

# Package ‘modelbpp’

February 20, 2024

**Title** Model BIC Posterior Probability

**Version** 0.1.3

**Description** Fits the neighboring models of a fitted structural equation model and assesses the model uncertainty of the fitted model based on BIC posterior probabilities, using the method presented in Wu, Cheung, and Leung (2020)  [<doi:10.1080/00273171.2019.1574546>](https://doi.org/10.1080/00273171.2019.1574546).

**License** GPL (>= 3)

**Encoding** UTF-8

**RoxygenNote** 7.3.0

**Suggests** knitr, rmarkdown, tinytest

**Depends** R (>= 4.0.0)

**Imports** lavaan, parallel, pbapply, igraph

**VignetteBuilder** knitr

**URL** <https://sfcheung.github.io/modelbpp/>

**BugReports** <https://github.com/sfcheung/modelbpp/issues>

**LazyData** true

**NeedsCompilation** no

**Author** Shu Fai Cheung [aut, cre] ( [<https://orcid.org/0000-0002-9871-9448>](https://orcid.org/0000-0002-9871-9448)),  
Huiping Wu [aut],  
Shing On Leung [aut] ( [<https://orcid.org/0000-0002-0962-339X>](https://orcid.org/0000-0002-0962-339X)),  
Ivan Jacob Agaloos Pesigan [ctb]  
( [<https://orcid.org/0000-0003-4818-8420>](https://orcid.org/0000-0003-4818-8420))

**Maintainer** Shu Fai Cheung  [<shufai.cheung@gmail.com>](mailto:shufai.cheung@gmail.com)

**Repository** CRAN

**Date/Publication** 2024-02-20 14:20:02 UTC

**R topics documented:**

c.partables . . . . .	2
dat_cfa . . . . .	4
dat_path_model . . . . .	5
dat_path_model_p06 . . . . .	6
dat_sem . . . . .	6
dat_serial_4 . . . . .	7
dat_serial_4_weak . . . . .	8
fit_many . . . . .	8
get_add . . . . .	10
get_drop . . . . .	12
min_prior . . . . .	13
model_graph . . . . .	15
model_set . . . . .	18
partables_helpers . . . . .	22
plot.model_graph . . . . .	24
print.model_set . . . . .	25
print.partables . . . . .	27
print.sem_outs . . . . .	28

<b>Index</b>	<b>30</b>
--------------	-----------

---

c.partables	<i>Manipulate Parameter Tables</i>
-------------	------------------------------------

---

**Description**

Functions to manipulate a partables-class object

**Usage**

```
## S3 method for class 'partables'
c(...)

## S3 method for class 'model_set'
c(...)

partables_drop(x, model_names = NULL)
```

**Arguments**

...	An arbitrary number of objects. All invalid objects (see details) will be discarded. If an object is named and is not partables object, its name will be used.
x	A partables-class object.
model_names	A character vector of the names of models in a partables-class object.

## Details

The partables-class objects have a `c()` method that can be used to combine parameter tables. Each object must be

a. a partables-class object, b. a `model_set`-class object, c. a lavaan-class object, or d. a parameter table of the class `lavaan.data.frame()`, usually generated by `lavaan::parameterTable()`. Other objects will be discarded.

Names will be used when combining objects. If two objects have the same names, then only the first one will be retained. No warning message will be issued. Users are encouraged to explicitly name all objects carefully.

Note that, to invoke this method, the first object must be a partables object.

The `model_set` class also has a `c`-method. It will replace the first object by the stored partables and then call the `c`-method of partables objects.

The function `partables_drop()` is for dropping models from a partables-class object.

## Value

A partables-class objects with all the objects supplied combined together. If an object is a lavaan-class object, its parameter table will be retrieved by `lavaan::parameterTable()`. If an object is a `model_set`-class object, the stored partables-class object will be retrieved.

## Author(s)

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

## Examples

```
library(lavaan)

moda <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1
ab := a*b
"

fita <- sem(moda, dat_path_model, fixed.x = TRUE)

outa <- model_set(fita,
  progress = FALSE,
  parallel = FALSE)

modb <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x2
ab := a*b
"
```

```
fitb <- sem(modb, dat_path_model, fixed.x = TRUE)

outb <- model_set(fitb,
                  progress = FALSE,
                  parallel = FALSE)

mod2 <-
"
x2 ~ 0*x3 + 0*x4
x1 ~ 0*x3
"
fit2 <- sem(mod2, dat_path_model)

mod3 <-
"
x2 ~ x3 + 0*x4
x1 ~ x3
"
fit3 <- sem(mod3, dat_path_model)

out <- c(outa$models, user2 = fit2, outb$models, user3 = fit3)
out

out2 <- c(outa, user2 = fit2, outb$models, user3 = fit3)
out2

out3 <- c(outa, user2 = fit2, outb, user3 = fit3)
out3
```

---

dat\_cfa

*A Sample Dataset Based On a Confirmatory Factor Analysis Model  
(For Testing)*

---

### **Description**

Generated from a confirmatory factor analysis model (n = 200).

