

# Package ‘phantSEM’

July 23, 2025

**Title** Create Phantom Variables in Structural Equation Models for Sensitivity Analyses

**Version** 1.0.0.0

**Description** Create phantom variables, which are variables that were not observed, for the purpose of sensitivity analyses for structural equation models. The package makes it easier for a user to test different combinations of covariances between the phantom variable(s) and observed variables. The package may be used to assess a model's or effect's sensitivity to temporal bias (e.g., if cross-sectional data were collected) or confounding bias.

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**Imports** tidyrr(>= 1.3.0), dplyr(>= 1.1.0), corpcor(>= 1.6.10),  
lavaan(>= 0.6-11)

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0), devtools, tidyverse

**VignetteBuilder** knitr

**Depends** R (>= 3.5.1)

**Config/testthat/edition** 3

**License** MIT + file LICENSE

**NeedsCompilation** no

**Author** Alexis Georgeson [aut, cre, cph] (ORCID:  
<<https://orcid.org/0000-0002-6426-9258>>)

**Maintainer** Alexis Georgeson <[georgeson.alexis@gmail.com](mailto:georgeson.alexis@gmail.com)>

**Repository** CRAN

**Date/Publication** 2023-09-07 08:50:02 UTC

## Contents

ghost_par_est	2
SA_lookup	3
SA_step1	4
SA_step2	5
SA_step3	6

---

ghost_par_est	<i>Provide parameter estimates from sensitivity analysis function</i>
---------------	-----------------------------------------------------------------------

---

**Description**

ghost\_par\_est() Selects certain parameter estimates from the output of the sensitivity analysis.

**Usage**

```
ghost_par_est(step3, parameter_label, remove_NA = FALSE)
```

**Arguments**

step3	The object returned from SA_step3.
parameter_label	The label used for the parameter in the lavaan code.
remove_NA	Remove rows for combinations of phantom variable parameters that resulted in inadmissible solutions in lavaan.

**Value**

A dataframe of the parameter estimates from the lavaan model.

**Examples**

```
# example code

covmatrix <- matrix(c(
  0.25, 0.95, 0.43,
  0.95, 8.87, 2.66,
  0.43, 2.66, 10.86
), nrow = 3, byrow = TRUE)
colnames(covmatrix) <- c("X", "M2", "Y2")

# lavann syntax for observed model
observed <- " M2 ~ X
            Y2 ~ M2+X "

# lavaan output
obs_output <- lavaan::sem(model = observed, sample.cov = covmatrix, sample.nobs = 200)

# lavaan syntax for phantom variable model
phantom <- " M2 ~ M1 + Y1 + a*X
            Y2 ~ M1 + Y1 + b*M2 + cp*X "

Step1 <- SA_step1(
  lavoutput = obs_output,
  mod_obs = observed,
```

```

  mod_phant = phantom
)

phantom_assignment <- list(
  "CovM1X" = 0,
  "CovY1M1" = "CovY2M2",
  "CovY1X" = 0,
  "VarM1" = 1,
  "VarY1" = 1,
  "CovM1M2" = seq(.4, .6, .1),
  "CovY1Y2" = "CovM1M2",
  "CovY1M2" = seq(.2, .4, .1),
  "CovM1Y2" = "CovY1M2"
)
Step2 <- SA_step2(
  phantom_assignment = phantom_assignment,
  step1 = Step1
)
Step3 <- SA_step3(
  step2 = Step2,
  n = 200
)

b_results <- ghost_par_ests(
  step3 = Step3,
  parameter_label = "b",
  remove_NA = TRUE
)

```

---

SA\_lookup

*Lookup Table for Sensitivity Analysis*


---

### Description

SA\_lookup() is used to look up the sensitivity analysis results for a two-wave mediation model when provided with the cross-sectional correlations.

### Usage

```
SA_lookup(CorXM, CorXY, CorMY)
```

### Arguments

CorXM	The observed correlation between predictor X and mediator M.
CorXY	The observed correlation between predictor X and outcome Y.
CorMY	The observed correlation between mediator M and outcome Y.

### Value

Results of a sensitivity analysis with varying cross-lagged and autoregressive correlations.

**Examples**

```
# specify correlations
xm <- .2
xy <- .3
my <- .4

output <- SA_lookup(
  CorXM = xm,
  CorXY = xy,
  CorMY = my
)
```

SA\_step1

*Sensitivity Analysis Function Step 1***Description**

SA\_step1() is used to identify the phantom variables and generate names for their covariance parameters. The output of this function will be used in SA\_step2().

**Usage**

```
SA_step1(lavoutput, mod_obs, mod_phant)
```

**Arguments**

lavoutput	The lavaan output object output from lavaan functions sem() or lavaan() when fitting your observed model.
mod_obs	A lavaan syntax for the observed model.
mod_phant	A lavaan syntax for the phantom variable model.

**Value**

a list containing the names of all phantom covariance parameters.

