

# Package ‘semboottools’

March 19, 2026

**Title** Bootstrapping Helpers for Structural Equation Modelling

**Version** 0.1.2

**Description** A collection of helper functions for forming bootstrapping confidence intervals and examining bootstrap estimates in structural equation modelling, introduced in Yang and Cheung (2026) <[doi:10.3758/s13428-025-02911-z](https://doi.org/10.3758/s13428-025-02911-z)> The function currently support models fitted by the 'lavaan' package by Rosseel (2012) <[doi:10.18637/jss.v048.i02](https://doi.org/10.18637/jss.v048.i02)>.

**License** GPL (>= 3)

**Encoding** UTF-8

**RoxygenNote** 7.3.3

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

**Config/testthat/edition** 3

**Config/testthat/parallel** true

**VignetteBuilder** knitr

**Imports** boot, lavaan, psych, lavaan.printer, ggplot2, GGally, rlang, patchwork

**URL** <https://Yangzhen1999.github.io/semboottools/>

**BugReports** <https://github.com/Yangzhen1999/semboottools/issues>

**NeedsCompilation** no

**Author** Wendie Yang [aut, cre] (ORCID: <<https://orcid.org/0009-0000-8388-6481>>),  
Shu Fai Cheung [aut] (ORCID: <<https://orcid.org/0000-0002-9871-9448>>)

**Maintainer** Wendie Yang <1581075494q@gmail.com>

**Repository** CRAN

**Date/Publication** 2026-03-19 08:20:02 UTC

## Contents

gg_hist_qq_boot . . . . .	2
gg_scatter_boot . . . . .	5
hist_qq_boot . . . . .	7
jab_after_boot . . . . .	11
parameterEstimates_boot . . . . .	13
print.sbt_std_boot . . . . .	15
print.sbt_ustd_boot . . . . .	17
standardizedSolution_boot . . . . .	19
store_boot . . . . .	21

<b>Index</b>	<b>25</b>
--------------	-----------

---

gg_hist_qq_boot	<i>Diagnostic Plots of Bootstrap Estimates (ggplot2, Modular)</i>
-----------------	---

---

## Description

Produce diagnostic plots of bootstrap estimates using **ggplot2**. Compared to `hist_qq_boot()` (base graphics), this function is *modular* and *modern*: optional layers (histogram, mean line, CI, SD arrow, QQ), and the ability to return only the data or the ggplot object for further customization.

## Usage

```
gg_hist_qq_boot(
  object,
  param,
  standardized = NULL,
  bins = 35,
  show_ci = TRUE,
  show_mean = TRUE,
  show_sd_arrow = TRUE,
  show_qq = TRUE,
  show_hist = TRUE,
  text_size = 3,
  ci_label_digits = 3,
  mean_label_digits = 3,
  sd_label_digits = 3,
  sd_clip_to_density = TRUE,
  sd_arrow_height = 0.58,
  trim_quantiles = NULL,
  ci_line_color = "#D62728",
  dens_line_color = "#8B0000CC",
  bar_fill = "#5DADE233",
  bar_color = "#1B4F72",
  enforce_serif = TRUE,
  theme_override = NULL,
```

```

dens_adjust = 1,
dens_from = NULL,
dens_to = NULL,
show_boot_mean = TRUE,
point_color = "#000000",
boot_mean_color = "#AA3377",
output = c("draw", "ggplot", "data"),
return = NULL,
...
)

```

## Arguments

object	A lavaan object with stored bootstrap results, or an sbt_std_boot (from standardizedSolution_boot() or sbt_ustd_boot (from parameterEstimates_boot()) object.
param	Character. Name of the parameter to plot (as in coef() or "lhs op rhs").
standardized	Logical. Use standardized estimates. Ignored for sbt_* inputs.
bins	Integer. Number of histogram bins when show_hist = TRUE. Default 35.
show_ci	Logical. Draw CI vertical dashed lines and labels. Default TRUE.
show_mean	Logical. Draw <b>point estimate</b> vertical line and label. Default TRUE.
show_sd_arrow	Logical. Draw $\pm$ SD double-headed arrow and label. Default TRUE.
show_qq	Logical. Add a side-by-side QQ plot (uses patchwork if available). Default TRUE.
show_hist	Logical. Draw histogram (otherwise density only). Default TRUE.
text_size	Numeric. Font size for labels. Default 3.
ci_label_digits, mean_label_digits, sd_label_digits	Integer digits for labels. Default 3.
sd_clip_to_density	Logical. Clip SD arrow within density curve span. Default TRUE.
sd_arrow_height	Numeric in (0,1]. Relative height for the SD arrow. Default 0.58.
trim_quantiles	Length-2 numeric. Trim x-axis by quantiles (e.g., c(.01, .99)). NULL = no trimming.
ci_line_color	Color for CI lines. Default "#D62728".
dens_line_color	Color for density curve. Default "#8B0000CC".
bar_fill, bar_color	Histogram fill/border colors. Defaults "#5DADE233" / "#1B4F72".
enforce_serif	Logical. Use serif base family in the theme. Default TRUE.
theme_override	Optional ggplot2::theme() added on top of base theme.
dens_adjust	Numeric $\geq 0$ . Bandwidth adjust for stats::density(). Default 1.
dens_from, dens_to	Optional numeric. Force from/to range for density().

show_boot_mean	Logical. Draw <b>bootstrap mean</b> vertical line. Default TRUE.
point_color	Color for the <b>point estimate</b> line. Default "#000000".
boot_mean_color	Color for the <b>bootstrap mean</b> line. Default "#AA3377".
output	One of "draw", "ggplot", or "data". What to return/draw. Default "draw".
return	Deprecated. Backward-compatible alias of output; if supplied, it overrides output.
...	Additional arguments passed to ggplot layers (e.g., geom_histogram()).