### **Usage**

dat\_cfa

### **Format**

A data frame with six variables:

**x1** Indicator

**x2** Indicator

**x3** Indicator

- x4** Indicator
- x5** Indicator
- x6** Indicator

### Details

The model used to generate this dataset:

```
f1 =~ x1 + x2 + x3 + x5
f2 =~ x3 + x4 + x5 + x6
f1 ~~ f2
```

---

dat_path_model	<i>A Sample Dataset Based on a Path Model (For Testing)</i>
----------------	---

---

### Description

Generated from the a path model (n = 100).

### Usage

```
dat_path_model
```

### Format

A data frame with four variables:

- x1** Predictor
- x2** Predictor
- x3** Mediator
- x4** Outcome

### Details

The model used to generate this dataset:

```
x1 ~~ x2
x3 ~ x1 + x2
x4 ~ x3 + x1 + x2
```

---

dat\_path\_model\_p06      *A Sample Dataset Based On a Complex Path Model (For Testing)*

---

**Description**

Generated from a complex path model (n = 200).

**Usage**

dat\_path\_model\_p06

**Format**

A data frame with six variables:

**x1** Predictor

**x2** Predictor

**x3** Predictor

**y4** Mediator

**y5** Mediator

**y6** Outcome

**Details**

The model used to generate this dataset:

$y4 \sim x1 + x2 + x3$

$y5 \sim y4 + x1 + x2$

$y6 \sim y4 + y5 + x1 + x2 + x3$

$x1 \sim x2 + x3$

$x2 \sim x3$

---

dat\_sem      *A Sample Dataset Based On a Structural Model (For Testing)*

---

**Description**

Generated from a structural model with latent variables (n = 250).

**Usage**

dat\_sem

**Format**

An object of class `data.frame` with 250 rows and 16 columns.

**Details**

The model to be fitted:

```
f1 =~ x1 + x2 + x3 + x4
f2 =~ x5 + x6 + x7 + x8
f3 =~ x9 + x10 + x11 + x12
f4 =~ x13 + x14 + x15 + x16
f3 ~ f1 + f2
f4 ~ f3
```

---

dat\_serial\_4

*A Sample Dataset Based On a Serial Mediation Model (For Testing)*

---

**Description**

Generated from a serial mediation model ( $n = 100$ ).

**Usage**

```
dat_serial_4
```

**Format**

A data frame with four variables:

**x** Predictor  
**m1** Mediator  
**m2** Mediator  
**y** Outcome

**Details**

The model used to generate this dataset:

```
m1 ~ x
m2 ~ m1
y ~ m2
```

---

dat_serial_4_weak	<i>A Sample Dataset Based On a Serial Mediation Model With Weak Paths (For Testing)</i>
-------------------	---

---

**Description**

Generated from a serial mediation model (n = 100).

**Usage**

```
dat_serial_4_weak
```

**Format**

A data frame with four variables:

**x** Predictor

**m1** Mediator

**m2** Mediator

**y** Outcome

**Details**

The model to be fitted:

$$m1 \sim x$$
$$m2 \sim m1 + x$$
$$y \sim m2 + m1 + x$$

---

fit_many	<i>Fit a List of Models</i>
----------	-----------------------------

---

**Description**

Fit a list of models to a dataset.

**Usage**

```
fit_many(  
  model_list,  
  sem_out,  
  original = NULL,  
  parallel = FALSE,  
  ncores = max(parallel::detectCores(logical = FALSE) - 1, 1),  
  make_cluster_args = list(),  
  progress = TRUE,  
  verbose = TRUE  
)
```



**Arguments**

model_list	A list of parameter tables to be used by <code>lavaan::lavaan()</code> or <code>update()</code> . Usually generated by <code>get_add()</code> or <code>get_drop()</code> .
sem_out	The output from an structural equation modeling function. It currently supports <code>lavaan::lavaan</code> objects only. Usually the one used in <code>model_set()</code> , <code>get_add()</code> or <code>get_drop()</code> to generate the list of models.
original	String. If provided, it should be a name of a model in <code>model_list</code> , with which differences in model degrees of freedom will be computed for other models. If NULL, the default, then the model in <code>sem_out</code> will be used to computed the differences in model degrees of freedom. If NA, then differences in model <i>df</i> will not be computed.
parallel	If TRUE, parallel processing will be used to fit the models. Default is FALSE.
ncores	Numeric. The number of CPU cores to be used if <code>parallel</code> is TRUE.
make_cluster_args	A list of named arguments to be passed to <code>parallel::makeCluster()</code> . Used by advanced users to configure the cluster if <code>parallel</code> is TRUE. Default is <code>list()</code> .
progress	Whether a progress bar will be displayed, implemented by the <code>pbapply</code> package. Default is TRUE.
verbose	Whether additional messages will be displayed, such as the expected processing time. Default is TRUE.

**Details**

It receives a list of models, defined by lavaan parameter tables (usually generated by `model_set()`, `get_add()` or `get_drop()`), and fit them to a dataset stored in a lavaan-class object.

This function is called by `model_set()` and usually users do not need to call it. It is exported for advanced users.

