Package 'paramhetero'

January 8, 2020

Title Numeric and Visual Comparisons of Heterogeneity in Parametric Models

Version 0.2.0

Description Performs statistical tests to compare coefficients and residual variance across multiple models. Also provides graphical methods for assessing heterogeneity in coefficients and residuals. Currently supports linear and generalized linear models, as well as their multi-level and complex survey-weighted variations from the 'lme4' and 'survey' packages, respectively. Reference: Li (2015) <https://scholarscompass.vcu.edu/etd/3985/>.

Depends R (>= 3.5.0)

Imports ggplot2, ggpubr, stats, lme4, survey

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

NeedsCompilation no

Author Travis Loux [aut, cre], Cara Wiskow [aut]

Maintainer Travis Loux <travis.loux@slu.edu>

Repository CRAN

Date/Publication 2020-01-08 18:30:02 UTC

R topics documented:

ci_matrix 2
coefficient_anova 3
coefficient_forestplot 4
coefficient_manova 5
coef_welch 6
make_resid_df 7
parametric_heterogeneity 8
residual_levene 9
residual_plots 10

Index **12**

`ci_matrix` *Create confidence intervals for all coefficients.*

Description

Create confidence intervals for all coefficients across models, excluding the intercept.

Usage

```
ci_matrix(model_list, model_names = NULL, levels = c(0.95, 0.5))
```

Arguments

`model_list`	A list of regression models.
`model_names`	A list of names for the regression models (default is `NULL`).
`levels`	At most two confidence levels.

Details

This function is called from `coefficient_forestplot` and might not be of direct use for most users. Confidence intervals are obtained through `confint` in the `stats` package. Intercepts are excluded from the analysis as they don't relate to interactions or "effect modification," but would be most analogous to an average effect.

Value

A data frame of model coefficients with their respective confidence intervals.

Examples

```
states = as.data.frame(state.x77)

m1 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='Northeast')
m2 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='South')
m3 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='North Central')
m4 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='West')

mList = list(m1, m2, m3, m4)

paramhetero:::ci_matrix(model_list = mList, levels = 0.95)
```

`coefficient_anova` *Compare coefficients of all shared predictors across models via ANOVA.*

Description

Compare coefficients of all shared predictors across models via ANOVA using `coef_welch`.

Usage

```
coefficient_anova(model_list, model_names = NULL, padj = p.adjust.methods)
```

Arguments

`model_list`	A list of regression models.
`model_names`	A list of names for the regression models (default is `NULL`).
`padj`	Adjustment of p-values for multiple comparisons. Value must come from `p.adjust.methods`.

Details

For each predictor variable in the models, runs `coef_welch` to test for equality of regression parameters across models.

Value

Data frame of coefficient information. Data frame includes coefficient estimates for each model as well as results from ANOVA test comparing coefficients across models.

See Also

`coef_welch`

Examples

```
states = as.data.frame(state.x77)

m1 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='Northeast')
m2 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='South')
m3 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='North Central')
m4 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='West')

mList = list(m1, m2, m3, m4)

coefficient_anova(model_list = mList,
                  model_names = c('Northeast', 'South',
```

```
                                    'North Central', 'West'))
```

`coefficient_forestplot` *Create forest plot of model coefficients with confidence intervals.*

Description

Create a ggplot forest plot of model coefficients with confidence intervals.

Usage

```
coefficient_forestplot(
  model_list,
  model_names = NULL,
  levels = c(0.95, 0.5),
  horiz = TRUE
)
```

Arguments

`model_list`	A list of regression models.
`model_names`	A list of names for the regression models (default is `NULL`).
`levels`	A list of levels of confidence for confidence intervals (max 2 levels).
`horiz`	Toggle whether confidence intervals are displayed horizontally or vertically. Default is set to `TRUE`.

Details

The forest plot groups variables along the axis determined by the `horiz` parameter and colors the data by model. If `model_names = NULL`, the default, models are numbered sequentially in the order they appear in `model_list` (Model 1, Model 2, Model 3, etc.). If two confidence levels are provided the shorter interval is drawn with a thicker line.

