

# Package ‘simstandard’

July 23, 2025

**Title** Generate Standardized Data

**Version** 0.6.3

**Date** 2021-05-07

**Description**

Creates simulated data from structural equation models with standardized loading. Data generation methods are described in Schneider (2013) <[doi:10.1177/0734282913478046](https://doi.org/10.1177/0734282913478046)>.

**Depends** R (>= 3.4.0)

**License** CC0

**Encoding** UTF-8

**Language** en-US

**RoxygenNote** 7.1.1

**Imports** lavaan, mvtnorm, tibble, stats, magrittr, rlang, purrr

**Suggests** knitr, rmarkdown, ggplot2, dplyr, tidyr, forcats, stringr, testthat, covr, badger

**VignetteBuilder** knitr

**URL** <https://github.com/wjschne/simstandard>

**BugReports** <https://github.com/wjschne/simstandard/issues>

**Maintainer** W. Joel Schneider <[w.joel.schneider@gmail.com](mailto:w.joel.schneider@gmail.com)>

**NeedsCompilation** no

**Author** W. Joel Schneider [aut, cre] (ORCID:  
<<https://orcid.org/0000-0002-8393-5316>>)

**Repository** CRAN

**Date/Publication** 2021-05-07 18:40:02 UTC

## Contents

add_composite_scores . . . . .	2
add_factor_scores . . . . .	3
fixed2free . . . . .	4

get_factor_score_coefficients . . . . .	5
get_factor_score_validity . . . . .	6
get_factor_score_validity_se . . . . .	6
get_model_implied_correlations . . . . .	7
get_model_names . . . . .	8
lav2ram . . . . .	9
matrix2lavaan . . . . .	9
model_complete . . . . .	11
sim_standardized . . . . .	12
sim_standardized_matrices . . . . .	13

<b>Index</b>	<b>14</b>
--------------	-----------

---

add_composite_scores	<i>For each latent variable in a structural model, add a composite score to observed data.</i>
----------------------	--

---

## Description

For each latent variable in a structural model, add a composite score to observed data.

## Usage

```
add_composite_scores(
  d,
  m,
  mu = 0,
  sigma = 1,
  names_suffix = "_Composite",
  keep_observed_scores = TRUE,
  ...
)
```

## Arguments

d	A data.frame with observed data in standardized form (i.e, z-scores)
m	A character string with lavaan model
mu	Score means. Composite scores will also have this mean. Defaults to 0.
sigma	Score standard deviations. Composite scores will also have this standard deviation. Defaults to 1.
names_suffix	A character string added to each composite score name
keep_observed_scores	The observed scores are returned along with the composite scores.
...	parameters passed to simstandardized_matrices

**Value**

data.frame with observed data and estimated factor scores

**Examples**

```
library(simstandard)
# lavaan model
m = "
X =~ 0.9 * X1 + 0.8 * X2 + 0.7 * X3
"

# Make data.frame for two cases
d <- data.frame(
  X1 = c(1.2, -1.2),
  X2 = c(1.5, -1.8),
  X3 = c(1.8, -1.1))

# Compute composite scores for two cases
add_composite_scores(d, m)
```

---

add_factor_scores	<i>For each latent variable in a structural model, add an estimated factor score to observed data.</i>
-------------------	--

---

**Description**

For each latent variable in a structural model, add an estimated factor score to observed data.

**Usage**

```
add_factor_scores(
  d,
  m,
  mu = 0,
  sigma = 1,
  CI = FALSE,
  p = 0.95,
  names_suffix = "_FS",
  keep_observed_scores = TRUE,
  ...
)
```

**Arguments**

d	A data.frame with observed data in standardized form (i.e, z-scores)
m	A character string with lavaan model
mu	Population mean of the observed scores. Factor scores will also have this mean. Defaults to 0.