### Details

- For free (unstandardized) parameters, bootstrap draws are extracted directly from the lavaan object;
- For user-defined parameters or standardized solutions, `store_boot()` or `standardizedSolution_boot()` must be called in advance;
- Supports objects of class `sbt_std_boot` (standardized) and `sbt_ustd_boot` (unstandardized).

### Return mode

- `output = "draw"`: draw the plot (default) and invisibly return the ggplot object;
- `output = "ggplot"`: return the ggplot object without drawing;
- `output = "data"`: return a list containing the data frames for histogram, density, and CI positions (class "sbt\_boot\_plotdata").

### Axis control

- By default, the x-axis has a small margin beyond the observed range; set `trim_quantiles = c(.01, .99)` to crop extreme tails by quantiles.

### Optional layers

- `show_hist`, `show_mean`, `show_ci`, `show_sd_arrow`, `show_qq`.

### Value

- If `output = "draw"`: draws the plot and invisibly returns the ggplot object;
- If `output = "ggplot"`: returns a ggplot object (no drawing);
- If `output = "data"`: returns a list with
  - `t`: bootstrap draws;
  - `t0`: point estimate;
  - `sd`: sample standard deviation;
  - `hist`: data frame for histogram layer;
  - `dens`: data frame for density curve (x, y);
  - `ci`: list with lower, upper, `y_lower`, `y_upper`.

If `output = "draw"`, draws the plot and (invisibly) returns the ggplot object; if `output = "ggplot"`, returns the ggplot (no drawing); if `output = "data"`, returns a list with `t`, `t0`, `sd`, `hist`, `dens`, and `ci` (class "sbt\_boot\_plotdata").

## References

Rousselet, G. A., Pernet, C. R., & Wilcox, R. R. (2021). The percentile bootstrap: A primer with step-by-step instructions in R. *Advances in Methods and Practices in Psychological Science*, 4(1), 1–10. doi:10.1177/2515245920911881

## See Also

[store\\_boot\(\)](#), [standardizedSolution\\_boot\(\)](#), [hist\\_qq\\_boot\(\)](#)

---

gg\_scatter\_boot

*Scatterplot Matrix of Bootstrap Estimates (ggplot2/GGally)*

---

## Description

Generates a scatterplot matrix of the bootstrap estimates for two or more model parameters using ggplot2 and GGally, with optional smoothing lines, confidence ellipses, and customizable density or histogram plots on the diagonal.

## Usage

```
gg_scatter_boot(  
  object,  
  params,  
  standardized = NULL,  
  title = "Bootstrap Estimates",  
  point_size = 1.8,  
  point_alpha = 0.35,  
  point_color = "#5DADE233",  
  show_smooth = TRUE,  
  smooth_method = "lm",  
  smooth_se = FALSE,  
  show_ellipse = TRUE,  
  ellipse_level = 0.95,  
  ellipse_color = "#8B0000CC",  
  diag_type = c("both", "density", "hist", "blank"),  
  hist_border_size = 0.2,  
  bins = 30,  
  dens_fill = "#5DADE233",  
  dens_color = "#8B0000CC",  
  hist_fill = "#5DADE233",  
  hist_color = "#1B4F72",  
  show_corr = TRUE,  
  corr_text_size = 7,  
  panel_border_color = "grey40",  
  panel_border_size = 2,  
  corr_digits = 2,  
  show_mean_diag = TRUE,
```

```

  mean_line_color = "black",
  mean_linewidth = 0.7,
  mean_linetype = "dashed",
  output = c("draw", "ggplot")
)

```

### Arguments

<code>object</code>	A bootstrap result object of class "sbt_std_boot" or "sbt_ustd_boot".
<code>params</code>	Character vector of parameter names to plot ( $\geq 2$ ).
<code>standardized</code>	Logical; whether to use standardized estimates.
<code>title</code>	Plot title.
<code>point_size, point_alpha, point_color</code>	Aesthetics for points in lower panels.
<code>show_smooth, smooth_method, smooth_se</code>	Control the regression smoother in lower panels.
<code>show_ellipse, ellipse_level, ellipse_color</code>	Confidence ellipse options in lower panels.
<code>diag_type</code>	One of "both", "density", "hist", "blank" for diagonal panels.
<code>hist_border_size</code>	Histogram outline width in diagonal "hist"/"both".
<code>bins, dens_fill, dens_color, hist_fill, hist_color</code>	Diagonal panel styles.
<code>show_corr</code>	Logical; if TRUE, show correlation coefficients in upper panels.
<code>corr_text_size, corr_digits</code>	Text size and digits for upper-panel correlation.
<code>panel_border_color, panel_border_size</code>	Panel border color and width for all panels.
<code>show_mean_diag, mean_line_color, mean_linewidth, mean_linetype</code>	Draw a vertical mean line in diagonal panels.
<code>output</code>	Either "draw" (print) or "ggplot" (return ggplot object).

### Details

The function `gg_scatter_boot()` is an enhanced ggplot2-based version of `scatter_boot()`, designed to produce publication-quality scatterplot matrices of bootstrap estimates. It can be applied to the output of `standardizedSolution_boot()` or `parameterEstimates_boot()`.

### Value

Invisibly returns the ggplot object (or prints it if `output="draw"`).

### See Also

[scatter\\_boot\(\)](#), [hist\\_qq\\_boot\(\)](#), [standardizedSolution\\_boot\(\)](#), [parameterEstimates\\_boot\(\)](#)

**Description**

Plots for examining the distribution of bootstrap estimates in a model fitted by lavaan.

