Package ‘SEMsens’

October 12, 2022

Type Package

Title A Tool for Sensitivity Analysis in Structural Equation Modeling

Version 1.5.5

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) Leite, W., & Shen, Z., Marcoulides, K., Fish, C., & Harring, J. (2022).
<doi:10.1080/10705511.2021.1881786>
(2) Harring, J. R., McNeish, D. M., & Hancock, G. R. (2017)
<doi:10.1080/10705511.2018.1506925>;
(3) Fisk, C., Harring, J., Shen, Z., Leite, W., Suen, K., & Marcoulides, K.
(2022). <doi:10.1177/00131644211073121>;
(4) Socha, K., & Dorigo, M. (2008) <doi:10.1016/j.ejor.2006.06.046>.
We also thank Dr. Krzysztof Socha for sharing his research on ant colony optimization
algorithm with continuous domains and associated R code,
which provided the base for the development of this package.

Imports lavaan, stats

Depends R (>= 3.5.0)

License GPL-3

Encoding UTF-8

LazyData true

Suggests knitr, rmarkdown

VignetteBuilder rmarkdown, knitr

RoxygenNote 7.2.0

NeedsCompilation no

Author Walter Leite [aut, cre],
Zuchao Shen [aut],
Charles Fisk [aut],
King Yiu Suen [aut],
Katerina Marcoulides [aut],
Gail Fish [ctb],
YongSeok Lee [ctb],

Sanaz Nazari [ctb],
Jia Quan [ctb],
Eric Wright [ctb],
Huibin Zhang [ctb]

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

Repository CRAN

Date/Publication 2022-08-30 22:30:14 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). The specific reference is Leite, W., & Shen, Z., Marcoulides, K., Fish, C., & Harring, J. (in press). Using ant colony optimization for sensitivity analysis in structural equation modeling. Structural Equation Modeling: A Multidisciplinary Journal.

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 (running in background for the sa.aco function), `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.neighbors.tabu — *A Function that Generates a List of Neighbors in Tabu Search*

Description

This function generates a list of neighbors for tabu search that can be used in tabu search for sensitivity analysis.

Usage

```
gen.neighbors.tabu(
  current.param,
  maximum,
  neigh.size,
  tabu.list,
  max.len,
  range,
  r,
  f,
  max.attempts = 10
)
```

Arguments

`current.param`	The center of the hypercubes.
`maximum`	Logical. Maximize the objective function if TRUE, minimize the objective function if FALSE.
`neigh.size`	Number of neighbors to search for in each iteration.
`tabu.list`	The list of tabu.
`max.len`	The length of the largest hypercube.
`range`	The range for the parameter space in the tabu search.
`r`	Radius of a tabu ball.
`f`	The objective function to be optimized.
`max.attempts`	The maximum number of attempts to find a neighbor that is not near the points in the tabu list, default is 10.

Value

A list of information about the best neighbor, including best parameters, objective function values, and the model.

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 = 500,
  xi = 0.5
)
```

Arguments

`dist.mean`	List of means - coordinates
`dist.rank`	Rank of the archived values of objective function
`n.of.ants`	Number of ants used in each iteration after the initialization of k converged sensitivity analysis models, 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 100.
`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

Leite, W., & Shen, Z., Marcoulides, K., Fish, C., & Harring, J. (in press). Using ant colony optimization for sensitivity analysis in structural equation modeling. Structural Equation Modeling: A Multidisciplinary Journal.

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 Ant Colony Optimization (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,
  opt.fun,
  d = NULL,
  paths = NULL,
  verbose = TRUE,
  max.value = Inf,
  max.iter = 1000,
  e = 1e-10,
  n.of.ants = 10,
  k = 100,
  q = 1e-04,
  sig.level = 0.05,
  rate.of.conv = 0.1,
  measurement = FALSE,
  xi = 0.5,
```

```
  seed = NULL,
  ...
)
```