**Value**

An object of the class `sem_outs`, a list with the following major elements:

- `fit`: A named list of `lavaan::lavaan()` output objects or `update()` for fitting a model with the added parameters.
- `change`: A numeric vector, of the same length as `fit`. The change in model *df* for each fit compared to the original model. A positive number denotes one less free parameter. A negative number denotes one more free parameter or one less constraint.
- `converged`: A named vector of boolean values, of the same length as `fit`. Indicates whether each fit converged or not.
- `post_check`: A named vector of boolean values, of the same length as `fit`. Indicates whether the solution of each fit is admissible or not. Checked by `lavaan::lavInspect()` with the what argument set to "post.check".

**Author(s)**

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

## Examples

```
library(lavaan)
dat <- dat_path_model
mod <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1
ab := a*b
"
fit <- sem(mod, dat_path_model, fixed.x = TRUE)
mod_to_add <- get_add(fit)
fit_many(mod_to_add, fit)
```

---

get\_add

*Models That Are Less Restricted*

---

## Description

Generate a list of models with one or more fixed parameter freed.

## Usage

```
get_add(
  sem_out,
  must_add = NULL,
  must_not_add = NULL,
  remove_constraints = TRUE,
  exclude_error_cov = TRUE,
  df_change = 1,
  model_id = NA,
  keep_correct_df_change = TRUE,
  remove_duplicated = TRUE,
  progress = FALSE
)
```

## Arguments

sem_out	The original model, which is the output from an structural equation modeling function. Currently support <code>lavaan::lavaan</code> objects only.
must_add	A character vector of parameters, named in <code>lavaan::lavaan()</code> style (e.g., "y ~ x"), that must be added. Default is NULL.
must_not_add	A character vector of parameters, named in <code>lavaan::lavaan()</code> style (e.g., "x1 ~~ x1"), that must not be added. Default is NULL.
remove_constraints	Whether equality constraints will be removed. Default is TRUE.

exclude_error_cov	Exclude error covariances of indicators. Default is TRUE.
df_change	How many degrees of freedom ( <i>df</i> ) away in the list. All models with <i>df</i> change less than or equal to this number will be included, taking into account requirements set by other arguments. Default is 1.
model_id	The identification number of the starting model. Default is NA, no identification number.
keep_correct_df_change	Keep only models with actual <i>df</i> change equal to expected <i>df</i> change.
remove_duplicated	If TRUE, the default, duplicated models are removed.
progress	Whether a progress bar will be displayed, implemented by the pbapply package. Default is FALSE.

### Details

It generates a list of models with one or more fixed parameter freed (and the degrees of freedom, *df*, increases by one or more). If a model has equality constraints, models with one or more of the constraints between two free parameters released will also be included.

Graphically, paths or covariances are "added" to form the list of models.

The models to be included are identified by `lavaan::modificationIndices()`.

The models will be checked by lavaan to make sure that the decrease in model degrees of freedom is of the expected value.

This function is called by `model_set()` and usually users do not need to call it. It is exported for advanced users.

### Value

An object of the class `partables`, a named list of parameter tables, each of them to be used by `lavaan::lavaan()` or `update()` for fitting a model with the added parameters.

### Author(s)

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

### See Also

`print.partables()`

### Examples

```
library(lavaan)
dat <- dat_path_model
mod <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1
```

```

ab := a*b
"
fit <- sem(mod, dat_path_model, fixed.x = TRUE)
mod_to_add <- get_add(fit)
mod_to_add

```

---

get\_drop

*Models That Are More Restricted*


---

## Description

Generate a list of models with one or more free parameter dropped (fixed to zero).

## Usage

```

get_drop(
  sem_out,
  must_drop = NULL,
  must_not_drop = NULL,
  df_change = 1,
  model_id = NA,
  keep_correct_df_change = TRUE,
  remove_duplicated = TRUE,
  progress = FALSE
)

```

## Arguments

sem_out	The original model, which is the output from an structural equation modeling function. Currently support <code>lavaan::lavaan</code> objects only.
must_drop	A character vector of parameters, named in <code>lavaan::lavaan()</code> style (e.g., "y ~ x"), that must be included. Default is NULL.
must_not_drop	A character vector of parameters, named in <code>lavaan::lavaan()</code> style (e.g., "x1 ~~ x1"), that must not be included. Default is NULL.
df_change	How many degrees of freedom away in the list. All models with <i>df</i> change less than or equal to this number will be included, taking into account requirements set by other arguments. Default is 1.
model_id	The identification number of the starting model. Default is NA, no identification number.
keep_correct_df_change	Keep only models with actual <i>df</i> change equal to expected <i>df</i> change.
remove_duplicated	If TRUE, the default, duplicated models are removed.
progress	Whether a progress bar will be displayed, implemented by the <code>pbapply</code> package. Default is FALSE.

### Details

It generates a list of models with one or more free parameters dropped, that is, fixed to zero (with degrees of freedom, *df*, increases by one or more).

All free parameters are included in the pool of candidates, except for those explicitly requested to be kept.