sigma	Population standard deviation of the observed scores. Factor scores will also have this standard deviation. Defaults to 1.
CI	Add confidence intervals? Defaults to 'FALSE'. If 'TRUE', For each factor score, a lower and upper bound of the confidence interval is created. For example, the lower bound of factor score 'X' is 'X_LB', and the upper bound is 'X_UB'.
p	confidence interval proportion. Defaults to 0.95
names_suffix	A character string added to each factor score name
keep_observed_scores	The observed scores are returned along with the factor scores.
...	parameters passed to simstandardized_matrices

**Value**

data.frame with observed data and estimated factor scores

**Examples**

```
library(simstandard)
# lavaan model
m = "
X =~ 0.9 * X1 + 0.8 * X2 + 0.7 * X3
"

# Make data.frame for two cases
d <- data.frame(
  X1 = c(1.2, -1.2),
  X2 = c(1.5, -1.8),
  X3 = c(1.8, -1.1))

# Compute factor scores for two cases
add_factor_scores(d, m)
```

---

fixed2free

*Remove fixed parameters from a lavaan model*

---

**Description**

Remove fixed parameters from a lavaan model

**Usage**

```
fixed2free(m)
```

**Arguments**

m Structural model represented by lavaan syntax

**Value**

character string representing lavaan model

**Examples**

```
library(simstandard)
# lavaan model with fixed parameters
m = "
Latent_1 =~ 0.9 * Ob_11 + 0.8 * Ob_12 + 0.7 * Ob_13
Latent_2 =~ 0.9 * Ob_21 + 0.6 * Ob_22 + 0.4 * Ob_23
"
# Same model, but with fixed parameters removed.
m_free <- fixed2free(m)
cat(m_free)
```

---

get\_factor\_score\_coefficients

*Return factor score coefficients*

---

**Description**

Return factor score coefficients

**Usage**

```
get_factor_score_coefficients(m, latent = TRUE, errors = FALSE, ...)
```

**Arguments**

m	Structural model represented by lavaan syntax or output of sim_standardized_matrices function.
latent	Include latent variables.
errors	Include observed error and latent disturbances variables.
...	parameters passed to the 'sim_standardized_matrices' function

**Value**

A matrix of factor score coefficients

**Examples**

```
m <- "
A =~ 0.5 * A1 + 0.8 * A2 + 0.8 * A3
B =~ 0.5 * B1 + 0.8 * B2 + 0.8 * B3
B ~ 0.5 * A
"
get_factor_score_coefficients(m)
```

---

```
get_factor_score_validity
```

*Return factor score validity coefficients*

---

### Description

Return factor score validity coefficients

### Usage

```
get_factor_score_validity(m, latent = TRUE, errors = FALSE, ...)
```

### Arguments

m	Structural model represented by lavaan syntax or output of sim_standardized_matrices function.
latent	Include latent variables.
errors	Include observed error and latent disturbances variables.
...	parameters passed to the 'sim_standardized_matrices' function

### Value

A matrix of validity coefficients

### Examples

```
m <- "
A =~ 0.5 * A1 + 0.8 * A2 + 0.8 * A3
B =~ 0.5 * B1 + 0.8 * B2 + 0.8 * B3
B ~ 0.5 * A
"
get_factor_score_validity(m)
```

---

```
get_factor_score_validity_se
```

*Return factor score validity coefficient standard errors*

---

### Description

Return factor score validity coefficient standard errors

### Usage

```
get_factor_score_validity_se(m, latent = TRUE, errors = FALSE, ...)
```

**Arguments**

m	Structural model represented by lavaan syntax or output of sim_standardized_matrices function.
latent	Include latent variables.
errors	Include observed error and latent disturbances variables.
...	parameters passed to the 'sim_standardized_matrices' function

**Value**

A matrix of factor score standard errors

**Examples**

```
m <- "  
A =~ 0.5 * A1 + 0.8 * A2 + 0.8 * A3  
B =~ 0.5 * B1 + 0.8 * B2 + 0.8 * B3  
B ~ 0.5 * A  
"  
get_factor_score_validity_se(m)
```

---

get\_model\_implied\_correlations

*Return model-implied correlation matrix*

---

**Description**

Function that takes a lavaan model with standardized parameters and returns a model-implied correlation matrix