Arguments

`data`	The data set used for analysis.
`sample.cov`	covariance matrix for SEM analysis when data are not available.
`sample.nobs`	Number of observations for covariance matrix.
`model`	The analytic model of interest.
`sens.model`	Sensitivity analysis model template for structural equation modeling with a phantom variable. This is the model of interest with a phantom variable and sensitivity parameters added. See examples provided.
`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)))); When opt.fun is 5, we assess the change of RMSEA or y <- abs(unname(lavaan::fitmeasures(new.out)["rm - unname(lavaan::fitmeasures(old.out)["rmsea"])); When opt.fun is 6, we optimize how close RMSEA is to 0.05 or y <- 1/abs(unname(lavaan::fitmeasures(new.out)["rmsea"]) - 0.05).
`d`	Domains for initial sampling, default is c(-1 ,1) for all sensitivity analysis parameters. It can be specified as a list of ranges. For example, d = list(-0.8, 0.8, -0.9, 0.9) for two sampling domains with the first from -0.8 to 0.8 and the second from -0.9 to 0.9.
`paths`	Paths in the model to be evaluated in a sensitivity analysis. If not specified, all paths will be evaluated. It can be specified in a numeric format or in a model format. For example, if we evaluate the changes (in p value or parameter estimation) for paths in an analytic model, we may specify paths in a model format, e.g., paths = 'm ~ x y ~ x + m'. Or, alternatively, as specify paths = c(1:3) if these paths present in line 1 to 3 in the sensitivity analysis model results.
`verbose`	Print out evaluation process if TRUE, default is TRUE.
`max.value`	Maximal value of optimization when used as the stopping criterion. Default is infinite.
`max.iter`	Maximal number of function evaluations when used as the stopping criterion.
`e`	Maximum error value used when solution quality used as the stopping criterion, default is 1e-10.

n.of.ants	Number of ants used in each iteration after the initialization of k converged sensitivity analysis models, default value is 10.
k	Size of the solution archive, default is 100.
q	Locality of the search (0,1), default is 0.0001.
sig.level	Significance level, default value is 0.05.
rate.of.conv	The convergence rate threshold for sensitivity analysis models, default is .10.
measurement	Logical. If TRUE, the argument paths will include measurement paths in the lavaanify format. Default is FALSE.
xi	Convergence pressure (0, Inf), suggested: (0,1), default is 0.5.
seed	Random seed if specified, default is NULL.
...	Additional arguments from the lavaan package.

Value

Sensitivity analysis results, including the number of evaluations (n.eval), number of iterations (n.iter), the maximum value of the objective function (max.y) and associated sensitivity parameters values (phantom.coef), analytic model (old.model), its results (old.model.par) and fit measures (old.model.fit), sensitivity analysis model (sens.model), its fit measures (sens.fit), outcome of the objective function (outcome), sensitivity parameters across all converged evaluations (sens.pars), sensitivity analysis model results (model.results), analytic model results (old.out), and the first converged sensitivity analysis model results (sens.out).

References

Leite, W., & Shen, Z., Marcoulides, K., Fish, C., & Harring, J. (accepted). Using ant colony optimization for sensitivity analysis in structural equation modeling. Structural Equation Modeling: A Multidisciplinary Journal.

Socha, K., & Dorigo, M. (2008). Ant colony optimization for continuous domains. *European Journal of Operational Research, 185*(3), 1155-1173. <doi:10.1016/j.ejor.2006.06.046>

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), 616-631. <doi:10.1080/10705511.2018.1506925>

We thank Dr. Krzysztof Socha for providing us the ACO code for continuous domains (http://iridia.ulb.ac.be/supp/IridiaSupp2
001/) that the current function is based on.

Examples

```
library(lavaan)
# Generate data, this is optional as lavaan also takes variance covariance matrix
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 the sensitivity analysis model.
#        The sensitivity parameters are phantom1, phantom2, and phantom3 in this example.
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: Set up the paths of interest to be evaluated in sensitivity analysis.
# Suppose we are interested in all direct and indirect paths.
  paths <- 'm ~ x
            y ~ x + m'

# Step 4: Perform sensitivity analysis
my.sa <- sa.aco(data, model = model, sens.model = sens.model,
                opt.fun = 3, k = 5, #p-value
                paths = paths,
                max.iter = 10)
#Note, please specify larger numbers for k (e.g., 100) and max.iter (e.g., 1000)

# Step 5: Summarize sensitivity analysis results.
# See sens.tables function for explanation of results.
tables <- sens.tables(my.sa)
```

`sa.sa` *Sensitivity Analysis for Structural Equation Modeling Using Simulated Annealing (SA)*

Description

This function can perform sensitivity analysis for structural equation modeling using simulated annealing (SA)

Usage

```
sa.sa(
```

```
data = NULL,
  sample.cov,
  sample.nobs,
  model,
  sens.model,
  opt.fun = 1,
  d = NULL,
  paths = NULL,
  verbose = TRUE,
  n.iter = 10,
  e = 1e-10,
  k = 10,
  sig.level = 0.05,
  Ntemps = 10,
  C.criteria = 1,
  steepness = 6,
  measurement = FALSE
)
```

Arguments

`data`	The data set used for analysis.
`sample.cov`	covariance matrix for SEM analysis when data are not available.
`sample.nobs`	Number of observations for covariance matrix.
`model`	The analytic model of interest.
`sens.model`	Sensitivity analysis model template for structural equation modeling with a phantom variable. This is the model of interest with a phantom variable and sensitivity parameters added. See examples provided.
`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$est.std[paths] - old.par$est.std[paths]))/mean(abs(old.par$est.std[patl When opt.fun is 2, the optimization function is the standard deviation of deviance divided by the old estimate y <- stats::sd(new.par$est.std[paths] - old.par$est.std[paths])/ mean(abs(old.par$est.std[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)))); When opt.fun is 5, we assess the change of RMSEA or y <- abs(unname(lavaan::fitmeasures(new.out)["rmsea"]) - unname(lavaan::fitmeasures(old.out)[" When opt.fun is 6, we optimize how close RMSEA is to 0.05 or y <- 1/abs(unname(lavaan::fitmeasures(nev - 0.05).
`d`	Domains for initial sampling, default is c(-1 ,1) for all sensitivity analysis parameters.