**Usage**

```
hist_qq_boot(
  object,
  param,
  standardized = NULL,
  nclass = NULL,
  hist_color = "#5DADE233",
  hist_linewidth = 1.5,
  hist_border_color = "#1B4F72",
  density_line_type = "solid",
  density_line_color = "#8B0000CC",
  density_line_linewidth = 2,
  est_line_color = "#154360",
  est_line_type = "dashed",
  est_line_linewidth = 2,
  qq_dot_pch = 21,
  qq_dot_color = "#1B4F72",
  qq_dot_fill = "#5DADE233",
  qq_dot_size = 1.3,
  qq_line_color = "#8B0000CC",
  qq_line_linewidth = 2.1,
  qq_line_linetype = "solid"
)

scatter_boot(
  object,
  params,
  standardized = NULL,
  main = "Bootstrap Estimates",
  ...
)
```

**Arguments**

**object** Either a lavaan-class object with bootstrap estimates stored, or the output of [standardizedSolution\\_boot\(\)](#). For standardized solution and user-defined parameters, if the object is a 'lavaan-class' object, the estimates need to be stored by [store\\_boot\(\)](#).

param	String. The name of the parameter to be plotted, which should be the name as appeared in a call to <code>coef()</code> .
standardized	Logical. Whether the estimates from the standardized solution are to be plotted. Default is NULL. If object is a lavaan object, then this is a required parameter and users need to explicitly set it to TRUE or FALSE. If object is the output of <code>standardizedSolution_boot()</code> , then this argument is ignored ( forced to be TRUE internally).
nclass	The number of breaks. This argument will be passed to <code>hist()</code> . Default is NULL.
hist_color	String. The color of the bars in the histogram. It will be passed to <code>hist()</code> for the argument <code>col</code> . Default is light blue ( <code>scales::alpha("#5DADE2", 0.2)</code> ).
hist_linewidth	The width of the borders of the bars in the histogram. Default is 1.5.
hist_border_color	String. The color of the borders (outline) of the bars in the histogram. It will be passed to <code>hist()</code> for the argument <code>border</code> . Default is a dark blue color ( <code>"#1B4F72"</code> ).
density_line_type	String. The type of the line of the density curve in the histogram. It will be passed to <code>lines()</code> for the argument <code>lty</code> . Default is "solid".
density_line_color	String. The color of the density curve in the histogram. It will be passed to <code>lines()</code> for the argument <code>col</code> . Default is "blue".
density_line_linewidth	The width of the density curve in the histogram. It will be passed to <code>lines()</code> for the argument <code>lwd</code> . Default is 2.
est_line_color	String. The color of the vertical line showing the point estimate in the histogram. It will be passed to <code>abline()</code> for the argument <code>col</code> .
est_line_type	String. The type of the vertical line in the histogram showing the point estimate of the parameter. It will be passed to <code>abline()</code> for the argument <code>lty</code> . Default is "dashed",
est_line_linewidth	The width of the vertical line showing the point estimate in the histogram. It will be passed to <code>hist()</code> for the argument <code>lwd</code> . Default is 2.
qq_dot_pch	Numeric. The shape of the points in the normal QQ-plot. It will be passed to <code>qqnorm()</code> for the argument <code>pch</code> . Default is 21.
qq_dot_color	String. The color of the points in the normal QQ-plot. It will be passed to <code>qqnorm()</code> for the argument <code>col</code> .
qq_dot_fill	String. The fill color of the points in the normal QQ-plot. Only applicable when <code>qq_dot_pch</code> is set to a symbol that allows fill color (e.g., <code>pch = 21</code> ). It will be passed to <code>qqnorm()</code> for the argument <code>bg</code> . Default is a semi-transparent light blue ( <code>scales::alpha("#5DADE2", 0.2)</code> ).
qq_dot_size	The size of the points in the normal QQ-plot. It will be passed to <code>qqnorm()</code> for the argument <code>cex</code> . Default is 2.
qq_line_color	String. The color of the diagonal line to be drawn in the normal QQ-plot. It will be passed to <code>qqline()</code> for the argument <code>col</code> .

qq_line_linewidth	The width of the diagonal line to be drawn in the normal QQ-plot. It will be passed to <code>qqline()</code> for the argument <code>lwd</code> . Default is 2.1.
qq_line_linetype	The type of the diagonal line to be drawn in the normal QQ-plot. Default is "solid".
params	The vector of the names of the parameters to be plotted, which should be the names as appeared in a call to <code>coef()</code> . The function <code>scatter_boot()</code> requires two or more parameters selected by this argument.
main	The title of the scatterplot matrix. Default is "Bootstrap Estimates".
...	Arguments to be passed to <code>psych::pairs.panels()</code> . Please refer to the help page of <code>psych::pairs.panels()</code> for arguments to customize the plot.

### Details

Rousselet, Pernet, and Wilcox (2021) argued that when using bootstrapping, it is necessary to examine the distribution of bootstrap estimates. This can be done when `boot::boot()` is used because it has a plot method for its output. This cannot be easily done in model fitted by `lavaan::lavaan()`, such as `lavaan::sem()` and `lavaan::cfa()`.

The function `hist_qq_boot()` is used for plotting the distribution of bootstrap estimates for a model fitted by `lavaan` in a format similar to that of the output of `boot::boot()`, with a histogram on the left and a normal QQ-plot on the right.

For free parameters in a model (unstandardized), it can be called directly on the output of `lavaan` and retrieves the stored estimates.

For estimates of user-defined parameters, call `store_boot()` first to compute and store the bootstrap estimates first.

For estimates in standardized solution, for both free and user-defined parameters, call `store_boot()` first to compute and store the bootstrap estimates in the standardized solution.