**Usage**

```
get_model_implied_correlations(  
  m,  
  observed = TRUE,  
  latent = FALSE,  
  errors = FALSE,  
  factor_scores = FALSE,  
  composites = FALSE,  
  ...  
)
```

**Arguments**

m	Structural model represented by lavaan syntax or output of sim_standardized_matrices function.
observed	Include observed variables
latent	Include latent variables
errors	Include observed error and latent disturbances variables
factor_scores	Include factor score variables
composites	Include composite variables
...	parameters passed to the 'sim_standardized_matrices' function

**Value**

A correlation matrix

**Examples**

```
library(simstandard)
# lavaan model
m = "Latent_1 =~ 0.8 * Ob_1 + 0.7 * Ob_2 + 0.4 * Ob_3"

get_model_implied_correlations(m)
```

---

get_model_names	<i>Return model names</i>
-----------------	---------------------------

---

**Description**

Return model names

**Usage**

```
get_model_names(m, ...)
```

**Arguments**

m	Structural model represented by lavaan syntax or output of sim_standardized_matrices function.
...	parameters passed to the 'sim_standardized_matrices' function

**Value**

A list of variable names



**Examples**

```
m <- "  
A =~ 0.5 * A1 + 0.8 * A2 + 0.8 * A3  
B =~ 0.5 * B1 + 0.8 * B2 + 0.8 * B3  
B ~ 0.5 * A  
"  
get_model_names(m)
```

---

**lav2ram***Extract standardized RAM matrices from a lavaan object*

---

**Description**

Extract standardized RAM matrices from a lavaan object

**Usage**

```
lav2ram(fit)
```

**Arguments**

**fit**                    An object of class lavaan

**Value**

list of RAM matrices A (asymmetric paths), S (symmetric paths), and F (filter matrix)

---

**matrix2lavaan***Create lavaan model syntax from matrix coefficients*

---

**Description**

Create lavaan model syntax from matrix coefficients

**Usage**

```
matrix2lavaan(  
  measurement_model = NULL,  
  structural_model = NULL,  
  covariances = NULL  
)
```

**Arguments**

- `measurement_model`  
A matrix or data.frame with measurement model loadings. Column names are latent variables. Row names or the first column of a data.frame are indicator variables.
- `structural_model`  
A matrix or data.frame with structural model coefficients (i.e., regressions). Column names are "causal" variables. Row names or the first column of a data.frame are "effect" variables.
- `covariances`  
A matrix or data.frame with model covariances. Column names must match the row names. If a data.frame, row variable names can be specified in the first column.

**Value**

a character string with lavaan syntax

**Examples**

```
library(simstandard)
# Specifying the measurement model:
# For a data.frame, the column names are latent variables,
# and the indicators can be specified as rownames.
m <- data.frame(X = c(0.7,0.8,0,0),
                Y = c(0,0,0.8,0.9))
rownames(m) <- c("A", "B", "C", "D")
# Indicator variables can also be specified
# as the first column variable
# with subsequent column names as latent variables
m <- data.frame(Indicators = c("A", "B", "C", "D"),
                X = c(0.7,0.8,0,0),
                Y = c(0,0,0.8,0.9))
# Alternately, a matrix can be used:
m <- matrix(c(0.7,0.8,0,0,
              0,0,0.8,0.9),
            ncol = 2,
            dimnames = list(c("A", "B", "C", "D"),
                           c("X", "Y")))

# Specifying the structural coefficients:
# The regression coefficients of the structural model can be
# specified as either a data.frame or a matrix. Column names
# are the predictors and row names are the criterion variables.
# With a data.frame, criterion variables can alternately be
# specified with as the first column.
s <- matrix(0.5, nrow = 1, ncol = 1, dimnames = list("Y", "X"))
# The covariance matrix must be symmetric. Can also be specified
# as a data. frame.
Sigma <- matrix(c(1, 0.3,
                  0.3, 1),
                nrow = 2,
                ncol = 2,
```

```

      dimnames = list(c("B", "C"),
                     c("B", "C")) )
model <- matrix2lavaan(measurement_model = m,
                      structural_model = s,
                      covariances = Sigma)

cat(model)