Value

A ggplot object to compare model coefficient estimates with their corresponding confidence interval(s), grouped by coefficient.

Examples

```
states = as.data.frame(state.x77)

m1 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='Northeast')
m2 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
```

```
        subset=state.region=='South')
m3 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='North Central')
m4 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='West')

mList = list(m1, m2, m3, m4)

coefficient_forestplot(model_list = mList,
                       model_names =c('Northeast', 'South',
                                      'North Central', 'West'),
                       horiz = FALSE)
```

`coefficient_manova` *Compare coefficient vectors using MANOVA.*

Description

Compare coefficient vectors using MANOVA across multiple models.

Usage

```
coefficient_manova(model_list)
```

Arguments

`model_list`	A list of regression models.

Details

Removes intercept from each coefficient vector, then performs a MANOVA analysis to compare remaining regression coefficients across models. Specifically, for models with matrix notation

$$\mathbf{Y}_m = \beta_{m0} + \mathbf{X}_m\mathbf{B}_m$$

for $i = 1, 2, ..., M$, test the null hypothesis

$$\mathbf{B}_1 = \mathbf{B}_2 = ... = \mathbf{B}_m$$

against the alternative hypothesis that not all coefficient vectors are equal.

Value

Numeric vector of MANOVA test results. This includes Wilk's lambda, MANOVA F-statistic, numerator degrees of freedom, denominator degrees of freedom, and the associated p-value.

Examples

```
states = as.data.frame(state.x77)

m1 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='Northeast')
m2 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='South')
m3 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='North Central')
m4 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='West')

mList = list(m1, m2, m3, m4)

coefficient_manova(model_list = mList)
```

`coef_welch` *Compare single regression coefficient across models.*

Description

Compare the regression coefficients of a predictor across models.

Usage

```
coef_welch(bs, vars, ns, p)

coef_anova(bs, vars, ns, p)
```

Arguments

`bs`	Vector of regression coefficients.
`vars`	Vector of variances of estimates in `bs`.
`ns`	Vector of sample sizes for each model.
`p`	Single integer, number of shared predictors in models.

Details

Both `coef_anova` and `coef_welch` compare the coefficient of a predictor across multiple models, testing the null hypothesis that all parameters are equal. `coef_anova` assumes homogeneity across models, while `coef_welch` allows for heterogeneity in the natural variation. Currently, only `coef_welch` is implemented as it is a slightly more generally applicable test. An option for `coef_anova` may appear in later versions.

For example, given three models

$$y_i = \beta_{10} + \beta_{11} x_{i1} + \beta_{12} x_{i2} + \epsilon_{1i}$$

$$y_i = \beta_{20} + \beta_{21} x_{i1} + \beta_{22} x_{i2} + \epsilon_{2i}$$

$$y_i = \beta_{30} + \beta_{31} x_{i1} + \beta_{32} x_{i2} + \epsilon_{3i}$$

Would test hypotheses $H_0 : \beta_{11} = \beta_{21} = \beta_{31}$ against the alternative that not all three parameters for x_1 are equal. (Or similarly for the parameters of x_2.)

This is the workhorse function for `coefficient_anova`.

Value

Vector containing the resulting F statistic, numerator degrees of freedom, denominator degrees of freedom, and the p-value for the test of equality.

References

Liu, H. "Comparing Welch's ANOVA, a Kruskal-Wallis Test and Traditional ANOVA in Case of Heterogeneity of Variance." Master's Thesis. Available at https://scholarscompass.vcu.edu/etd/3985/.

Examples

```
coefs = c(-0.92, -1.13, -6.91, -0.09)
ses = c(1.44, 0.57, 1.65, 0.67)
grp_n = c(9, 16, 12, 13)

coef_welch(bs = coefs, vars = ses^2, ns = grp_n, p=2)
```

---

make_resid_df *Extract residuals from multiple models.*

Description

Extracts residuals from multiple models and returns them as a data frame.

