Package 'lmhelprs'

June 30, 2026

Title Helper Functions for Linear Model Analysis

Version 0.4.5

Description A collection of helper functions for multiple regression models fitted by lm(). Most of them are simple functions for simple tasks which can be done with coding, but may not be easy for occasional users of R. Most of the tasks addressed are those sometimes needed when using the 'manymome' package (Cheung and Cheung, 2023, <doi:10.3758/s13428-023-02224-z>) and 'stdmod' package (Cheung, Cheung, Lau, Hui, and Vong, 2022, <doi:10.1037/hea0001188>). However, they can also be used in other scenarios.

URL https://sfcheung.github.io/lmhelprs/

BugReports https://github.com/sfcheung/lmhelprs/issues

License GPL (>= 3)

Encoding UTF-8

Suggests knitr, rmarkdown, tinytest, semPlot

VignetteBuilder knitr

Depends R (>= 2.10)

LazyData true

Imports stats

Config/roxygen2/version 8.0.0

NeedsCompilation no

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

Maintainer Shu Fai Cheung <shufai.cheung@gmail.com>

Repository CRAN

Date/Publication 2026-06-30 11:00:02 UTC

Contents

data_test1 2
hierarchical 3
hierarchical_lm 4
lm_list_to_partable 5
many_lm 8
print.hierarchical_lm 10
print.lm_list_lmhelprs 11
summary.lm_list_lmhelprs 12
test_highest 13

Index **15**

`data_test1` *Sample Data: For Testing*

Description

A eight-variable dataset with 100 cases.

Usage

```
data_test1
```

Format

A data frame with 100 rows and 8 variables:

x1 Predictor. Numeric.

x2 Predictor. Numeric.

x3 Predictor. Numeric.

x4 Predictor. Numeric.

x5 Predictor. Numeric.

y Outcome. Numeric.

cat1 Predictor. String. Values: "Alpha", "Beta", "Gamma"

cat2 Predictor. String. Values: "North", "South", "East", "West"

Examples

```
data(data_test1)
lm(y ~ x1 + cat2 + cat1 + cat2:cat1, data_test1)
```

`hierarchical` *Check Models Hierarchy*

Description

Check a list of 'lm' objects to see whether they are be ordered in a way for doing hierarchical regression analysis.

Usage

```
hierarchical(...)
```

Arguments

`...`	The outputs of `lm()`, that is, one or more `lm`-class objects. The outputs of other model fitting functions may also be used, but should be used with cautions. Please refer to the "How it works" section in "Details."

Details

Two models can be compared by hierarchical regression analysis if one model can be formed by adding one or more terms to the other model.

This function checks whether a list of `lm` outputs can be ordered from the simplest model to the most complex model, with a more complex model formed by adding one or more terms to a simpler model.

How it works:

It extracts the terms in each model by `stats::terms()` and then extracts the labels of the terms by `labels()`. The labels are then used to determine the hierarchical order.

Therefore, in principle, this function can be used for the outputs of other model fitting functions as long as their outputs support the `stats::terms()` and the labels can be used to determine hierarchical order of two models.

Value

If the models can be ordered in a hierarchical way, the output is a list of the original `lm` outputs, sorted from the model with the smallest number of terms to the model with the largest number of terms. If the models cannot be ordered this way, `NA` is returned.

Author(s)

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

See Also

`stats::lm()`

Examples

```
dat <- data_test1
lm1 <- lm(y ~ x1 + x2, dat)
lm2 <- lm(y ~ x1 + x2 + x3 + x4, dat)
lm3 <- lm(y ~ x1 + cat1 + cat2 + x2 + x3 + x4, dat)
lm4 <- lm(y ~ x1 + x2*x3 + x4, dat)

# The order of entry does not matter
hierarchical(lm1, lm4, lm2)

# The following three models yield NA
hierarchical(lm3, lm4, lm2)
```

`hierarchical_lm` *Hierarchical Regression Analysis*

Description

Do hierarchical regression analysis on two or more models fitted by 'lm()'.

Usage

```
hierarchical_lm(...)
```

Arguments

`...`	The outputs of `lm()`, that is, one or more `lm`-class objects. The outputs of other model fitting functions may also be used, but should be used with cautions. Please refer to the "How it works" section in "Details." It also supports the output of `many_lm()`, and can mix the outputs of `many_lm()` with those of `lm()`.

Details

It conducted hierarchical regression analysis on two or more models fitted by `stats::lm()`. The models must be able to be ordered from the simplest to the most complex, with each more complex model formed by adding one or more terms to the simpler model.

ANOVA will be conducted to compare each model with the next more complex model in the order, with R-squared change computed.

