Package: paramhetero (via r-universe)

March 3, 2025

Title Numeric and Visual Comparisons of Heterogeneity in Parametric Models

Version 1.2.0

Description Performs statistical tests to compare coefficients and residual variance across models. Also provides graphical methods for assessing heterogeneity in coefficients and residuals. Currently supports linear models, generalized linear models, and their survey-weighted analogues.

Depends R (>= 4.0.0)

Imports ggplot2, MASS, stats, survey

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.2

Config/pak/sysreqs make

Repository https://tloux.r-universe.dev

RemoteUrl https://github.com/tloux/paramhetero

RemoteRef HEAD

RemoteSha 064b3d331108161ed3acea67c8e17c212e9e28d4

Contents

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,
  varlabs = NULL,
  conflevel = 0.95,
  horiz = TRUE
)
```

Arguments

`model_list`	A list of regression models.
`model_names`	A list of names for the regression models.
`varlabs`	A vector of labels for model variables.
`conflevel`	Confidence level for intervals.
`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.).

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)
```

---

compare_coefs *Compare shared coefficients across models*

Description

Compares predictor coefficients across models.

Usage

```
compare_coefs(model_list, padj = "none")
```

Arguments

model_list	A list of regression models.
padj	A method from `p.adjust.methods` for adjusting coefficient p-values for multiple testing.

Details

This function currently supports comparing coefficients from two models. For each model predictor, coefficients are compared across models. P-values come from a two-sided alternative hypothesis. They can, and should, be adjusted for multiple testing to reduce the probability of chance significant findings.

Value

Data frame of shared coefficients, the difference between them, the standard error of the difference, the test statistic comparing them, and the p-value adjusted using the method provided in `padj`.

Examples

```
##Simulate data

N = 500

m = rep(1:2, each=N)
```

```
x1 = rnorm(n=N*2)
x2 = rnorm(n=N*2)
x3 = rnorm(n=N*2)

y = x1 + x2 + x3 + rnorm(n=N*2)

dat = data.frame(m, x1, x2, x3, y)

m1 = lm(y ~ x1 + x2 + x3, data=dat, subset=m==1)
m2 = lm(y ~ x1 + x2 + x3, data=dat, subset=m==2)

mList = list(m1, m2)

compare_coefs(model_list = mList, padj='fdr')
```

`compare_coef_vectors` *Compare coefficient vectors across models*

Description

Compare coefficient vectors, after removing intercept, across multiple models.

Usage

```
compare_coef_vectors(model_list)
```

Arguments

`model_list`	A list of regression models.

Details

This function currently supports comparing coefficient vectors from two models. The intercepts of the models are removed, if they exist, and the coefficient vectors are compared by Hotelling's T^2 test. This can be considered as an initial omnibus test for differences among the coefficients before searching through all coefficients for individual differences using, for example, `compare_coefs`.

Value

List of test results. This includes the chi-squared statistic, degrees of freedom, and p-value.

Examples

```
##Simulate data

N = 500

m = rep(1:2, each=N)
```

```
x1 = rnorm(n=N*2)
x2 = rnorm(n=N*2)
x3 = rnorm(n=N*2)

y = x1 + x2 + x3 + rnorm(n=N*2)

dat = data.frame(m, x1, x2, x3, y)

m1 = lm(y ~ x1 + x2 + x3, data=dat, subset=m==1)
m2 = lm(y ~ x1 + x2 + x3, data=dat, subset=m==2)

mList = list(m1, m2)

compare_coef_vectors(model_list = mList)
```

---

compare_resids *Compare regression residual standard deviation across models*

Description

Compare residual standard deviation across models. Works for linear regression (`lm`) only.

Usage

```
compare_resids(model_list)
```

Arguments

`model_list`	A list of regression models.

Details

This function currently supports comparing residual standard deviation from two models. Residuals are assumed to be normally distributed (as also assumed in the linear model itself) and are compared by an F test.

Value

Vector of results. This includes the residual standard deviation from each model, the F statistic comparing the standard deviations, the numerator and denominator degrees of freedom, and the p-value.

Examples

```
##Simulate data

N = 500

m = rep(1:2, each=N)

x1 = rnorm(n=N*2)
x2 = rnorm(n=N*2)
x3 = rnorm(n=N*2)

y = x1 + x2 + x3 + rnorm(n=N*2)

dat = data.frame(m, x1, x2, x3, y)

m1 = lm(y ~ x1 + x2 + x3, data=dat, subset=m==1)
m2 = lm(y ~ x1 + x2 + x3, data=dat, subset=m==2)

mList = list(m1, m2)

compare_resids(model_list = mList)
```

Index