It can also plot bootstrap estimates in the output of `standardizedSolution_boot()` or `parameterEstimates_boot()`.

The function `scatter_boot()` is used to generate a scatterplot matrix of the bootstrap estimates of two or more parameters. The function `psych::pairs.panels()` from the package `psych` is used.

Like `hist_qq_boot()`, it can also be used on the output of `standardizedSolution_boot()` or `parameterEstimates_boot()`.

### Value

Return the original `lavaan::lavaan` object invisibly. Called for its side-effect (plotting the graphs).

### Author(s)

Shu Fai Cheung <https://orcid.org/0000-0002-9871-9448>

### References

Rousselet, G. A., Pernet, C. R., & Wilcox, R. R. (2021). The percentile bootstrap: A primer with step-by-step instructions in R. *Advances in Methods and Practices in Psychological Science*, 4(1), 1–10. doi:10.1177/2515245920911881

**See Also**

[store\\_boot\(\)](#) and [standardizedSolution\\_boot\(\)](#).

**Examples**

```
library(lavaan)

set.seed(5478374)
n <- 50
x <- runif(n) - .5
m <- .40 * x + rnorm(n, 0, sqrt(1 - .40))
y <- .30 * m + rnorm(n, 0, sqrt(1 - .30))
dat <- data.frame(x = x, y = y, m = m)

mod <-
"
m ~ a * x
y ~ b * m + x
ab := a * b
"
fit <- sem(mod,
           data = dat,
           se = "bootstrap",
           bootstrap = 50,
           iseed = 985714)

# Can plot bootstrap estimates for
# free parameters directly
# Note that 'standardized' must be always be set to
# either TRUE or FALSE. No default value.
hist_qq_boot(fit, "a", standardized = FALSE)

# For estimates of user-defined parameters,
# call store_boot() first.
fit <- store_boot(fit)
hist_qq_boot(fit, "ab", standardized = FALSE)

# For estimates in standardized solution,
# call store_boot() first.
fit <- store_boot(fit)
hist_qq_boot(fit, "a", standardized = TRUE)
hist_qq_boot(fit, "ab", standardized = TRUE)

# It can also plot the estimates stored
# in the output of standardizedSolution_boot().
std_boot <- standardizedSolution_boot(fit)
hist_qq_boot(std_boot, "ab")
hist_qq_boot(fit, "ab", standardized = TRUE)

# Scatterplot matrix of bootstrap estimates for
```

```

# two or more free parameters
scatter_boot(fit, c("a", "b", "ab"), standardized = FALSE)

# Can include user-defined parameters in
# scatterplot matrix, if their bootstrap
# estimates have been stored
scatter_boot(fit, c("ab", "a", "b"), standardized = FALSE)

# scatter_boot also supports the
# standardized solution
scatter_boot(fit, c("a", "b", "ab"), standardized = TRUE)

```

---

jab_after_boot	<i>Jackknife-after-Bootstrap (JAB) influence for lavaan bootstrap results</i>
----------------	---

---

## Description

Compute Jackknife-after-Bootstrap (JAB) influence values for a single model parameter and diagnose observation-level influence by comparing the full bootstrap distribution with leave-one-out (LOO) subdistributions obtained via `boot.idx` (stored by `store_boot(keep.idx = TRUE)`).

The ALL-summary (mean/SE/CI) is taken directly and consistently from `standardizedSolution_boot()` (standardized case) or `parameterEstimates_boot()` (unstandardized case). JAB influence is defined as  $I_j = n\{\bar{\theta}_{(-j)} - \bar{\theta}\}$ , where  $\bar{\theta}$  is the bootstrap mean over all replicates, and  $\bar{\theta}_{(-j)}$  is the bootstrap mean over the replicates that exclude observation  $j$ .

## Usage

```

jab_after_boot(
  fit,
  param,
  standardized = TRUE,
  top_k = 5L,
  ci_level = 0.95,
  min_keep = NULL,
  plot = FALSE,
  plot_engine = c("ggplot2", "base"),
  ylab_override = NULL,
  verbose = TRUE,
  font_family = "sans"
)

```

## Arguments

<code>fit</code>	A lavaan object for which <code>store_boot(keep.idx = TRUE)</code> has been called. Must contain <code>fit@external\$sbt_boot_ustd</code> (with <code>boot.idx</code> attribute) and typically <code>sbt_boot_std</code> .
------------------	--

param	Character(1). Target parameter. Accepts "lhs op rhs", a " := " label, or a free-parameter label.
standardized	Logical. If TRUE, use standardized bootstrap; else unstandardized.
top_k	Integer. Number of top cases (by  JAB_value ) to report.
ci_level	Numeric (0,1). Percentile CI level for LOO subdistributions.
min_keep	Integer. Minimum number of bootstrap replicates kept in each LOO subset; default $\max(30, \text{floor}(0.2 * B))$ .
plot	Logical. If TRUE, draw a JAB diagnostic plot.
plot_engine	Character. Plot engine for plot = TRUE. One of "ggplot2" or "base". Default "ggplot2" if available, otherwise "base".
ylab_override	Optional character. Override the default y-axis label in the plot.
verbose	Logical. If TRUE, print compact summaries.
font_family	Character. Graphics font family (e.g., "serif", "sans", "Times New Roman").

### Details

Parameter identification is robust: it accepts "lhs op rhs", " := " user-defined name, and *label*. For unstandardized cases, if *label* is given and the column is missing in `attr(unstd_boot, "boot_est_ustd")`, the function will fall back to the raw matrix `fit@external$bt_boot_ustd` if a column with that label exists (e.g., "b"), keeping SE/CI from the `parameterEstimates_boot()` row if available.