```

---

model_complete	<i>Function that takes a lavaan model with standardized paths and loadings and returns a complete lavaan model syntax with standardized variances</i>
----------------	---

---

## Description

Function that takes a lavaan model with standardized paths and loadings and returns a complete lavaan model syntax with standardized variances

## Usage

```
model_complete(m)
```

## Arguments

m                      Structural model represented by lavaan syntax

## Value

character string representing lavaan model

## Examples

```

library(simstandard)
# lavaan model
m = "
Latent_1 =~ 0.9 * Ob_11 + 0.8 * Ob_12 + 0.7 * Ob_13
Latent_2 =~ 0.9 * Ob_21 + 0.6 * Ob_22 + 0.4 * Ob_23
Latent_2 ~ 0.6 * Latent_1
"
# Same lavaan syntax, but with standardized variances
m_complete <- model_complete(m)
cat(m_complete)

```

---

sim\_standardized      *Generates simulated data with standardized parameters.*

---

### Description

This function takes a lavaan model with standardized parameters and simulates latent scores, errors, disturbances, and observed scores.

### Usage

```
sim_standardized(
  m,
  n = 1000,
  observed = TRUE,
  latent = TRUE,
  errors = TRUE,
  factor_scores = FALSE,
  composites = FALSE,
  matrices = FALSE,
  ...
)
```

### Arguments

m	Structural model represented by lavaan syntax
n	Number of simulated cases
observed	Include observed variables
latent	Include latent variables
errors	Include observed error and latent disturbances variables
factor_scores	Include factor score variables
composites	Include composite variables
matrices	Include matrices as attribute of tibble
...	Arguments passed to 'simstandardized_matrices'

### Details

This function supports the '~' operator for regressions, the '~~' for covariances (but not variances), and the '=~' latent variable loadings. It does not support intercepts (e.g., 'y ~ 1'), thresholds, scaling factors, formative factors, or equality constraints.

### Value

tibble with standardized data

## Examples

```
library(simstandard)
# Lavaan model
m = "Latent_1 =~ 0.8 * Ob_1 + 0.7 * Ob_2 + 0.4 * Ob_3"

# simulate 10 cases
sim_standardized(m, n = 10)
```

---

sim\_standardized\_matrices

*Return model characteristics*

---

## Description

Function that takes a lavaan model with standardized parameters and returns a list with model characteristics

## Usage

```
sim_standardized_matrices(m, max_iterations = 100, composite_threshold = NULL)
```

## Arguments

**m** Structural model represented by lavaan syntax

**max\_iterations** Maximum number of iterations before the algorithm fails

**composite\_threshold** Loadings with absolute values less than this threshold will not be counted as composite indicators

## Details

This function supports the ‘~’ operator for regressions, the ‘~~’ for covariances (but not variances), and the ‘=~’ latent variable loadings. It does not support intercepts (e.g., ‘y ~ 1’), thresholds, scaling factors, formative factors, or equality constraints.

## Value

list of path and covariance coefficients

## Examples

```
library(simstandard)
# lavaan model
m = "Latent_1 =~ 0.8 * Ob_1 + 0.7 * Ob_2 + 0.4 * Ob_3"

sim_standardized_matrices(m)
```

# Index

[add\\_composite\\_scores](#), 2  
[add\\_factor\\_scores](#), 3  
  
[fixed2free](#), 4  
  
[get\\_factor\\_score\\_coefficients](#), 5  
[get\\_factor\\_score\\_validity](#), 6  
[get\\_factor\\_score\\_validity\\_se](#), 6  
[get\\_model\\_implied\\_correlations](#), 7  
[get\\_model\\_names](#), 8  
  
[lav2ram](#), 9  
  
[matrix2lavaan](#), 9  
[model\\_complete](#), 11  
  
[sim\\_standardized](#), 12  
[sim\\_standardized\\_matrices](#), 13