Package ‘pubh’

October 14, 2022

Title    A Toolbox for Public Health and Epidemiology
Version  1.2.7
Maintainer Josie Athens <josie.athens@gmail.com>
Description A toolbox for making R functions and capabilities more accessible to students and professionals from Epidemiology and Public Health related disciplines. Includes a function to report coefficients and confidence intervals from models using robust standard errors (when available), functions that expand ‘ggplot2’ plots and functions relevant for introductory papers in Epidemiology or Public Health. Please note that use of the provided data sets is for educational purposes only.
Depends   R (>= 4.1.0), emmeans, ggformula, gtsummary, huxtable, magrittr, stats
Imports   car, dplyr, Epi, epiR, epitools, ggplot2, Hmisc, lmtest, moonBook, sandwich, sjlabelled, sjmisc, sjPlot, survival, tibble, tidyselect
Suggests  broom, effectsize, ggeffects, ggfortify, ggsci, ISwR, jtools, knitr, latex2exp, lme4, MASS, nlme, nnet, parameters, performance, report, rmarkdown, rms, rstatix, see, tidyverse
License   GPL-2
Encoding  UTF-8
LazyData  true
RoxygenNote 7.1.2
VignetteBuilder knitr
BugReports https://github.com/josie-athens/pubh/issues
NeedsCompilation no
Author    Josie Athens [aut, cre],
         Frank Harell [ctb],
         John Fox [ctb],
         R-Core [ctb]
Repository CRAN
Date/Publication 2022-04-04 13:50:02 UTC
R topics documented:

axis_labs .................................................. 3
bar_error .................................................. 4
Bernard ..................................................... 5
bland_altman .............................................. 6
box_plot ................................................... 8
Brenner .................................................... 9
bst .......................................................... 10
chisq.fisher .............................................. 10
coeff_det .................................................. 11
contingency .............................................. 12
contingency2 ............................................. 13
cosm_reg .................................................. 14
cosm_sum .................................................. 15
cross_tab .................................................. 16
cross_tbl .................................................. 17
diag_test .................................................. 19
diag_test2 ............................................... 20
estat ...................................................... 20
expand_df ............................................... 21
Fentress ................................................... 22
freq_cont .................................................. 23
gen_bst_df ............................................... 24
geo_mean .................................................. 25
get_r2 ..................................................... 25
gf_star .................................................... 26
glm_coef .................................................. 28
harm_mean ................................................ 29
hist_norm .................................................. 30
Hodgkin ................................................... 31
inv_logit .................................................. 32
jack_knife ............................................... 32
Kirkwood .................................................. 33
knife_mean ............................................... 34
leverage ................................................... 34
logistic_gof ............................................. 35
Macmahon .................................................. 36
mhor ....................................................... 36
model_labels ............................................. 37
multiple ................................................... 38
odds_trend .............................................. 40
Oncho ..................................................... 41
predict_inv ............................................. 42
prop_or .................................................... 43
pseudo_r2 ............................................... 44
qq_plot .................................................... 44
rank_influence ......................................... 46
**axis_labs**  

Apply labels from variables to axis-labels in plots.

Description  

`axis_labs` takes labels from labelled data to use them as axis-labels for plots generated by `gformula` or `ggplot2`.

Usage  

```r  
axis_labs(object)  
```

Arguments  

- **object**: ggplot2 object (see examples).

Details  

This function is helpful when data has been already labelled by `sjlabelled`. It retrieves variable labels and uses them for plotting.

Value  

A ggplot2 object.
Examples

data(kfm, package = "ISwR")
require(sjlabelled, quietly = TRUE)
kfm <- kfm %>%
  var_labels(
    dl.milk = "Breast-milk intake (dl/day)",
    sex = "Sex",
    weight = "Child weight (kg)",
    ml.suppl = "Milk substitute (ml/day)",
    mat.weight = "Maternal weight (kg)",
    mat.height = "Maternal height (cm)"
  )

kfm %>%
gf_point(weight ~ dl.milk) %>%
gf_lm(col = 2, interval = "confidence", col = 2) %>%
axis_labs()

kfm %>%
  box_plot(dl.milk ~ sex, fill = "thistle", alpha = 0.8) %>%
  axis_labs() %>%
gf_star(x1 = 1, y1 = 10.9, x2 = 2, y2 = 11, y3 = 11.2)

---

bar_error

Bar charts with error bars.