### Value

A list with:

param The target parameter string.

standardized Logical flag as input.

full\_summary Data frame with ALL distribution summary: scope="ALL", param, mean, SE, CI.Lo, CI.Up.

cases\_summary Data frame (top top\_k) with columns: case, JAB\_value, mean, SE, CI.Lo, CI.Up.

F The occurrence matrix  $B \times n$ .

tstar Full bootstrap vector for param.

### See Also

[store\\_boot](#), [standardizedSolution\\_boot](#), [parameterEstimates\\_boot](#)

---

 parameterEstimates\_boot

*Bootstrap CIs for Parameter Estimates*


---

## Description

Functions for forming bootstrap confidence intervals for the parameter estimates.

## Usage

```
parameterEstimates_boot(
  object,
  level = 0.95,
  boot_org_ratio = FALSE,
  boot_ci_type = c("perc", "bc", "bca.simple"),
  save_boot_est = TRUE,
  boot_pvalue = TRUE,
  boot_pvalue_min_size = 1000,
  standardized = FALSE,
  ...
)
```

## Arguments

object	A 'lavaan'-class object, fitted with 'se = "boot"'.
level	The level of confidence of the confidence intervals. Default is .95.
boot_org_ratio	The ratio of (a) the distance of the bootstrap confidence limit from the point estimate to (b) the distance of the original confidence limit in object from the point estimate. Default is FALSE.
boot_ci_type	The type of the bootstrapping confidence intervals. Support percentile confidence intervals ("perc", the default) and bias-corrected confidence intervals ("bc" or "bca.simple").
save_boot_est	Whether the bootstrap estimates of the parameter estimates are saved. If saved, the bootstrap estimates of the free parameters will be stored in the attribute boot_est_ustd, while the bootstrap estimates of user-defined parameters, if any, will be stored in the attribute boot_def. Default is TRUE.
boot_pvalue	Whether asymmetric bootstrap $p$ -values are computed. Default is TRUE.
boot_pvalue_min_size	Integer. The asymmetric bootstrap $p$ -values will be computed only if the number of valid bootstrap estimates is at least this value. Otherwise, NA will be returned. If the number of valid bootstrap samples is less than this value, then boot_pvalue will be set to FALSE.
standardized	The type of standardized estimates. The same argument of <code>lavaan::parameterEstimates()</code> , and support all values supported by <code>lavaan::parameterEstimates()</code> . It is recommended to use <code>standardizedSolution_boot()</code> or <code>lavaan::standardizedSolution()</code>

because this function only report the point estimates of the standardized solution, without standard errors or confidence intervals.

... Other arguments to be passed to `lavaan::parameterEstimates()`.

## Details

`parameterEstimates_boot()` receives a `lavaan::lavaan` object and form bootstrap confidence intervals for the parameter estimates.

The function `store_boot()` should be called first to compute and store bootstrap estimates. This function will then retrieve them.

### Bootstrap Confidence Intervals:

It supports percentile and bias-corrected bootstrap confidence intervals.

### Bootstrap Standard Errors:

The standard errors are the standard deviation of the bootstrap estimates.

### Bootstrap Asymmetric $p$ -Values:

If percentile bootstrap confidence interval is requested, asymmetric bootstrap  $p$ -values are also computed, using the method presented in Asparouhov and Muthén (2021).

## Value

The output of `lavaan::parameterEstimates()`, with bootstrap confidence intervals appended to the right, with class set to `sbt_ustd_boot`. It has a print method (`print.sbt_ustd_boot()`) that can be used to print the parameter estimates in a format similar to that of the printout of the `summary()` of a `lavaan::lavaan` object.

## Author(s)

Shu Fai Cheung <https://orcid.org/0000-0002-9871-9448>.

## References

Asparouhov, A., & Muthén, B. (2021). Bootstrap  $p$ -value computation. Retrieved from <https://www.statmodel.com/download/Bootstrap%20-%20Pvalue.pdf>

## See Also

`lavaan::parameterEstimates()`, `store_boot()`

## Examples

```
library(lavaan)
set.seed(5478374)
n <- 50
x <- runif(n) - .5
m <- .40 * x + rnorm(n, 0, sqrt(1 - .40))
y <- .30 * m + rnorm(n, 0, sqrt(1 - .30))
dat <- data.frame(x = x, y = y, m = m)
```

```

model <-
'
m ~ a*x
y ~ b*m
ab := a*b
'

# Should set bootstrap to at least 2000 in real studies
fit <- sem(model, data = dat, fixed.x = FALSE)
summary(fit)
fit <- store_boot(fit,
                  do_bootstrapping = TRUE,
                  R = 100,
                  iseed = 1234)
est <- parameterEstimates_boot(fit)
est

```

---

print.sbt\_std\_boot      *Print a 'sbt\_std\_boot' Object*

---

## Description

Print method for a 'sbt\_std\_boot' object, which is the output of [standardizedSolution\\_boot\(\)](#).

