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       0.2943 0.0375 20000 0.1679 0.1981 0.2203 0.3672 0.3915 0.4189
#> b        0.4527 0.0334 20000 0.3445 0.3682 0.3866 0.5185 0.5396 0.5659
#> a        0.4639 0.0325 20000 0.3607 0.3808 0.4005 0.5274 0.5474 0.5654
#> Y~~Y     1.0713 0.0479 20000 0.9117 0.9494 0.9770 1.1640 1.1920 1.2295
#> M~~M     0.9820 0.0441 20000 0.8378 0.8675 0.8959 1.0687 1.0952 1.1246
#> indirect 0.2100 0.0215 20000 0.1470 0.1580 0.1696 0.2539 0.2675 0.2871
#> direct   0.2943 0.0375 20000 0.1679 0.1981 0.2203 0.3672 0.3915 0.4189
```

```
#> total    0.5043 0.0371 20000 0.3833 0.4069 0.4310 0.5772 0.5976 0.6242
```

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        1.451681e-03 -5.162513e-04  7.587210e-07  1.654173e-05  2.039510e-06
#> b        -5.162513e-04  1.103011e-03  2.534150e-07  1.660595e-05 -9.294270e-06
#> a         7.587210e-07  2.534150e-07  1.075962e-03  1.801036e-05 -1.479724e-05
#> Y~~Y      1.654173e-05  1.660595e-05  1.801036e-05  2.294499e-03 -1.199753e-05
#> M~~M      2.039510e-06 -9.294270e-06 -1.479724e-05 -1.199753e-05  1.917854e-03
#> X~~X     -1.764091e-05 -9.959920e-07  1.026470e-05 -5.436566e-07 -1.036634e-05
#> indirect -2.390762e-04  5.120960e-04  4.874387e-04  1.553085e-05 -1.085070e-05
#> direct    1.451681e-03 -5.162513e-04  7.587210e-07  1.654173e-05  2.039510e-06
#> total     1.212605e-03 -4.155325e-06  4.881975e-04  3.207258e-05 -8.811195e-06
#>                    X~~X      indirect        direct         total
#> cp       -1.764091e-05 -2.390762e-04  1.451681e-03  1.212605e-03
#> b        -9.959920e-07  5.120960e-04 -5.162513e-04 -4.155325e-06
#> a         1.026470e-05  4.874387e-04  7.587210e-07  4.881975e-04
#> Y~~Y     -5.436566e-07  1.553085e-05  1.654173e-05  3.207258e-05
```

```
#> M~~M     -1.036634e-05 -1.085070e-05  2.039510e-06 -8.811195e-06
#> X~~X      1.622053e-03  4.547966e-06 -1.764091e-05 -1.309294e-05
#> indirect  4.547966e-06  4.596726e-04 -2.390762e-04  2.205963e-04
#> direct   -1.764091e-05 -2.390762e-04  1.451681e-03  1.212605e-03
#> total    -1.309294e-05  2.205963e-04  1.212605e-03  1.433201e-03
```

```
MCStd(unstd)

#> Standardized Monte Carlo Confidence Intervals
#>              est     se     R  0.05%   0.5%   2.5%  97.5%  99.5% 99.95%
#> cp        0.2284 0.0289 20000 0.1358 0.1522 0.1709 0.2846 0.3021 0.3245
#> b         0.4006 0.0274 20000 0.3133 0.3297 0.3470 0.4539 0.4712 0.4878
#> a         0.4069 0.0264 20000 0.3147 0.3367 0.3546 0.4577 0.4729 0.4927
#> Y~~Y      0.7129 0.0240 20000 0.6328 0.6486 0.6646 0.7581 0.7729 0.7887
#> M~~M      0.8345 0.0215 20000 0.7572 0.7763 0.7905 0.8743 0.8867 0.9010
#> X~~X      1.0000 0.0000 20000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
#> indirect 0.1630 0.0156 20000 0.1143 0.1245 0.1335 0.1946 0.2043 0.2193
#> direct    0.2284 0.0289 20000 0.1358 0.1522 0.1709 0.2846 0.3021 0.3245
#> total     0.3914 0.0268 20000 0.3007 0.3205 0.3379 0.4425 0.4581 0.4734
```

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