Intervals
#>             est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp       0.2050 0.0285 20000 0.1101 0.1318 0.1494 0.2613 0.2775 0.2964
#> b        0.4465 0.0268 20000 0.3564 0.3752 0.3931 0.4975 0.5153 0.5360
#> a        0.4426 0.0255 20000 0.3570 0.3749 0.3917 0.4916 0.5074 0.5211
#> Y~~Y     0.6776 0.0243 20000 0.5932 0.6137 0.6280 0.7241 0.7371 0.7538
#> M~~M     0.8041 0.0225 20000 0.7285 0.7425 0.7583 0.8466 0.8595 0.8725
#> X~~X     1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1976 0.0167 20000 0.1457 0.1558 0.1651 0.2310 0.2425 0.2539
#> direct   0.2050 0.0285 20000 0.1101 0.1318 0.1494 0.2613 0.2775 0.2964
#> total    0.4026 0.0265 20000 0.3150 0.3331 0.3502 0.4536 0.4681 0.4885
```

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