`paths`	Paths in the model to be evaluated in a sensitivity analysis. If not specified, all paths will be evaluated. It can be specified in a numeric format or in a model format. For example, if we evaluate the changes (in p value or parameter estimation) for paths in an analytic model, we may specify paths in a model format, e.g., paths = 'm ~ x y ~ x + m'. Or, alternatively, as specify paths = c(1:3) if these paths present in line 1 to 3 in the sensitivity analysis model results.
`verbose`	Print out evaluation process if TRUE, default is TRUE.
`n.iter`	Maximal number of function evaluations within each temperature.
`e`	Maximum error value used when solution quality used as the stopping criterion, default is 1e-10.
`k`	Size of the solution archive, default is 100.
`sig.level`	Significance level, default value is 0.05.
`Ntemps`	Number of temperatures that the algorithm visits. Default value is 10.
`C.criteria`	Convergence criterion. Default value is 1.
`steepness`	Steepness of cooling schedule. Default value is 6.
`measurement`	Logical. If TRUE, the argument paths will include measurement paths in the lavaanify format. Default is FALSE.

Value

Sensitivity analysis results, including the number of evaluations (n.eval), number of iterations (n.iter), the maximum value of the objective function (max.y) and associated sensitivity parameters values (phantom.coef), analytic model (old.model), its results (old.model.par) and fit measures (old.model.fit), sensitivity analysis model (sens.model), its fit measures (sens.fit), outcome of the objective function (outcome), sensitivity parameters across all converged evaluations (sens.pars), sensitivity analysis model results (model.results), analytic model results (old.out), and the first converged sensitivity analysis model results (sens.out).

References

Fisk, C., Harring, J., Shen, Z., Leite, W., Suen, K., & Marcoulides, K. (2022). Using simulated annealing to investigate sensitivity of SEM to external model misspecification. Educational and Psychological Measurement. <doi:10.1177/00131644211073121>

Examples

```
library(lavaan)
# Generate data, this is optional as lavaan also takes variance covariance matrix
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 the sensitivity analysis model.
#        The sensitivity parameters are phantom1, phantom2, and phantom3 in this example.
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: Set up the paths of interest to be evaluated in sensitivity analysis.
# Suppose we are interested in all direct and indirect paths.
  paths <- 'm ~ x
            y ~ x + m'

# Step 4: Perform sensitivity analysis
mysa <- sa.sa(data = data, model = model,
sens.model = sens.model, paths = paths,
 n.iter = 3,  Ntemps = 2)
# We set Ntemps = 2 and n.iter = 3 to reduce the running time.
# You may leave them as default values or specify larger numbers.
```

---

sa.tabu *Sensitivity Analysis for SEM using Tabu Search*

Description

This function conducts sensitivity analysis for SEM using tabu search.

Usage

```
sa.tabu(
  model,
  sens.model,
  data = NULL,
  sample.cov = NULL,
  sample.nobs = NULL,
  opt.fun = 1,
  sig.level = 0.05,
  ...
)
```

Arguments

- `model` The analytic model of interest.
- `sens.model` Sensitivity analysis model template for structural equation modeling with a phantom variable. This is the model of interest with a phantom variable and sensitivity parameters added. See examples provided.
- `data` The data set used for analysis.
- `sample.cov` covariance matrix for SEM analysis when data are not available.
- `sample.nobs` Number of observations for covariance matrix.
- `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)))); When opt.fun is 5, we assess the change of RMSEA or y <- abs(unname(lavaan::fitmeasures(new.out)["rm - unname(lavaan::fitmeasures(old.out)["rmsea"])); When opt.fun is 6, we optimize how close RMSEA is to 0.05 or y <- 1/abs(unname(lavaan::fitmeasures(new.out)["rmsea"]) - 0.05).
- `sig.level` Significance level, default value is 0.05.
- `...` Additional arguments from the lavaan package.

Value

A list with five components: model: The old model; old.model.par: Parameters of the old model; model.results: Sensitivity analysis model results; best.param: Parameters that optimize the objective function; best.obj: The optimized objective function value; sens.par: NULL. Included for compatibility; outcome: NULL. Included for compatibility.