The models will be checked by lavaan to make sure that the increase in model degrees of freedom is of the expected value.

This function is called by `model_set()` and usually users do not need to call it. It is exported for advanced users.

### Value

An object of the class `partables`, a named list of parameter tables, each of them to be used by `lavaan::lavaan()` or `update()` for fitting a model with the added parameters.

### Author(s)

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

### See Also

`print.partables()`

### Examples

```
library(lavaan)

dat <- dat_path_model
mod <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1 + x2
ab := a*b
"

fit <- sem(mod, dat_path_model, fixed.x = TRUE)
mod_to_drop <- get_drop(fit)
mod_to_drop
```

---

min\_prior

*Minimum Prior*

---

### Description

Find the minimum prior probability required to achieve the desired BIC posterior probability.

**Usage**

```
min_prior(bic, bpp_target, target_name = "original")
```

**Arguments**

bic	A named vector of BIC values for a set of models.
bpp_target	A value from zero to 1. The desired BIC posterior probability.
target_name	The name of the original model, as appeared in the names of bic.

**Details**

It assumes that all models other than the original model have the same prior probabilities.

This function is called by `model_set()` or `print.model_set()` and usually users do not need to call it. It is exported for advanced users.

**Value**

A scalar. The required prior probability.

**Author(s)**

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

**References**

Wu, H., Cheung, S. F., & Leung, S. O. (2020). Simple use of BIC to assess model selection uncertainty: An illustration using mediation and moderation models. *Multivariate Behavioral Research*, 55(1), 1–16. doi:10.1080/00273171.2019.1574546

**See Also**

[model\\_set\(\)](#) and [print.model\\_set\(\)](#)

**Examples**

```
library(lavaan)

dat <- dat_path_model

mod <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1
ab := a*b
"

fit <- sem(mod, dat_path_model, fixed.x = TRUE)

out <- model_set(fit)
```

```
min_prior(out$bic, bpp_target = .8)
```

---

model_graph	<i>Generate a Graph of Models</i>
-------------	-----------------------------------

---

## Description

Generate an 'igraph' object from a 'model\_set' object.

## Usage

```
model_graph(
  object,
  node_size_by_x = TRUE,
  x = NULL,
  node_size = 5,
  min_size = 5,
  max_size = 35,
  color_original = "lightblue",
  color_add = "burlywood1",
  color_drop = "lightgreen",
  color_others = "lightgrey",
  color_label = "black",
  node_label_size = 1,
  original = "original",
  drop_redundant_direct_paths = TRUE,
  label_arrow_by_df = NULL,
  arrow_label_size = 1,
  weight_arrows_by_df = c("inverse", "normal", "none"),
  arrow_min_width = 0.5,
  arrow_max_width = 2,
  progress = TRUE,
  short_names = FALSE,
  ...
)
```

## Arguments

object	Must be a model_set-class object for now.
node_size_by_x	Logical. Whether node (vertex) sizes are determined by a variable. Default is TRUE. See x below on how size is determined.
x	If not NULL, it should be a numeric vector of length equal to the number of models. The node sizes will be proportional to the values of x, offset by min_size. If NULL, the default, the BIC posterior probabilities stored in object will be retrieved.

node_size	If node_size_by_x is FALSE, this is the size for all nodes.
min_size	The minimum size of a node. Default is 5.
max_size	The maximum size of a node. Default is 35.
color_original	The color of node of the original model. Default is "lightblue".
color_add	The color of the nodes of models formed by adding one or more free parameters to the original model. Default is "burlywood1".
color_drop	The color of the nodes of models formed by dropping one or more free parameters from the original model. Default is "lightgreen".
color_others	The color of other models not specified above. Default is "grey50".
color_label	The color of the text labels of the nodes. Default is "black".
node_label_size	The size of the labels of the nodes. Default is 1.
original	String. The name of the original model (target model). Default is "original".
drop_redundant_direct_paths	Logical. Whether the redundant direct path between two models. A direct path is redundant if two models are also connected through at least one another model. Default is TRUE.
label_arrow_by_df	If TRUE, then an arrow (edge) is always labelled by the difference in model <i>dfs</i> . If FALSE, then no arrows are labelled. If NULL, then arrows are labelled when not all differences in model <i>dfs</i> are equal to one. Default is NULL.
arrow_label_size	The size of the labels of the arrows (edges), if labelled. Default is 1.
weight_arrows_by_df	String. Use if model <i>df</i> differences are stored. If "inverse", larger the difference in model <i>df</i> , <i>narrower</i> an arrow. That is, more similar two models are, thicker the arrow. If "normal", larger the difference in model <i>df</i> , <i>wider</i> an arrow. If "none", then arrow width is constant, set to arrow_max_width. Default is "inverse".
arrow_min_width	If weight_arrows_by_df is not "none", this is the minimum width of an arrow.
arrow_max_width	If weight_arrows_by_df is not "none", this is the maximum width of an arrow. If weight_arrows_by_df is "none", this is the width of all arrows.
progress	Whether a progress bar will be displayed for some steps (e.g., checking for nested relations). Default is TRUE.
short_names	If TRUE and short model names are stored, they will be used as model labels. Please print the object with short_names = TRUE to find the corresponding full model names.
...	Optional arguments. Not used for now.



