Package ‘SEMsens’

July 18, 2020

Type Package

Title A Tool for Sensitivity Analysis in Structural Equation Modeling

Version 0.2.6

Author Walter Leite [aut],
Zuchao Shen [aut]

Maintainer Walter Leite <walter.leite@coe.ufl.edu>

Description Perform sensitivity analysis in structural equation modeling using meta-heuristic optimization methods (e.g., ant colony optimization and others).
The references for the proposed methods are:
(1) Harring, J. R., McNeish, D. M., & Hancock, G. R. (2017)
<doi:10.1080/10705511.2018.1506925>;
(2) Socha, K., & Dorigo, M. (2008) <doi:10.1016/j.ejor.2006.06.046>.
We also thank Dr. Krzysztof Socha for sharing his thesis and R code,
which provided the base for the development of this package.

Imports lavaan, stats

Depends R (>= 4.0.0)

License GPL-3

Encoding UTF-8

LazyData true

Suggests knitr, rmarkdown

VignetteBuilder rmarkdown, knitr

RoxygenNote 7.1.0

NeedsCompilation no

Repository CRAN

Date/Publication 2020-07-18 10:00:05 UTC

R topics documented:

SEMsens-package *A Tool for Sensitivity Analysis in Structural Equation Modeling*

Description

This package is to help researchers perform and report sensitivity analysis in structural equation modeling using a phantom variable approach proposed by (Harring, McNeish, & Hancock, 2017).

Details

The package covers sensitivity analysis using ant colony optimization and other meta-heuristic optimization methods (in development) to automatically search a phantom variable, if there is any, that meets the optimization function. The current package includes three main functions and they are `gen.sens.pars` function that generates sensitivity parameters, `sa.aco` function that performs sensitivity analysis, and `sens.tables` function that summarizes sensitivity analysis results.

Author(s)

Walter Leite, Zuchao Shen

Maintainer: Walter Leite walter.leite@coe.ufl.edu (University of Florida)

gen.sens.pars *Generate Sensitivity Parameters*

Description

This function can generate a set of path coefficients from a phantom variable to variables in a structural equation model based on given distributions of the rank of optimization target (with probability of using a distribution based on its rank).

Usage

```
gen.sens.pars(dist.mean, dist.rank, n.of.ants, nl, q = 1e-04, k = 50, xi = 0.5)
```

Arguments

`dist.mean`	List of means - coordinates
`dist.rank`	Rank of the archived values of optimization object
`n.of.ants`	Number of ants used in each iteration after the initialization of k length, default value is 10.
`nl`	Neighborhood of the search area
`q`	Locality of the search (0,1), default is 0.0001
`k`	Size of the solution archive, default is 50.
`xi`	Convergence pressure (0,Inf), suggested: (0,1), default is 0.5

Value

Generated sensitivity parameter values (i.e., a matrix with n.of.ants rows and n.of.sens.pars columns)

References

Socha, K., & Dorigo, M. (2008). Ant colony optimization for continuous domains. European Journal of Operational Research, 185(3), 1155-1173.

We thank Dr. Krzysztof Socha for providing us the original code (http://iridia.ulb.ac.be/supp/IridiaSupp2008-001/) for this function.

Examples

```
k <- 50 # size of archive
# Generate dist.mean and dist.rank
dist.mean <- cbind(rnorm(k), rnorm(k), rnorm(k), rnorm(k), rnorm(k))
y <- rowMeans(dist.mean)
dist.rank <- rank(-y, ties.method = "random")
# set up neighborhood
nl <- matrix(NA, k, k-1)
for (i in 1:k){
  nl[i,] <- (1:k)[1:k != i]
 }
my.sens.pars <- gen.sens.pars(dist.mean, dist.rank, n.of.ants = 10,
                              nl, q = 0.0001, k =50, xi = 0.50)
my.sens.pars
```

sa.aco *Sensitivity Analysis for Structural Equation Modeling Using ACO*

Description

This function can perform sensitivity analysis for structural equation modeling using ant colony optimization (ACO).

Usage

```
sa.aco(
  data = NULL,
  sample.cov,
  sample.nobs,
  model,
  sens.model,
  n.of.sens.pars = NULL,
  opt.fun,
  d = NULL,
  paths = NULL,
  verbose = TRUE,
  max.value = Inf,
  max.iter = 1000,
  e = 1e-10,
  n.of.ants = 10,
  k = 50,
  q = 1e-04,
  sig.level = 0.05,
  xi = 0.5,
  seed = NULL
)
```

Arguments

`data` The data set used for analysis.

`sample.cov` covariance matrix

`sample.nobs` Number of observations for covariance matrix

`model` The analytic model of interest set up as a lavaan format.

`sens.model` Sensitivity analysis model template for structural equation modeling with a phantom variable. This is the model of interest with phantom variable and sensitivity parameters added. See examples provided.

`n.of.sens.pars` number of sensitivity parameters added in the sens.model.