## Usage

```

## S3 method for class 'sbt_std_boot'
print(
  x,
  ...,
  nd = 3,
  output = c("lavaan.printer", "text", "table"),
  standardized_only = TRUE,
  boot_ci_only = FALSE,
  drop_cols = "Z"
)

```

## Arguments

x	Object of the class <code>sbt_std_boot</code> , the output of <a href="#">standardizedSolution_boot()</a> .
...	Optional arguments to be passed to <a href="#">print()</a> methods.
nd	The number of digits after the decimal place. Default is 3.
output	String. How the results are printed. If set to "table", the results are printed in a table format similar to that of <a href="#">lavaan::standardizedSolution()</a> . If set to "text", the results will be printed in a text format similar to the printout of the output of <a href="#">summary()</a> of a 'lavaan'-class object. If set to "lavaan.printer", the default, <code>lavaan.printer</code> will be used to print a more compact version of the "text" output.

standardized_only	Logical. If TRUE, the default, only the results for the standardized solution will be printed. If FALSE, then the standardized solution is printed alongside the unstandardized solution, as in the printout of the output of <code>summary()</code> of a 'lavaan'-class object.
boot_ci_only	Logical. Whether only bootstrap confidence intervals are printed. If FALSE, the default, the delta method confidence intervals by <code>lavaan::standardizedSolution()</code> are also printed.
drop_cols	The name(s) of the column(s) to drop if output format is "lavaan.printer". Default is "Z", to fit the print out to the usual screen width of 80.

### Details

The default format of the printout, "lavaan.printer", is a compact version of the lavaan-style printout, generated by `lavaan.printer`. Alternatively, users can request a format similar to that of the printout of the summary of a lavaan output by setting `output` to "text". This format can be used if "lavaan.printer" failed.

Users can also print the content just as a data frame by setting `output` to "table". Not easy to read much more compact.

For the "text" or "lavaan.printer" format, users can also select whether only the standardized solution is printed (the default) or whether the standardized solution is appended to the right of the printout.

### Value

`x` is returned invisibly. Called for its side effect.

### Author(s)

Shu Fai Cheung <https://orcid.org/0000-0002-9871-9448>

### See Also

`standardizedSolution_boot()`

### Examples

```
library(lavaan)
set.seed(5478374)
n <- 50
x <- runif(n) - .5
m <- .40 * x + rnorm(n, 0, sqrt(1 - .40))
y <- .30 * m + rnorm(n, 0, sqrt(1 - .30))
dat <- data.frame(x = x, y = y, m = m)
model <-
'
m ~ a*x
y ~ b*m
ab := a*b
'
```

```
# Should set bootstrap to at least 2000 in real studies
fit <- sem(model, data = dat, fixed.x = FALSE,
           se = "boot",
           bootstrap = 50)
std_out <- standardizedSolution_boot(fit)
std_out
print(std_out, standardized_only = FALSE)
```

---

```
print.sbt_ustd_boot   Print a 'sbt_ustd_boot' Object
```

---

## Description

Print method for a 'sbt\_ustd\_boot' object, which is the output of [parameterEstimates\\_boot\(\)](#).

## Usage

```
## S3 method for class 'sbt_ustd_boot'
print(
  x,
  ...,
  nd = 3,
  output = c("lavaan.printer", "text", "table"),
  drop_cols = "Z"
)
```

## Arguments

x	Object of the class <code>sbt_ustd_boot</code> , the output of <a href="#">parameterEstimates_boot()</a> .
...	Optional arguments to be passed to <a href="#">print()</a> methods.
nd	The number of digits after the decimal place. Default is 3.
output	String. How the results are printed. If set to "table", the results are printed in a table format similar to that of <a href="#">lavaan::parameterEstimates()</a> . If set to "text", the results will be printed in a text format similar to the printout of the output of <a href="#">summary()</a> of a 'lavaan'-class object. If set to "lavaan.printer", the default, <code>lavaan.printer</code> will be used to print a more compact version of the "text" output.
drop_cols	The name(s) of the column(s) to drop if output format is "lavaan.printer". Default is "Z", to fit the print out to the usual screen width of 80.

## Details

The default format of the printout, "lavaan.printer", is a compact version of the lavaan-style printout, generated by lavaan.printer. Alternatively, users can request a format similar to that of the printout of the summary of a lavaan output by setting output to "text". This format can be used if "lavaan.printer" failed.

Users can also print the content just as a data frame by setting output to "table". Not easy to read much more compact.

## Value

x is returned invisibly. Called for its side effect.

## Author(s)

Shu Fai Cheung <https://orcid.org/0000-0002-9871-9448>

## See Also

[parameterEstimates\\_boot\(\)](#)

## Examples

```
library(lavaan)
set.seed(5478374)
n <- 50
x <- runif(n) - .5
m <- .40 * x + rnorm(n, 0, sqrt(1 - .40))
y <- .30 * m + rnorm(n, 0, sqrt(1 - .30))
dat <- data.frame(x = x, y = y, m = m)
model <-
'
m ~ a*x
y ~ b*m
ab := a*b
'

# Should set bootstrap to at least 2000 in real studies
fit <- sem(model, data = dat, fixed.x = FALSE)

fit <- store_boot(fit,
                 do_bootstrapping = TRUE,
                 R = 100,
                 iseed = 1234)

est <- parameterEstimates_boot(fit)
est
```

---

 standardizedSolution\_boot

*Bootstrap CIs for Standardized Solution*


---

## Description

Functions for forming bootstrap confidence intervals for the standardized solution.