## Details

It extracts the model list stored in object, creates an adjacency matrix, and then creates an igraph object customized for visualizing model relations.

### Construction of the Graph:

This is the default way to construct the graph when the model set is automatically by `model_set()`.

- Each model is connected by an arrow, pointing from one model to another model that
  - a. can be formed by adding one or more free parameter, or
  - b. can be formed by releasing one or more equality constraint between two parameters.
  - c. has nested relation with this model as determined by the method proposed by Bentler and Satorra (2010), if the models are not generated internally.

That is, it points to a model with more degrees of freedom (more complicated), and is nested within that model in either parameter sense or covariance sense.

- By default, the size of the node for each model is scaled by its BIC posterior probability, if available. See *The Size of a Node* below.
- If a model is designated as the original (target) model, than he original model, the models with more degrees of freedom than the original model, and the models with fewer degrees of freedom than the original models, are colored differently.
- The default layout is the Sugiyama layout, with simpler models (models with fewer degrees of freedom) on the top. The lower a model is in the network, the more the degrees of freedom it has. This layout is suitable for showing the nested relations of the models. Models on the same level (layer) horizontally have the same model *df*.

The output is an igraph object. Users can customize it in any way they want using functions from the igraph package.

If a model has no nested relation with all other model, it will not be connected to other models.

If no model is named original (default is "original"), then no model is colored as the original model.

### User-Provided Models:

If object contained one or more user-provided models which are not generated automatically by `model_set()` or similar functions (e.g., `gen_models()`), then the method by Bentler and Satorra (2010) will be used to determine model relations. Models connected by an arrow has a nested relation based on the NET method by Bentler and Satorra (2010). An internal function inspired by `semTools::net()` is used to implement the NET method.

### The Size of a Node:

When a model is scaled by  $x$ , which usually is the BIC posterior probability, its size is determined by:

$$\text{max\_size} * (x - \min(x)) / (\text{max}(x) - \min(x)) + \text{min\_size}$$

## Value

A `model_graph`-class object that can be used as as an igraph-object, with a plot method (`plot.model_graph()`) with settings suitable for plotting a network of models with BIC posterior probabilities computed.

**Author(s)**

Shu Fai Cheung <https://orcid.org/0000-0002-9871-9448> The internal function for nesting inspired by `semTools::net()`, which was developed by Terrence D. Jorgensen.

**References**

Bentler, P. M., & Satorra, A. (2010). Testing model nesting and equivalence. *Psychological Methods*, 15(2), 111–123. doi:10.1037/a0019625

Asparouhov, T., & Muthén, B. (2019). Nesting and Equivalence Testing for Structural Equation Models. *Structural Equation Modeling: A Multidisciplinary Journal*, 26(2), 302–309. doi:10.1080/10705511.2018.1513795

**Examples**

```
library(lavaan)

mod <-
"
m1 ~ x
y ~ m1
"

fit <- sem(mod, dat_serial_4, fixed.x = TRUE)

out <- model_set(fit)
out

g <- model_graph(out)
plot(g)
```

---

model\_set

*BIC Posterior Probabilities of Neighboring Models*


---

**Description**

Identify neighboring models, fit them, and return the BIC posterior probabilities.

**Usage**

```
model_set(
  sem_out,
  partables = NULL,
  model_set_out = NULL,
  prior_sem_out = NULL,
  must_add = NULL,
  must_not_add = NULL,
  must_drop = NULL,
```

```

    must_not_drop = NULL,
    remove_constraints = TRUE,
    exclude_error_cov = TRUE,
    df_change_add = 1,
    df_change_drop = 1,
    remove_duplicated = TRUE,
    fit_models = ifelse(!is.null(model_set_out$fit), FALSE, TRUE),
    compute_bpp = TRUE,
    original = "original",
    parallel = FALSE,
    ncores = max(parallel::detectCores(logical = FALSE) - 1, 1),
    make_cluster_args = list(),
    progress = TRUE,
    verbose = TRUE,
    skip_check_sem_out = FALSE
  )

  gen_models(
    sem_out,
    must_add = NULL,
    must_not_add = NULL,
    must_drop = NULL,
    must_not_drop = NULL,
    remove_constraints = TRUE,
    exclude_error_cov = TRUE,
    df_change_add = 1,
    df_change_drop = 1,
    remove_duplicated = TRUE,
    progress = TRUE,
    output = c("partables", "model_set")
  )

```

## Arguments

sem_out	It can be the output from an SEM function. Currently it supports <a href="#">lavaan::lavaan</a> objects only. If it is a named list of <a href="#">lavaan::lavaan</a> objects, then all arguments for model generation will be ignored, and models will not be refitted. Users need to ensure that the models can be meaningfully compared because they will not be checked.
partables	A partables-class object, usually generated by <a href="#">get_add()</a> or <a href="#">get_drop()</a> . A named list of parameter tables to be fitted along with the original model in sem_out. If supplied, all arguments related to identifying models will be ignored. Default is NULL.
model_set_out	If set to the output of a previous call to <a href="#">model_set()</a> (a model_set-class object), the list of stored models will be used. All arguments related to generate neighboring models will be ignored. If supplied, sem_out will also be ignored and will be retrieved from model_set_out, and partables will also be ignored. Default is NULL.