Description

date_error constructs bar charts with error bars showing 95 confidence intervals around mean values. High of bars represent mean values.

Usage

bar_error(
  object = NULL,
  formula = NULL,
  data = NULL,
  fill = "indianred3",
  col = "black",
  alpha = 0.7,
  ...
)

Arguments

object When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.

formula A formula with shape: y ~ x or y ~ x | z where y is a numerical variable and both x and z are factors.
Bernard

data A data frame where the variables in the formula can be found.
fill Colour used to fill the bars.
col Colour used for the borders of the bars.
alpha Opacity of the colour fill (0 = invisible, 1 = opaque).
... Additional information passed to `gf_summary`.

Examples

```r
require(dplyr, quietly = TRUE)
require(sjlabelled, quietly = TRUE)
data(birthwt, package = "MASS")
birthwt <- birthwt %>%
  mutate(
    smoke = factor(smoke, labels = c("Non-smoker", "Smoker")),
    Race = factor(race > 1, labels = c("White", "Non-white"))
  ) %>%
  var_labels(
    bwt = "Birth weight (g)",
    smoke = "Smoking status"
  )
birthwt %>%
  bar_error(bwt ~ smoke, fill = "plum3")
birthwt %>%
  bar_error(bwt ~ smoke | Race, fill = "plum3")
birthwt %>%
  bar_error(bwt ~ smoke, fill = ~ Race)
```

Survival of patients with sepsis.

Description

A randomised, double-blind, placebo-controlled trial of intravenous ibuprofen in 455 patients who had sepsis, defined as fever, tachycardia, tachypnea, and acute failure of at least one organ system.

Usage

Bernard

Format

A labelled tibble with 455 rows and 9 variables:

id Patient ID
**treat**  Treatment, factor with levels "Placebo" and "Ibuprofen".

**race**  Race/ethnicity, factor with levels "White", "African American" and "Other".

**fate**  Mortality status at 30 days, factor with levels "Alive" and "Dead".

**apache**  Baseline APACHE score.

**o2del**  Oxygen delivery at baseline.

**followup**  Follow-up time in hours.

**temp0**  Baseline temperature in centigrades.

**temp10**  Temperature after 36 hr in centigrades.

**Source**


**Examples**

```r
require(dplyr, quietly = TRUE)
data(Bernard)

Bernard %>%
  select(fate, treat) %>%
  cross_tbl(by = "fate")

contingency(fate ~ treat, data = Bernard)
```

---

**bland_altman**  
*Bland-Altman agreement plots.*

**Description**

Bland-Altman agreement plots.

**Usage**

```r
bland_altman(
  object = NULL,
  formula = NULL,
  data = NULL,
  pch = 20,
  size = 1,
  col = "black",
  transform = FALSE,
  ...
)
```
**bland_altman**

**Arguments**

- **object**: When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
- **formula**: A formula with shape: \( y \sim x \) (see details).
- **data**: A data frame where the variables in the formula can be found.
- **pch**: Symbol for plotting data.
- **size**: Size of the symbol using to plot data.
- **col**: Colour used for the symbol to plot data.
- **transform**: Logical, should ratios instead of difference be used to construct the plot?
- **...**: Further arguments passed to `gf_point`.

**Details**

`bland_altman` constructs Bland-Altman agreement plots.

Variables in `formula` are continuous paired observations. When the distribution of the outcome is not normal, but becomes normal with a log-transformation, `bland_altman` can plot the ratio between outcomes (difference in the log scale) by using option `transform = TRUE`.