## Usage

```
standardizedSolution_boot(
  object,
  level = 0.95,
  type = "std.all",
  boot_delta_ratio = FALSE,
  boot_ci_type = c("perc", "bc", "bca.simple"),
  save_boot_est_std = TRUE,
  boot_pvalue = TRUE,
  boot_pvalue_min_size = 1000,
  ...
)
```

## Arguments

object	A 'lavaan'-class object, fitted with 'se = "boot"'.
level	The level of confidence of the confidence intervals. Default is .95.
type	The type of standard estimates. The same argument of <code>lavaan::standardizedSolution()</code> , and support all values supported by <code>lavaan::standardizedSolution()</code> . Default is "std.all".
boot_delta_ratio	The ratio of (a) the distance of the bootstrap confidence limit from the point estimate to (b) the distance of the delta-method limit from the point estimate. Default is FALSE.
boot_ci_type	The type of the bootstrapping confidence intervals. Support percentile confidence intervals ("perc", the default) and bias-corrected confidence intervals ("bc" or "bca.simple").
save_boot_est_std	Whether the bootstrap estimates of the standardized solution are saved. If saved, they will be stored in the attribute <code>boot_est_std</code> . Default is TRUE.
boot_pvalue	Whether asymmetric bootstrap $p$ -values are computed. Default is TRUE.
boot_pvalue_min_size	Integer. The asymmetric bootstrap $p$ -values will be computed only if the number of valid bootstrap estimates is at least this value. Otherwise, NA will be returned. If the number of valid bootstrap samples is less than this value, then <code>boot_pvalue</code> will be set to FALSE.
...	Other arguments to be passed to <code>lavaan::standardizedSolution()</code> .

## Details

`standardizedSolution_boot()` receives a `lavaan::lavaan` object fitted with bootstrapping standard errors requested and forms the confidence intervals for the standardized solution.

It works by calling `lavaan::standardizedSolution()` with the bootstrap estimates of free parameters in each bootstrap sample to compute the standardized estimates in each sample.

Alternative, call `store_boot()` to compute and store bootstrap estimates of the standardized solution. This function will then retrieve them, even if `se` was not set to "boot" or "bootstrap" when fitting the model.

### Bootstrap Confidence Intervals:

It supports percentile and bias-corrected bootstrap confidence intervals.

### Bootstrap Standard Errors:

The standard errors are the standard deviation of the bootstrap estimates, which can be different from the delta-method standard errors.

### Bootstrap Asymmetric *p*-Values:

If percentile bootstrap confidence interval is requested, asymmetric bootstrap *p*-values are also computed, using the method presented in Asparouhov and Muthén (2021).

## Value

The output of `lavaan::standardizedSolution()`, with bootstrap confidence intervals appended to the right, with class set to `sbt_std_boot`. It has a print method (`print.sbt_std_boot()`) that can be used to print the standardized solution in a format similar to that of the printout of the `summary()` of a `lavaan::lavaan` object.

## Author(s)

Shu Fai Cheung <https://orcid.org/0000-0002-9871-9448>. Originally proposed in an issue at GitHub <https://github.com/simsem/semTools/issues/101#issue-1021974657>, inspired by a discussion at the Google group for lavaan <https://groups.google.com/g/lavaan/c/qQBXSz5cd0o/m/R8YT5HxNAgAJ>. `boot::boot.ci()` is used to form the percentile confidence intervals in this version.

## References

Asparouhov, A., & Muthén, B. (2021). Bootstrap *p*-value computation. Retrieved from <https://www.statmodel.com/download/Bootstrap%20-%20Pvalue.pdf>

## See Also

`lavaan::standardizedSolution()`, `store_boot()`

**Examples**

```

library(lavaan)
set.seed(5478374)
n <- 50
x <- runif(n) - .5
m <- .40 * x + rnorm(n, 0, sqrt(1 - .40))
y <- .30 * m + rnorm(n, 0, sqrt(1 - .30))
dat <- data.frame(x = x, y = y, m = m)
model <-
'
m ~ a*x
y ~ b*m
ab := a*b
'

# Should set bootstrap to at least 2000 in real studies
fit <- sem(model, data = dat, fixed.x = FALSE,
           se = "boot",
           bootstrap = 100)
summary(fit)

std <- standardizedSolution_boot(fit)
std

# Print in a friendly format with only standardized solution
print(std, output = "text")

# Print in a friendly format with both unstandardized
# and standardized solution
print(std, output = "text", standardized_only = FALSE)

# hist_qq_boot() can be used to examine the bootstrap estimates
# of a parameter
hist_qq_boot(std, param = "ab")

# scatter_boot() can be used to examine the bootstrap estimates
# of two or more parameters
scatter_boot(std, params = c("ab", "a", "b"))

```

---

store\_boot

---

*Compute and Store Bootstrap Estimates*


---

**Description**

This function computes bootstrap estimates of a fitted structural equation model and stores the estimates for further processing.

**Usage**

```
store_boot(
  object,
  type = "std.all",
  do_bootstrapping = TRUE,
  R = 1000,
  boot_type = "ordinary",
  parallel = c("no", "multicore", "snow"),
  ncpus = parallel::detectCores(logical = FALSE) - 1,
  iseed = NULL,
  keep.idx = FALSE,
  bootstrapLavaan_args = list()
)
```