prior_sem_out	The prior of the model fitted in sem_out. Default is NULL, and all models will have equal prior probabilities.
must_add	A character vector of parameters, named in <code>lavaan::lavaan()</code> style (e.g., "y ~ x"), that must be added. Default is 'NULL'.
must_not_add	A character vector of parameters, named in <code>lavaan::lavaan()</code> style (e.g., "x1 ~~ x1"), that must not be added. Default is NULL.
must_drop	A character vector of parameters, named in <code>lavaan::lavaan()</code> style (e.g., "y ~ x"), that must be included. Default is NULL.
must_not_drop	A character vector of parameters, named in <code>lavaan::lavaan()</code> style (e.g., "x1 ~~ x1"), that must not be included. Default is NULL.
remove_constraints	Whether equality constraints will be removed. Default is "TRUE".
exclude_error_cov	Exclude error covariances of indicators. Default is TRUE.
df_change_add	How many degrees of freedom ( <i>df</i> ) away in the list. All models with <i>df</i> change less than or equal to this number will be included, taking into account requirements set by other arguments. Default is 1.
df_change_drop	How many degrees of freedom away in the list. All models with <i>df</i> change less than or equal to this number will be included, taking into account requirements set by other arguments. Default is 1.
remove_duplicated	If TRUE, the default, duplicated models are removed.
fit_models	If TRUE, the models will be fitted to the data, usually stored in sem_out. If FALSE, the models will be returned as is, in the element models of the output. If model_set_out is set and models have been fitted, then default is FALSE. Otherwise, default is TRUE.
compute_bpp	If TRUE, then BIC posterior probabilities will be computed. Default is TRUE.
original	String. The name of the original (target) model. Default is "original". Used if prior_sem_out is unnamed and only has one value.
parallel	If TRUE, parallel processing will be used to fit the models. Default is FALSE.
ncores	Numeric. The number of CPU cores to be used if parallel is TRUE.
make_cluster_args	A list of named arguments to be passed to <code>parallel::makeCluster()</code> . Used by advanced users to configure the cluster if parallel is TRUE. Default is <code>list()</code> .
progress	Whether a progress bar will be displayed, implemented by the <code>pbapply</code> package or by <code>utils::txtProgressBar</code> . Default is TRUE.
verbose	Whether additional messages will be displayed, such as the expected processing time. Default is TRUE.
skip_check_sem_out	If TRUE and sem_out is set, check whether sem_out is of a supported type (estimator is "ML" and the model has only one group). If not, an error will be raised. Can be set to FALSE for experimenting the functions on models not officially supported.
output	If "model_set", then the output is a model_set-class object. If "partables", the output is a partables-class object. Default is partables.

## Details

It computes the BIC posterior probabilities of a set of models by the method presented in Wu, Cheung, and Leung (2020).

First, a list of model is identified based on user-specified criteria. By default, models differ from a fitted model by one degree of freedom, the 1-df-away *neighboring* models, will be found using `get_add()` and `get_drop`.

Second, these models will be fitted to the sample dataset, and their BICs will be computed.

Third, their BIC posterior probabilities will be computed using their BICs. By default, equal prior probabilities for all the models being fitted will be assumed in the current version. This can be changed by `prior_sem_out`.

The results can then be printed, with the models sorted by descending order of BIC posterior probabilities. The results can also be visualized using `model_graph()`.

## Value

The function `model_set()` returns an object of the class `model_set`, a list with the following major elements:

- `models`: A named list of parameter tables. Each represent the models identified.
- `bic`: A numeric vector, of the same length as `model`. The BIC of each model.
- `postprob`: A numeric vector, of the same length as `model`. The BIC posterior probability of each model.
- `fit`: A named list of `lavaan::lavaan()` output objects or `update()` for fitting a model with the added parameters, of the same length as `model`.
- `change`: A numeric vector, of the same length as `model`. The change in model df for each fit. A positive number denotes one less free parameter. A negative number denotes one more free parameter or one less constraint.
- `converged`: A named vector of boolean values, of the same length as `model`. Indicates whether each fit converged or not.
- `post_check`: A named vector of boolean values, of the same length as `model`. Indicates whether the solution of each fit is admissible or not. Checked by `lavaan::lavInspect()`.

The object returned by `gen_models()` depends on the argument `output`. See the argument `output` for the details

## Functions

- `model_set()`: Compute the BPPs of a list of models. Can generate the models and/or fit the models. Can also accept pregenerated models, or just update BPPs.
- `gen_models()`: Generate a list of models (parameter tables).