**Examples**

```r
data(wright, package = "ISwR")
wright %>%
bland_altman(mini.wright ~ std.wright,
pch = 16,
ylab = "Large-mini expiratory flow rate (l/min)",
xlab = "Mean expiratory flow rate (l/min)"
) %>%
gf_labs(
y = "Large-mini expiratory flow rate (l/min)",
x = "Mean expiratory flow rate (l/min)"
) %>%
gf_theme(theme = sjPlot::theme_sjplot2(base_size = 9))
```

```r
data(Sharples)
Sharples %>%
bland_altman(srweight ~ weight, transform = TRUE) %>%
gf_labs(x = "Mean of weights (kg)", y = "Measured weight / Self-reported weight") %>%
gf_theme(theme = sjPlot::theme_sjplot2(base_size = 9))
```
Construct box plots.

Description

box_plot is a wrap function that calls gf_boxplot to construct more aesthetic box plots.

Usage

```r
box_plot(
  object = NULL,
  formula = NULL,
  data = NULL,
  fill = "indianred3",
  alpha = 0.7,
  outlier.shape = 20,
  outlier.size = 1,
  ...
)
```

Arguments

- **object**: When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
- **formula**: A formula with shape: \( y \sim x \) where \( y \) is a numerical variable and \( x \) is a factor.
- **data**: A data frame where the variables in the formula can be found.
- **fill**: Colour used for the box passed to gf_boxplot.
- **alpha**: Opacity (0 = invisible, 1 = opaque).
- **outlier.shape**: Shape (pch) used as symbol for the outliers.
- **outlier.size**: Size of the outlier symbol.
- **...**: Further arguments passed to gf_boxplot.

Examples

```r
data(kfm, package = "ISwR")
require(sjlabelled, quietly = TRUE)
kfm <- kfm %>%
  var_labels(
    dl.milk = "Breast-milk intake (dl/day)",
    sex = "Sex",
    weight = "Child weight (kg)",
    ml.suppl = "Milk substitute (ml/day)",
    mat.weight = "Maternal weight (kg)",
    mat.height = "Maternal height (cm)"
  )
```
Prevalence of Helicobacter pylori infection in preschool children.

Description

A data set containing the prevalence of Helicobacter pylori infection in preschool children according to parental history of duodenal or gastric ulcer.

Usage

Brenner

Format

A labelled tibble with 863 rows and 2 variables:

- **ulcer** History of duodenal or gastric ulcer, factor with levels "No" and "Yes".
- **infected** Infected with Helicobacter pylori, factor with levels "No" and "Yes".

Source


Examples

```r
require(dplyr, quietly = TRUE)
data(Brenner)

Brenner %>%
  select(infected, ulcer) %>%
cross_tbl(by = "infected")
contingency(infected ~ ulcer, data = Brenner, method = "cross.sectional")
```
## bst

**Bootstrap Confidence Intervals.**

### Description

bst estimates confidence intervals around the mean, median or geo_mean.

### Usage

bst(x, stat = "mean", n = 1000, CI = 95, digits = 2)

### Arguments

- **x**
  
  A numerical variable. Missing observations are removed by default.

- **stat**
  
  Statistic, either "mean" (default), "median" or "gmean" (geometric mean).

- **n**
  
  Number of replicates for the bootstrap (n=1000 by default).

- **CI**
  
  Confidence intervals (CI=95 by default).

- **digits**
  
  Number of digits for rounding (default = 2).

### Value

A data frame with the estimate and confidence intervals.

### Examples

```r
data(IgM, package = "ISwR")
bst(IgM, "median")
bst(IgM, "gmean")
```

## chisq.fisher

**Internal test for chi-squared assumption. Fisher (2 by 2). If results = T, it fails**

### Description

chisq.fisher is an internal function called by contingency and contingency2 that uses the Fisher exact test if results from the assumptions for the chi-squared test fail.

### Usage

chisq.fisher(tab)

### Arguments

- **tab**
  
  A numeric two by two table.
Description

`coef_det` estimates the coefficient of determination (r-squared) from fitted (predicted) and observed values. Outcome from the model is assumed to be numerical.

Usage

`coef_det(obs, fit)`

Arguments

- `obs` Vector with observed values (numerical outcome).
- `fit` Vector with fitted (predicted) values.

Value

A scalar, the coefficient of determination (r-squared).