Value

If the models can be ordered in a hierarchical way, the output is an ANOVA table with the R-squared estimate of each model, and the R-squared change of each model compared to the simpler model preceding this model in the order. The class of the output is `hierarchical_lm`, with a print method. If the models cannot be ordered this way, `NA` is returned.

How it works:

It call `hierarchical()` firsts to order the outputs for `stats::lm()`, If they can be ordered in a hierarchical way, they will be passed to `stats::anova()`. R-squared and R-squared change will be computed if they are available in the `summary()` method applied to each model.

Therefore, in principle, this function can also be used for the outputs of other model fitting functions if their outputs have `stats::anova()` and `summary()` methods.

Check Datasets Used:

The comparison is meaningful only if all models are fitted to the same datasets. There is not way to guarantee this is the case, given only the output of `lm()`. However, there are necessary conditions to claim that the same datasets are used: the number of cases are the same, the means, variances, and covariances of numerical variables, and the frequency distributions of variables common to two models are identical. If at least one of these conditions is not met, then two models must have been fitted to two different datasets.

The function will check these conditions and raise an error if any of these necessary conditions are not met.

Author(s)

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

See Also

`stats::lm()`, `hierarchical()`

Examples

```
dat <- data_test1
lm1 <- lm(y ~ x1 + x2, dat)
lm2 <- lm(y ~ x1 + x2 + x3 + x4, dat)
lm3 <- lm(y ~ x1 + cat1 + cat2 + x2 + x3 + x4, dat)
lm4 <- lm(y ~ x1 + x2*x3 + x4, dat)

hierarchical_lm(lm1, lm3, lm2)
hierarchical_lm(lm1, lm2, lm4)
# The following models will yield an error message:
tryCatch(hierarchical_lm(lm1, lm3, lm2, lm4), error = function(e) e)
```

---

lm_list_to_partable *Convert a 'lm_list' Object To a Parameter Table*

Description

Convert the output of `many_lm()` to a `lavann`-style parameter table.

Usage

```
lm_list_to_partable(
  object,
  keep_intercepts = FALSE,
  vcov_args = list(),
  pvalue_fun = NULL,
  rsquare = FALSE,
  ci = FALSE,
  ci_fun = stats::confint,
  ci_args = list(level = 0.95)
)
```

Arguments

object	The output of many_lm() or manymome::lm2list().
keep_intercepts	Logical. If TRUE, the intercepts of the regression models and the means of the "pure" predictors (variables not being the outcome variables of any of the regression models) are kept in the parameter table. If FALSE, the default, all intercepts and means will be removed.
vcov_args	A named list of arguments to be passed to stats::vcov() when computing the standard errors of the regression coefficients. Default is list(), an empty list.
pvalue_fun	The function to be used to compute the *p*-values of regression coefficients. Ignored for now. Included for adding this feature in the future.
rsquare	Logical. Whether R-squares will be included in the output, with r2 as the operator in the column op. Default is FALSE. Not included by default because semPlot::semPaths() will draw the R-squares over the residual variances.
ci	Logical. If TRUE, confidence intervals will be added, computed by stats::confint().
ci_fun	The function to be used to form the confidence intervals for regression coefficients. Default is stats::confint
ci_args	A named list of arguments to be passed to ci_fun. Default is list(level = .95), requesting 95% confidence intervals.

Details

This function convert a a lit of lm objects, such as the output of many_lm() or manymome::lm2list(), to a table of parameter estimates similar to the output of lavaan::parameterTable.

The output is designed to be used by semPlot::semPaths() and so contains only information necessary for the plot.

The output of stats::lm() is already supported by semPlot::semPaths(), and it can also combine a list of regression models into on single plot. However, it will convert interaction terms to knots. Moreover, if two interaction terms in two different models share the a variable, it will be incorrectly combined to become a single knot (Version 1.1.6). Therefore, this function was developed to let users to draw the model as if it were a path model in structural equation modeling.

Value

A data frame object with columns such as `lhs`, `op`, `rhs`, and `est`, major columns of the output of `lavaan::parameterTable()` necessary for plotting the model using `semPlot::semPaths()`.

Author(s)

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

See Also

`many_lm()` and `manymome::lm2list()`.