## Author(s)

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

## References

Wu, H., Cheung, S. F., & Leung, S. O. (2020). Simple use of BIC to assess model selection uncertainty: An illustration using mediation and moderation models. *Multivariate Behavioral Research*, 55(1), 1–16. doi:10.1080/00273171.2019.1574546

## See Also

[print.model\\_set\(\)](#)

## Examples

```
library(lavaan)

dat <- dat_path_model

mod <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1
ab := a*b
"

fit <- sem(mod, dat_path_model, fixed.x = TRUE)

out <- model_set(fit)
out
```

---

partables\_helpers

*Helper Functions For partables-Class Objects*

---

## Description

For tasks such as comparing two parameter tables inside a partables-class object.

## Usage

```
identical_partables(object1, object2)

is_partable(x)

same_variables(x)

get_partables(x)

to_partables(...)
```

**Arguments**

object1	A lavaan parameter table or similar object.
object2	A lavaan parameter table or similar object.
x	An object to be checked.
...	The objects to be combined.

**Details**

The function `identical_partables()` compare two lavaan parameter tables and see whether they are identical. (Adapted from a similar function in the package `semhelpinghands`).

The function `is_partable()` tries to check whether an object is "likely" to be a parameter table that can be used by `lavaan::lavaan()` and its wrappers, such as `lavaan::sem()`.

Note that there is no guarantee the the parameter table makes sense or will not lead to error when fitted. It can only check whether it has the required columns.

The function `same_variables()` check whether all parameter labels in a `partables`-class object use the same observed variables.

The function `get_partables()` extract the partable object from a `model_set`-class object.

The function `to_partables()` combine objects to create a `partables`-class object. The objects to be combined can be a `lavaan`-class object (e.g., the output of `lavaan::sem()`) or a parameter table of lavaan.

**Value**

The function `identical_partables()` returns either TRUE or FALSE.

The function `is_partable()` returns either TRUE or FALSE.

The function `same_variables()` returns either TRUE or FALSE.

The function `get_partables()` returns a `partables`-class object.

The function `to_partables()` returns a `partables`-class object, created from the objects supplied.

**Examples**

```
library(lavaan)
mod1 <-
"
x3 ~ x1
x2 ~ x4
"
mod2 <-
"
x2 ~ x4
x3 ~ x1
"
fit1 <- sem(mod1, dat_path_model)
fit2 <- sem(mod2, dat_path_model)
pt1 <- parameterTable(fit1)
```

```
pt2 <- parameterTable(fit2)
identical_partables(pt1, pt2)

is_partable(pt1)

out <- model_set(fit1,
                 fit_models = FALSE)
same_variables(get_partables(out))

out <- model_set(fit1,
                 fit_models = FALSE)
get_partables(out)

fit1 <- sem(mod1, dat_path_model)
fit2 <- sem(mod2, dat_path_model)
pt1 <- parameterTable(fit1)
pt2 <- parameterTable(fit2)

to_partables(fit1, fit2)
to_partables(pt1, pt2)
```

---

plot.model\_graph      *Plot a Network of Models*

---

## Description

Plot a network of models generated by `model_graph()`.

## Usage

```
## S3 method for class 'model_graph'
plot(x, ...)
```

## Arguments

`x`                    The output of `model_graph()`. (Named `x` because it is required in the naming of arguments of the `plot` generic function.)

`...`                 Additional arguments, passed to `plot.igraph()`.

## Details

This function is the plot method of `model_graph` objects, the output of `model_graph()`.

For now, it simply passes the object to `plot.igraph()`. This function is created for possible customization of the plot in the future.



**Value**

NULL. Called for its side effect.

**See Also**

[model\\_graph\(\)](#)

**Examples**

```
library(lavaan)

dat <- dat_path_model

mod <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1
ab := a*b
"

fit <- sem(mod, dat_path_model, fixed.x = TRUE)

out <- model_set(fit)
out

g <- model_graph(out)
plot(g)
```

---

print.model\_set      *Print a model\_set-Class Object*

---

**Description**

Print the content of a model\_set-class object.

**Usage**

```
## S3 method for class 'model_set'
print(
  x,
  bic_digits = 3,
  bpp_digits = 3,
  sort_models = TRUE,
  max_models = 20,
  bpp_target = NULL,
  target_name = "original",
```

```

    more_fit_measures = NULL,
    fit_measures_digits = 3,
    short_names = FALSE,
    ...
)

```

### Arguments

<code>x</code>	A <code>model_set</code> -class object.
<code>bic_digits</code>	The number of decimal places to be displayed for BIC. Default is 3.
<code>bpp_digits</code>	The number of decimal places to be displayed for BIC posterior probability and prior probabilities. Default is 3.
<code>sort_models</code>	Whether the models will be sorted by BIC posterior probability. Default is TRUE.
<code>max_models</code>	The maximum number of models to be printed. Default is 20.
<code>bpp_target</code>	The desired BIC probability. Used to compute and print the minimum prior probability of the target model required to achieve <code>bpp_target</code> . Default is NULL.
<code>target_name</code>	The name of the target model as appeared in the model list. Default is "original". Used if <code>bpp_target</code> is not NULL.
<code>more_fit_measures</code>	Character vector. To be passed to <code>lavaan::fitMeasures()</code> . Default is NULL. If not NULL, these are the additional measures to be printed.
<code>fit_measures_digits</code>	The number of decimal places to be displayed for additional fit measures, if requested. Default is 3.
<code>short_names</code>	If TRUE, then simple short names will be printed along with full model names. Default is FALSE. Short names can be used when interpreting the graph from <code>model_graph()</code> if short names are used in the graph.
<code>...</code>	Optional arguments. Ignored.