Usage

```
make_resid_df(model_list, model_names = NULL)
```

Arguments

`model_list`	A list of regression models.
`model_names`	A list of names for the regression models (default is `NULL`).

Details

This function is used within `residual_plots` and is not likely to be of much use to end users.

Value

A data frame of residuals and their corresponding model names.

Examples

```
states = as.data.frame(state.x77)

m1 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='Northeast')
m2 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='South')
m3 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='North Central')
m4 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='West')

mList = list(m1, m2, m3, m4)

paramhetero:::make_resid_df(model_list = mList)
```

`parametric_heterogeneity` *Test coefficient and residual heterogeneity across models.*

Description

An umbrella function to assess heterogeneity across functions.

Usage

```
parametric_heterogeneity(model_list)
```

Arguments

`model_list`	A list of regression models.

Details

`parametric_heterogeneity` is used to run and format results from `coefficient_manova`, `coefficient_anova`, and `residual_levene`, the last for `lm` models only. Please see those functions for more detail.

Value

Data frame containing name of items tested for heterogeneity, F-statistics for homogeneity tests, numerator degrees of freedom, denominator degrees of freedom, and the associated p-value.

See Also

- coefficient_manova
- coefficient_anova
- residual_levene

Examples

```
states = as.data.frame(state.x77)

m1 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='Northeast')
m2 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='South')
m3 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='North Central')
m4 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='West')

mList = list(m1, m2, m3, m4)

parametric_heterogeneity(model_list = mList)
```

`residual_levene` *Compare regression residual standard deviation across multiple models.*

Description

Compare regression residual standard deviation across multiple `lm` models using Levene's test.

Usage

```
residual_levene(model_list, model_names = NULL)
```

Arguments

`model_list`	A list of regression models.
`model_names`	A list of names for the regression models (default is `NULL`).

Details

Levene's test is often suggested as a diagnostic check for homoskedasticity in ANOVA analyses. This function works for linear models (`lm`) only.

Value

Numeric vector of Levene test results. This includes an F-statistic, numerator degrees of freedom, denominator degrees of freedom, and the associated p-value.

Examples

```
states = as.data.frame(state.x77)

m1 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='Northeast')
m2 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='South')
m3 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='North Central')
m4 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='West')

mList = list(m1, m2, m3, m4)

residual_levene(model_list = mList)
```

---

residual_plots *Plot residuals.*

Description

Plots residuals of multiple models as density plots and boxplots.

Usage

```
residual_plots(model_list, model_names = NULL, bw = "nrd0", thm = NULL)

residual_density(model_list, model_names = NULL, bw = "nrd0")

residual_boxplot(model_list, model_names = NULL)
```

Arguments

model_list	A list of regression models.
model_names	A list of names for the regression models (default is NULL).
bw	Bandwidth for density plots (default is "nrd0").
thm	A ggplot2 theme for the output plots.

Details

residual_plots plots density plots and boxplots in a two row, single column format. residual_density and residual_boxplots plot the component pieces. If model_names = NULL, the default, models are numbered sequentially in the order they appear in model_list (Model 1, Model 2, Model 3, etc.).

Value

A residual plot for inputted regression models.

Examples

```
states = as.data.frame(state.x77)

m1 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='Northeast')
m2 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='South')
m3 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='North Central')
m4 = lm(`Life Exp` ~ Income + Illiteracy, data=states,
        subset=state.region=='West')

mList = list(m1, m2, m3, m4)

residual_plots(model_list = mList, thm=ggplot2::theme_minimal())
```

Index

ci_matrix, 2
coef_anova *(*coef_welch*)*, 6
coef_welch, *3*, 6
coefficient_anova, 3, *7–9*
coefficient_forestplot, *2*, 4
coefficient_manova, 5, *8, 9*

lm, *9*

make_resid_df, 7

p.adjust.methods, *3*
parametric_heterogeneity, 8

residual_boxplot *(*residual_plots*)*, 10
residual_density *(*residual_plots*)*, 10
residual_levene, *8, 9*, 9
residual_plots, 10