**Examples**

```
# covariance matrix
covmatrix <- matrix(c(
  0.25, 0.95, 0.43,
  0.95, 8.87, 2.66,
  0.43, 2.66, 10.86
), nrow = 3, byrow = TRUE)
colnames(covmatrix) <- c("X", "M2", "Y2")

# lavann syntax for observed model
observed <- " M2 ~ X
            Y2 ~ M2+X "
```

```

# lavaan output
obs_output <- lavaan::sem(model = observed, sample.cov = covmatrix, sample.nobs = 200)

# lavaan syntax for phantom variable model
phantom <- " M2 ~ M1 + Y1 + a*X
           Y2 ~ M1 + Y1 + b*M2 + cp*X "

Step1 <- SA_step1(
  lavoutput = obs_output,
  mod_obs = observed,
  mod_phant = phantom
)

```

SA\_step2

*Step 2 of sensitivity analysis function***Description**

SA\_step2() is used to assign values to the phantom covariances. There are three options for assigning values to the phantom covariances: 1. Fix phantom covariance to a numeric value (i.e., 0 or 1), 2. Fix phantom covariance to be equal to another covariance, or 3. Test different values for the phantom covariance.

**Usage**

```
SA_step2(phantom_assignment, step1)
```

**Arguments**

phantom\_assignment

A list of all phantom parameter names (copied from SA\_step1() output) which assigns them to be equal to ONE of the following: 1) an observed parameter name, 2) a single numeric value, 3) a sequence of values, or 4) another phantom variable that has been set equal to 1-3.

step1

The output object created in SA\_step1()

**Value**

A list containing test covariance matrices that the phantom model will be fit to.

**Examples**

```

covmatrix <- matrix(c(
  0.25, 0.95, 0.43,
  0.95, 8.87, 2.66,
  0.43, 2.66, 10.86
), nrow = 3, byrow = TRUE)

```

```

colnames(covmatrix) <- c("X", "M2", "Y2")

# lavann syntax for observed model
observed <- " M2 ~ X
            Y2 ~ M2+X "

# lavaan output
obs_output <- lavaan::sem(model = observed, sample.cov = covmatrix, sample.nobs = 200)

# lavaan syntax for phantom variable model
phantom <- " M2 ~ M1 + Y1 + a*X
            Y2 ~ M1 + Y1 + b*M2 + cp*X "

Step1 <- SA_step1(
  lavoutput = obs_output,
  mod_obs = observed,
  mod_phant = phantom
)

phantom_assignment <- list(
  "CovM1X" = 0,
  "CovY1M1" = "CovY2M2",
  "CovY1X" = 0,
  "VarM1" = 1,
  "VarY1" = 1,
  "CovM1M2" = seq(0, .6, .1),
  "CovY1Y2" = "CovM1M2",
  "CovY1M2" = seq(-.6, .6, .1),
  "CovM1Y2" = "CovY1M2"
)

Step2 <- SA_step2(
  phantom_assignment = phantom_assignment,
  step1 = Step1
)

```

---

SA\_step3

*Step 3 of sensitivity analysis function*


---

### Description

SA\_step3() computes the parameter estimates in your phantom model defined in step 1 for the different values provided.

### Usage

```
SA_step3(step2, n)
```

**Arguments**

step2            The object returned from SA\_step2.  
 n                The sample size.

**Value**

A list of parameter estimates from each test covariance matrix.

**Examples**

```
#' @examples
covmatrix <- matrix(c(
  0.25, 0.95, 0.43,
  0.95, 8.87, 2.66,
  0.43, 2.66, 10.86
), nrow = 3, byrow = TRUE)
colnames(covmatrix) <- c("X", "M2", "Y2")

# lavann syntax for observed model
observed <- " M2 ~ X
            Y2 ~ M2+X "

# lavaan output
obs_output <- lavaan::sem(model = observed, sample.cov = covmatrix, sample.nobs = 200)

# lavaan syntax for phantom variable model
phantom <- " M2 ~ M1 + Y1 + a*X
            Y2 ~ M1 + Y1 + b*M2 + cp*X "

Step1 <- SA_step1(
  lavoutput = obs_output,
  mod_obs = observed,
  mod_phant = phantom
)

phantom_assignment <- list(
  "CovM1X" = 0,
  "CovY1M1" = "CovY2M2",
  "CovY1X" = 0,
  "VarM1" = 1,
  "VarY1" = 1,
  "CovM1M2" = seq(.4, .6, .1),
  "CovY1Y2" = "CovM1M2",
  "CovY1M2" = seq(.1, .3, .1),
  "CovM1Y2" = "CovY1M2"
)

Step2 <- SA_step2(
  phantom_assignment = phantom_assignment,
  step1 = Step1
)

Step3 <- SA_step3(
  step2 = Step2,
```

n = 200  
)



# Index

ghost\_par\_ests, [2](#)

SA\_lookup, [3](#)

SA\_step1, [4](#)

SA\_step2, [5](#)

SA\_step3, [6](#)