### Details

It is the print method of the output of `model_set()`.

### Value

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

### Author(s)

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

### See Also

A `model_set`-class object is generated by `model_set()`.

## Examples

```
library(lavaan)

dat <- dat_path_model

mod <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1
ab := a*b
"

fit <- sem(mod, dat_path_model, fixed.x = TRUE)

out <- model_set(fit)
out
```

---

print.partables      *Print a partables-Class Object*

---

## Description

Print the content of a partables-class object.

## Usage

```
## S3 method for class 'partables'
print(x, max_tables = 10, ...)
```

## Arguments

x	A partables-class object.
max_tables	The maximum number of models to be printed. Default is 10.
...	Optional arguments. Ignored.

## Details

The print method for the output of `gen_models()`, `get_add()`, and `get_drop()`.

## Value

x is returned invisibly. Called for its side effect.

## Author(s)

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

**See Also**

[gen\\_models\(\)](#), [get\\_add\(\)](#), and [get\\_drop\(\)](#).

**Examples**

```
library(lavaan)
dat <- dat_path_model
mod <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1
ab := a*b
"
fit <- sem(mod, dat_path_model, fixed.x = TRUE)
mod_to_add <- get_add(fit)
mod_to_add
print(mod_to_add, max_tables = 1)
mod_to_drop <- get_drop(fit)
mod_to_drop
print(mod_to_drop, max_tables = 1)
```

---

<code>print.sem_outs</code>	<i>Print an sem_outs-Class Object</i>
-----------------------------	---------------------------------------

---

**Description**

Print the content of an `sem_outs`-class object.

**Usage**

```
## S3 method for class 'sem_outs'
print(x, max_models = 20, ...)
```

**Arguments**

<code>x</code>	An <code>sem_outs</code> -class object.
<code>max_models</code>	The maximum number of models to be printed. Default is 20.
<code>...</code>	Optional arguments. Ignored.

**Details**

The print method for the output of `fit_many()`.

**Value**

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

**Author(s)**

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

**See Also**

An `sem_outs`-class object is generated by `fit_many()`.

**Examples**

```
library(lavaan)
dat <- dat_path_model
mod <-
"
x3 ~ a*x1 + b*x2
x4 ~ a*x1
ab := a*b
"
fit <- sem(mod, dat_path_model, fixed.x = TRUE)
mod_to_add <- get_add(fit)
out <- fit_many(mod_to_add, fit)
out
print(out, max_models = 1)
```

# Index

- \* **datasets**
  - dat\_cfa, 4
  - dat\_path\_model, 5
  - dat\_path\_model\_p06, 6
  - dat\_sem, 6
  - dat\_serial\_4, 7
  - dat\_serial\_4\_weak, 8
- c.model\_set (c.partables), 2
- c.partables, 2
- dat\_cfa, 4
- dat\_path\_model, 5
- dat\_path\_model\_p06, 6
- dat\_sem, 6
- dat\_serial\_4, 7
- dat\_serial\_4\_weak, 8
- fit\_many, 8
- gen\_models (model\_set), 18
- gen\_models(), 17, 21, 27, 28
- get\_add, 10
- get\_add(), 9, 19, 21, 27, 28
- get\_drop, 12, 21
- get\_drop(), 9, 19, 27, 28
- get\_partables (partables\_helpers), 22
- get\_partables(), 23
- identical\_partables
  - (partables\_helpers), 22
- identical\_partables(), 23
- is\_partable (partables\_helpers), 22
- is\_partable(), 23
- lavaan::fitMeasures(), 26
- lavaan::lavaan, 9, 10, 12, 19
- lavaan::lavaan(), 9–13, 20, 21, 23
- lavaan::lavInspect(), 9, 21
- lavaan::modificationIndices(), 11
- lavaan::parameterTable(), 3
- lavaan::sem(), 23
- manipulate\_partables (c.partables), 2
- min\_prior, 13
- model\_graph, 15
- model\_graph(), 21, 24, 25
- model\_set, 18
- model\_set(), 9, 11, 13, 14, 17, 19, 21, 26
- partables\_drop (c.partables), 2
- partables\_drop(), 3
- partables\_helpers, 22
- plot.igraph(), 24
- plot.model\_graph, 24
- plot.model\_graph(), 17
- print.model\_set, 25
- print.model\_set(), 14, 22
- print.partables, 27
- print.partables(), 11, 13
- print.sem\_outs, 28
- same\_variables (partables\_helpers), 22
- same\_variables(), 23
- semTools::net(), 17, 18
- to\_partables (partables\_helpers), 22
- to\_partables(), 23
- update(), 9, 11, 13, 21