**Arguments**

object	A 'lavaan'-class object, fitted with 'se = "boot"'.
type	The type of standard estimates. The same argument of <code>lavaan::standardizedSolution()</code> , and support all values supported by <code>lavaan::standardizedSolution()</code> . Default is "std.all".
do_bootstrapping	If TRUE and bootstrapping was not requested when fitting the model, bootstrapping will be done using <code>lavaan::bootstrapLavaan()</code> . Default is TRUE.
R	If <code>lavaan::bootstrapLavaan()</code> is called (see do_bootstrapping), this is the number of bootstrap samples, to be used by <code>lavaan::bootstrapLavaan()</code> .
boot_type	If <code>lavaan::bootstrapLavaan()</code> is called (see do_bootstrapping), this is type of bootstrapping, to be passed to the argument type of <code>lavaan::bootstrapLavaan()</code> . Default is "ordinary". See the help page of <code>lavaan::bootstrapLavaan()</code> for details.
parallel	If <code>lavaan::bootstrapLavaan()</code> is called (see do_bootstrapping), whether parallel processing will be used. to be passed to the argument of the same name in <code>lavaan::bootstrapLavaan()</code> . Default is "no". Can be "snow" or "multicore". See the help page of <code>lavaan::bootstrapLavaan()</code> for details.
ncpus	If <code>lavaan::bootstrapLavaan()</code> is called (see do_bootstrapping), and parallel processing is to be used, this is the number of CPU cores to use, to be passed to the argument of the same name in <code>lavaan::bootstrapLavaan()</code> . Default is <code>parallel::detectCores(logical = FALSE) - 1</code> , the number of physical cores minus 1, different from the default of <code>lavaan::bootstrapLavaan()</code> but identical to the default of <code>lavaan::sem()</code> and <code>lavaan::cfa()</code> .
iseed	If <code>lavaan::bootstrapLavaan()</code> is called (see do_bootstrapping), this should be an integer used to generate reproducible bootstrap results, to be passed to the argument of the same name in <code>lavaan::bootstrapLavaan()</code> . Default is NULL but it should nearly always be set to an arbitrary integer. See the help page of <code>lavaan::bootstrapLavaan()</code> for details.
keep.idx	Whether the indices of cases selected in each bootstrap sample is to be stored. To be passed to the argument of the same name in <code>lavaan::bootstrapLavaan()</code> . Default is FALSE.

bootstrapLavaan\_args

A named list of additional arguments to be passed to `lavaan::bootstrapLavaan()`. Note that the other arguments in `store_boot()` takes precedence, overriding arguments of the same names in this list, if any.

## Details

The function `store_boot()` receives a `lavaan::lavaan` object, optionally fitted with bootstrapping standard errors requested, and compute and store the bootstrap estimates of user-defined parameters and estimates in the standardized solution.

If bootstrapping was not requested when fitting the model (i.e., `se` not set to "boot" or "bootstrap"), then bootstrapping will be conducted using `lavaan::bootstrapLavaan()` to compute bootstrap estimates of free parameters. Otherwise, the stored bootstrap estimates will be used in subsequent steps.

For standardized solution bootstrap estimates, it works by calling `lavaan::standardizedSolution()` with the bootstrap estimates of free parameters in each bootstrap sample to compute the standardized estimates in each sample.

For user-defined parameters, it works by calling the function used to compute user-defined parameters with the bootstrap estimates of free parameters in each bootstrap samples to compute the user-defined parameters.

The bootstrap estimates are then stored in the external slot of the fit object for further processing.

## Value

The original lavaan object is returned with the following objects stored in the external slot:

- `sbt_boot_std`: The matrix of bootstrap estimates in the standardized solution.
- `sbt_boot_def`: The matrix of bootstrap estimates of user-defined parameters, if any.
- `sbt_boot_ustd`: The matrix of bootstrap estimates of free parameters, if bootstrapping is not requested when fitting the model (i.e., `se` is not set to "boot" or "bootstrap" when fitting the model in lavaan).

## Author(s)

Shu Fai Cheung <https://orcid.org/0000-0002-9871-9448>. Based on `semhelpinghands::standardizedSolution_bootstrap` which was originally proposed in an issue at GitHub <https://github.com/simsem/semTools/issues/101#issue-1021974657>, inspired by a discussion at the Google group for lavaan <https://groups.google.com/g/lavaan/c/qQBXSz5cd0o/m/R8YT5HxNAgAJ>. Unlike `semhelpinghands::standardizedSolution_bootstrap` this function only computes and stores the bootstrap estimates.

## Examples

```
library(lavaan)
set.seed(5478374)
n <- 50
x <- runif(n) - .5
m <- .40 * x + rnorm(n, 0, sqrt(1 - .40))
y <- .30 * m + rnorm(n, 0, sqrt(1 - .30))
dat <- data.frame(x = x, y = y, m = m)
```

```
model <-  
'  
m ~ a*x  
y ~ b*m  
ab := a*b  
'  
  
# Should set bootstrap to at least 2000 in real studies  
fit <- sem(model, data = dat, fixed.x = FALSE,  
           se = "boot",  
           bootstrap = 100)  
summary(fit)  
  
fit <- store_boot(fit)
```

# Index

abline(), 8

boot::boot(), 9  
boot::boot.ci(), 20

gg\_hist\_qq\_boot, 2  
gg\_scatter\_boot, 5  
gg\_scatter\_boot(), 6

hist(), 8  
hist\_qq\_boot, 7  
hist\_qq\_boot(), 2, 5, 6, 9

jab\_after\_boot, 11

lavaan::bootstrapLavaan(), 22, 23  
lavaan::cfa(), 9, 22  
lavaan::lavaan, 9, 14, 20, 23  
lavaan::lavaan(), 9  
lavaan::parameterEstimates(), 13, 14, 17  
lavaan::sem(), 9, 22  
lavaan::standardizedSolution(), 13, 15,  
16, 19, 20, 22, 23

lines(), 8

parameterEstimates\_boot, 12, 13  
parameterEstimates\_boot(), 6, 9, 14, 17,  
18

print(), 15, 17  
print.sbt\_std\_boot, 15  
print.sbt\_std\_boot(), 20  
print.sbt\_ustd\_boot, 17  
print.sbt\_ustd\_boot(), 14

psych::pairs.panels(), 9

qqline(), 8, 9  
qqnorm(), 8

scatter\_boot(hist\_qq\_boot), 7  
scatter\_boot(), 6, 9

semhelpinghands::standardizedSolution\_boot\_ci(),  
23  
standardizedSolution\_boot, 12, 19  
standardizedSolution\_boot(), 4–10, 13,  
15, 16, 20  
store\_boot, 12, 21  
store\_boot(), 4, 5, 7, 9, 10, 14, 20, 23  
summary(), 14–17, 20