`opt.fun` Customized or preset optimization function. The argument can be customized as a function, e.g., opt.fun = quote(new.par$pvalue[paths]-old.par$pvalue[paths]), where new.par and old.par are the parameter estimates from the sensitivity analysis and analytic models, respectively. When opt.fun is 1, the optimization function is the average departure of new estimate from the old estimate divided by the old estimate y <- mean(abs(new.par$est[paths] - old.par$est[paths]))/mean(abs(old.par$est[paths])); When opt.fun is 2, the optimization function is the standard deviation of deviance divided by the old estimate y <- stats::sd(new.par$est[paths] - old.par$est[paths])/ mean(abs(old.par$est[paths])); When opt.fun is 3, the optimization function is the average p value changed or y <- mean(abs(new.par$pvalue[paths] - old.par$pvalue[paths]))

When opt.fun is 4, the optimization function is the average distance from significance level or y <- mean(abs(new.par$pvalue[paths] - rep(sig.level,length(paths))))#’

`d` Domains for initial sampling, default is c(-1 ,1) for all. It can be specified as a list of ranges (e.g., d = list(-1, 1, -1, 1) for two sampling domains).

`paths` Paths in the model to be evaluated in a sensitivity analysis.

`verbose` Print out evaluation process if true, default is TRUE.

`max.value` Maximal value of optimization when used as the stopping criterion

`max.iter` Maximal number of function evaluations when used as the stopping criterion

`e` Maximum error value used when solution quality used as stopping criterion, default is 1e-10.

`n.of.ants` Number of ants used in each iteration after the initialization of k length, default value is 10.

`k`	Size of the solution archive, default is 50.
`q`	Locality of the search (0,1), default is 0.0001
`sig.level`	Significance level, default value is 0.05.
`xi`	Convergence pressure (0,Inf), suggested: (0,1), default is 0.5
`seed`	Random seed if specified, default is NULL.

Value

Sensitivity analysis results

References

Socha, K., & Dorigo, M. (2008). Ant colony optimization for continuous domains. *European Journal of Operational Research, 185*(3), 1155-1173.

Harring, J. R., McNeish, D. M., & Hancock, G. R. (2017). Using phantom variables in structural equation modeling to assess model sensitivity to external misspecification. *Psychological Methods, 22*(4), 616.

We thank Dr. Krzysztof Socha for providing us the original code (http://iridia.ulb.ac.be/supp/IridiaSupp2008-001/) that the current function is based on.

Examples

```
library(lavaan)
# generate data
sim.model <- ' x =~ x1 + 0.8*x2 + 1.2*x3
               y =~ y1 + 0.5*y2 + 1.5*y3
               m ~ 0.5*x
               y ~ 0.5*x + 0.8*m'
set.seed(10)
data <- simulateData(sim.model, sample.nobs = 1000L)
# standardize dataset
data = data.frame(apply(data,2,scale))

# Step 1: Set up the analytic model of interest
model <- 'x =~ x1 + x2 + x3
          y =~ y1 + y2 + y3
          m ~ x
          y ~ x + m'

# Step 2: Set up sensitivity analysis model
#         the sensitivity parameters are phantom1, phantom2 and phantom3
sens.model = 'x =~ x1 + x2 + x3
              y =~ y1 + y2 + y3
              m ~ x
              y ~ x + m
              x ~ phantom1*phantom
              m ~ phantom2*phantom
              y ~ phantom3*phantom
              phantom =~ 0 # added for mean of zero
              phantom ~~ 1*phantom' # added for unit variance
```

```
# Step 3: check the analytic model results and decide parameter of interests
#         for sensitivity analysis
old.model = lavaan::lavaanify(model = model, auto = TRUE, model.type="sem")
old.out = lavaan::lavaan(model = old.model, data = data)
old.par = lavaan::standardizedSolution(old.out, type="std.all")
old.par # we are interested in lines 7, 8 and 9 for the indirect and direct effects

# Step 4: perform sensitivity analysis
my.sa <- sa.aco(data, model = model, sens.model = sens.model,
                n.of.sens.pars = 3, k = 5,
                opt.fun = quote(1/abs(new.par$pvalue[9]-0.05)), #p-value
                paths = 9,
                max.iter = 10)
   # Note: We run with k = 5 and max.iter = 10 for illustration purpose in 5 seconds,
 #  please specify them as larger numbers (e.g., default value of k = 50 and mat.iter = 1000)


# Step 5: summarize sensitivity analysis results
tables <- sens.tables(my.sa)
tables
```

sens.tables *Summary of sensitivity analysis results*

Description

This function can summarize the sensitivity analysis results from `sa.aco` function.

Usage

```
sens.tables(expr = NULL, sig.level = 0.05, choice = NULL)
```

Arguments

`expr`	Returned object of `sa.aco` function.
`sig.level`	Significance level, default value is 0.05.
`choice`	Set up the length of summary; default is all.

Value

List of summary tables

Examples

```
# see examples in the \code{\link{sa.aco}} function
```

Index