Examples

```
data(data_test1)
mod <- "x3 ~ x2*x1
        x4 ~ x3
        x5 ~ x4 + x3"
out <- many_lm(mod, data_test1)
out_ptable <- lm_list_to_partable(out)
out_ptable

m <- matrix(c("x1", "x2", "x2:x1", NA, "x3", NA, "x4", NA, NA, NA, "x5", NA),
            nrow = 3, ncol = 4)
m

# The output can be used by semPlot::semPaths()

if (requireNamespace("semPlot", quietly = TRUE)) {
  library(semPlot)
  p <- semPaths(out_ptable,
                what = "paths",
                whatLabels = "est",
                nCharNodes = 0,
                style = "ram",
                layout = m,
                exoCov = FALSE,
                DoNotPlot = TRUE)
  plot(p)

  # If it is desired to use knots to
  # denote interaction terms, then,
  # the output of many_lm() can be used
  # directly.

  m2 <- matrix(c("x1", NA, "x2", NA, "x3", NA, "x4", NA, NA, NA, "x5", NA),
               nrow = 3, ncol = 4)
  p2 <- semPaths(out,
                 what = "paths",
                 whatLabels = "est",
                 nCharNodes = 0,
                 style = "ram",
```

```
                layout = m2,
                exoCov = FALSE,
                intercepts = FALSE,
                DoNotPlot = TRUE)
  plot(p2)

  # This illustrates the problem with using
  # the list of lm-outputs directly when
  # a variable is involved in the interaction terms
  # of two or more models.

  m3 <- matrix(c("x2",   NA, "x1",   NA, "x3",
                   NA,   NA,   NA,   NA,   NA,
                   NA, "x4",   NA, "x5",   NA),
           nrow = 5, ncol = 3)
  mod3 <- "x4 ~ x2*x1
           x5 ~ x3*x1"
  out3 <- many_lm(mod3, data_test1)
  p3 <- semPaths(out3,
                what = "paths",
                whatLabels = "est",
                nCharNodes = 0,
                style = "ram",
                layout = m3,
                exoCov = FALSE,
                intercepts = FALSE,
                DoNotPlot = TRUE)
  plot(p3)

}
```

---

many_lm *Fit Linear Models Defined By Model Syntax*

Description

Fit a list of linear models defined by model syntax.

Usage

```
many_lm(models, data, na_omit_all = TRUE, ...)
```

Arguments

models	Character. Model syntax. See Details.
data	The data frame. Must be supplied if na_omit_all is TRUE. If na_omit_all is FALSE, it can be omitted (though not suggested).

na_omit_all	How missing data is handled across models. If TRUE, the default, then only cases with no missing data on all variables used at least one of the models will be retained (i.e., listwise deletion). If FALSE, then missing data will be handled in each model separately by lm().
...	Additional arguments. To be passed to lm().

Details

This function extracts linear model formulas from a model syntax (a character vector), fits each of them by lm(), and stores the results in a list.

Lines with the first non-whitespace character "#" are treated as comments and ignored.

Each line must be a valid formula for lm().

Listwise deletion:

If na_omit_all is TRUE, the default, then cases with missing data on at least one of the variables used in the model will be removed. Each call to lm() will have subset set to an integer vector of cases *not* removed (i.e., cases retained)

Handling the subset argument:

If subset is used when calling this function, it will also be used to select cases.

Note that the subset argument in the call in each model will be replaced by a numeric vector of cases retained, determined by both missing data and the original value of the subset.

Value

A list of the output of lm(). The class is lm_list_lmhelprs.

Author(s)

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

See Also

stats::lm()

Examples

```
data(data_test1)
mod <- "x3 ~ x2 + x1
        x4 ~ x3
        x5 ~ x4*x1"
out <- many_lm(mod, data_test1)
summary(out)
```

print.hierarchical_lm *Print a* hierarchial_lm *Class Object*

Description

Print the content of a 'hierarchical_lm'-class object.

Usage

```
## S3 method for class 'hierarchical_lm'
print(
  x,
  digits = 4,
  signif.stars = getOption("show.signif.stars"),
  eps.Pvalue = 0.001,
  ...
)
```

Arguments

x	A hierarchical_lm-class object, usually the output of hierarchical_lm().
digits	The minimum number of significant digits to be used for most numbers. To be used by the print method of anova-class objects.
signif.stars	Logical. To be used by the print method of anova-class objects.
eps.Pvalue	To be passed to format.pval(). It controls how small *p*-values are displayed. Default is .001. That is, *p*-values less than .001 will be displayed as <.001.
...	Optional arguments. To be passed to the print method of anova-class objects.

Details

The printout is very similar to that of the print method of an anova object. It simply overrides the default values for some arguments, notably esp.Pvalue to prevent small *p*-values to be presented in scientific notation.

Value

x is returned invisibly. Called for its side effect.

Author(s)

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

See Also

hierarchical_lm()

Examples

```
dat <- data_test1
lm1 <- lm(y ~ x1 + x2, dat)
lm2 <- lm(y ~ x1 + x2 + x3 + x4, dat)
lm3 <- lm(y ~ x1 + cat1 + cat2 + x2 + x3 + x4, dat)
lm4 <- lm(y ~ x1 + x2*x3 + x4, dat)

hierarchical_lm(lm1, lm3, lm2)
hierarchical_lm(lm1, lm2, lm4)
```

---

print.lm_list_lmhelprs

Print an lm_list_lmhelprs*-Class Object*

Description

Print the content of the output of many_lm().