Examples

```
library(lavaan)
# Generate data, this is optional as lavaan also takes variance covariance matrix
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 the sensitivity analysis model.
#        The sensitivity parameters are phantom1, phantom2, and phantom3 in this example.
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: Set up the paths of interest to be evaluated in sensitivity analysis.
# Suppose we are interested in all direct and indirect paths.
  paths <- 'm ~ x
            y ~ x + m'
# Step 4: Perform sensitivity analysis
out <- sa.tabu(model = model,
               sens.model = sens.model,
               data = data,
               opt.fun = 1,
               max.iter = 2,
               max.iter.obj = 2)
 # Note, please specify larger numbers for
 # max.iter (e.g., 50) and max.iter.obj (e.g., 10)

# Step 5: Summarize sensitivity analysis results.
# See sens.tables function for explanation of results.
tables <- sens.tables(out)
```

`sa.tabu.helper` *Tabu Search to Optimize Functions of Continuous Variables*

Description

A helper function that implements the main logic of tabu search to optimize objective functions in continuous domains.

Usage

```
sa.tabu.helper(
  n.var,
  f,
  maximum = FALSE,
```

```
  max.len = 1,
  max.tabu.size = 5,
  neigh.size = NULL,
  max.iter = NULL,
  max.iter.obj = NULL,
  range = c(-1, 1),
  r = 1e-05,
  verbose = TRUE
)
```

Arguments

`n.var`	The dimension of search space.
`f`	The objective function to be optimized.
`maximum`	Logical. Maximize the objective function if TRUE, minimize the objective function if FALSE.
`max.len`	The length of the largest hypercube.
`max.tabu.size`	The maximum size of the tabu list.
`neigh.size`	The number of neighbors to search for in each iteration.
`max.iter`	The maximum number of iterations.
`max.iter.obj`	The maximum number of successive iterations without any improvement of the objective function value.
`range`	The range for the parameter space in the tabu search.
`r`	Radius of a tabu ball.
`verbose`	Logical. Print the current best and overall best objective functions if TRUE, no printing if FALSE.

Value

A list with three components: best.param (vector): the best set of parameters found; best.obj (scalar): the value of obj.fun corresponding to best.param; and model.history: the histry of model results.

References

P., & Berthiau, G. (1997). Fitting of tabu search to optimize functions of continuous variables. International journal for numerical methods in engineering, 40(13), 2449-2457.

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, path = TRUE, sort = TRUE)
```

Arguments

`expr`	Returned object of `sa.aco` function.
`sig.level`	Significance level, default value is 0.05.
`path`	Logical, if TRUE, the function only present results for structural paths. If FALSE, the function will present results for all paths (including structural paths and measurement paths). Default value is TRUE.
`sort`	Logical, if TRUE, the function will present sorted results. If FALSE, the function will present unsorted results. Default value is TRUE.

Value

Lists of 5 summary tables. The first table (sens.summary) provides analytic model results (model path coefficient/model.est, p value/pvalue), mean, minimum, and maximum values of estimated path coefficients across all sensitivity analysis models (mean.est.sens, min.est.sens, and max.est.sens). The second table (phan.paths) provides the summary of sensitivity parameters, including the mean, minimum, and maximum values of each sensitivity parameters (mean.phan, min.phan, max.phan). The third table (phan.min) provides the sensitivity parameters that lead to the minimum path coefficient estimation in a sensitivity analysis model. The fourth table (phan.max) provides the sensitivity parameters that lead to the maximum path coefficient estimation in a sensitivity analysis model. The fifth table (p.paths) provides the sensitivity parameters, if any, that lead to the change of p value across the significance level.

References

Leite, W., & Shen, Z., Marcoulides, K., Fish, C., & Harring, J. (in press). Using ant colony optimization for sensitivity analysis in structural equation modeling. Structural Equation Modeling: A Multidisciplinary Journal.

Examples

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

`smith19.use` *Smith19.use data.*

Description

A dataset is used in Parenting Risk and Resilience, Social-emotional Readiness, and Reading Achievement in Kindergarten Children from Low-income Families Model. Author(s): Sondra Smith-Adcock, Walter Leite, Yasemine Kaya & Ellen Amatea. **Source:** Journal of Child and Family Studies, vol 28(Oct., 2019), pp. 2826-2841. Published by: Springer Nature in Journal of Child and Family studies DOI:<https://doi.org/10.1007/s10826-019-01462-0>

Usage

```
smith19.use
```

Format

A data frame with 3444 observations and 39 variables

Details

The dataset was taken from the public-use data of the Early Childhood Longitudinal Study – Kindergarten Class of 1998-99 of the National Center for Educational Statistics (https://nces.ed.gov/ecls/kindergarten.asp/). This dataset should not be combined with other data for the purpose of identifying participants.

Index