Usage

```
## S3 method for class 'lm_list_lmhelprs'
print(x, ...)
```

Arguments

x	The output of many_lm().
...	Other arguments. Not used.

Details

Adapted from the package manymome such that many_lm() can be used with manymome.

Value

x is returned invisibly. Called for its side effect.

Examples

```
data(data_test1)
mod <- "x3 ~ x2 + x1
        x4 ~ x3
        x5 ~ x4*x1"
out <- many_lm(mod, data_test1)
out
```

summary.lm_list_lmhelprs *Summary of an* lm_list_lmhelprs*-Class Object*

Description

The summary of content of the output of many_lm().

Usage

```
## S3 method for class 'lm_list_lmhelprs'
summary(object, ...)

## S3 method for class 'summary_lm_list_lmhelprs'
print(x, digits = 3, ...)
```

Arguments

object	The output of many_lm().
...	Other arguments. Not used.
x	An object of class summary_lm_list_lmhelprs.
digits	The number of significant digits in printing numerical results.

Value

summary.lm_list_lmhelprs() returns a summary_lm_list_lmhelprs-class object, which is a list of the summary() outputs of the lm() outputs stored.

print.summary_lm_list_lmhelprs() returns x invisibly. Called for its side effect.

Adapted from the package manymome such that many_lm() can be used without manymome.

Functions

- print(summary_lm_list_lmhelprs): Print method for output of summary for lm_list_lmhelprs.

Examples

```
data(data_test1)
mod <- "x3 ~ x2 + x1
        x4 ~ x3
        x5 ~ x4*x1"
out <- many_lm(mod, data_test1)
summary(out)
```

test_highest *Test the Highest Order Term by ANOVA*

Description

Identify the highest order terms in a model fitted by 'lm()', and compare this model to a model with this term removed using ANOVA.

Usage

```
test_highest(lm_out)

highest_order(lm_out)
```

Arguments

lm_out	The output of stats::lm().

Details

The function test_highest() first check if a model fitted by stats::lm() has a unique highest order term (e.g., the term x1:x2, in the model y ~ x1 + x2 + x1:x2). If yes, it will fit a model with this term removed, and then call hierarchical_lm() to compare the original model with this reduced model.

If the model does not have a unique highest order term, an error will be raised.

Value

A hierarchical_lm-class object, which is the output of hierarchical_lm(). Two models are compared, the original model and the model with the unique highest order term removed.

Functions

- test_highest(): Test the highest order term.
- highest_order(): Find the highest order term.

Limitation

It relies on terms created by stats::lm() to determine the order of each term. If a higher order term is created manually (e.g., I(x1 * x2)), then it cannot know that this term is a second order term.

Author(s)

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

See Also

hierarchical_lm()

Examples

```
dat <- data_test1

lm1 <- lm(y ~ x1 + x2 + cat1*x3, dat)
lm2 <- lm(y ~ x1 + x2*x3 + x4, dat)

test_highest(lm1)
test_highest(lm2)

highest_order(lm1)
highest_order(lm2)

# The followings will yield an error

lm3 <- lm(y ~ x1 + x2 + x3, dat)
summary(lm3)
tryCatch(test_highest(lm3), error = function(e) e)
tryCatch(highest_order(lm3), error = function(e) e)

lm4 <- lm(y ~ x1 + x2*x3 + x4*x5, dat)
summary(lm4)
tryCatch(test_highest(lm4), error = function(e) e)
tryCatch(highest_order(lm4), error = function(e) e)
```

Index

∗ **datasets**
data_test1, 2

data_test1, 2

hierarchical, 3
hierarchical(), *5*
hierarchical_lm, 4
hierarchical_lm(), *10*, *13*, *14*
highest_order *(*test_highest*)*, 13

labels(), *3*
lavaan::parameterTable, *6*
lavaan::parameterTable(), *7*
lm(), *4*, *5*, *9*, *12*
lm_list_to_partable, 5

many_lm, 8
many_lm(), *4*, *6*, *7*, *11*, *12*
manymome::lm2list(), *6*, *7*

print.hierarchical_lm, 10
print.lm_list_lmhelprs, 11
print.summary_lm_list_lmhelprs
*(*summary.lm_list_lmhelprs*)*, 12
print.summary_lm_list_lmhelprs(), *12*

semPlot::semPaths(), *6*, *7*
stats::anova(), *5*
stats::confint(), *6*
stats::lm(), *3–6*, *9*, *13*
stats::terms(), *3*
stats::vcov(), *6*
summary(), *5*, *12*
summary.lm_list_lmhelprs, 12
summary.lm_list_lmhelprs(), *12*

test_highest, 13
test_highest(), *13*