Examples

```r
## Linear regression:
Riboflavin <- seq(0, 80, 10)
OD <- 0.0125 * Riboflavin + rnorm(9, 0.6, 0.03)
titration <- data.frame(Riboflavin, OD)
model1 <- lm(OD ~ Riboflavin, data = titration)
summary(model1)
coef_det(titration$OD, fitted(model1))

## Non-linear regression:
library(nlme, quietly = TRUE)
data(Puromycin)
mm.tx <- gnls(rate ~ SS MichealisMenten(conc, Vm, K),
             data = Puromycin,
             subset = state == "treated")
summary(mm.tx)
coef_det(Puromycin$rate[1:12], mm.tx$fitted)
```
Description

`contingency` is a wrap that calls `epi.2by2` from package `epiR`.

Usage

```r
contingency(
  object = NULL,
  formula = NULL,
  data = NULL,
  method = "cohort.count",
  ...
)
```

Arguments

- `object`: When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
- `formula`: A formula with shape: `outcome ~ exposure`.
- `data`: A data frame where the variables in the `formula` can be found.
- `method`: A character string with options: "cohort.count", "cohort.time", "case.control", or "cross.sectional".
- `...`: Further arguments passed to `epi.2by2`.

Details

`contingency` uses a formula as a way to input variables.
`contingency` displays the contingency table as a way for the user to check that the reference levels in the categorical variables (outcome and exposure) are correct. Then displays measures of association (table from `epi.2by2`). It also reports either chi-squared test or exact Fisher’s test; `contingency` checks which one of the tests two is appropriate.

See Also

`epi.2by2`.

Examples

```r
## A case-control study on the effect of alcohol on oesophageal cancer.
Freq <- c(386, 29, 389, 171)
status <- gl(2, 1, 4, labels = c("Control", "Case"))
alcohol <- gl(2, 2, labels = c("0-39", "40+"))
cancer <- data.frame(Freq, status, alcohol)
```
cancer <- expand_df(cancer)
ccontingency(status ~ alcohol, data = cancer, method = "case.control")

data(Oncho)
require(dplyr, quietly = TRUE)

Oncho %>%
  select(mf, area) %>%
cross_tbl(by = "mf") %>%
  theme_pubh(2)

Oncho %>%
  contingency(mf ~ area)

---

**contingency2**

*Measures of association from two by two contingency tables (direct input).*

**Description**

contingency2 is a wrap that calls epi.2by2 from package epiR.

**Usage**

`contingency2(aa, bb, cc, dd, ...)`

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aa</td>
<td>Number of cases where both exposure and outcome are present.</td>
</tr>
<tr>
<td>bb</td>
<td>Number of cases where exposure is present but outcome is absent.</td>
</tr>
<tr>
<td>cc</td>
<td>Number of cases where exposure is absent but outcome is present.</td>
</tr>
<tr>
<td>dd</td>
<td>Number of cases where both exposure and outcome are absent.</td>
</tr>
<tr>
<td>...</td>
<td>Further arguments passed to epi.2by2.</td>
</tr>
</tbody>
</table>

**See Also**

epi.2by2.

**Examples**

```r
# A case-control study on the effect of alcohol on oesophageal cancer.
Freq <- c(386, 29, 389, 171)
status <- gl(2, 1, 4, labels = c("Control", "Case"))
alcohol <- gl(2, 2, labels = c("0-39", "40+"))
cancer <- data.frame(Freq, status, alcohol)
cancer <- expand_df(cancer)
contingency2(171, 389, 29, 386, method = "case.control")
```
Cosmetics for tables of regression coefficients. Converts tables generated by `tbl_regression` to `huxtable` and adds some cosmetics.

**Description**

Cosmetics for tables of regression coefficients. Converts tables generated by `tbl_regression` to `huxtable` and adds some cosmetics.

**Usage**

`cosm_reg(gt_tbl, pad = 3, type = 3, bold = TRUE, head_label = "**Variable**")`

**Arguments**

- `gt_tbl` A table object generated by `tbl_regression`.
- `pad` Numerical, padding above and below rows.
- `type` Anova’s type to calculate global p-values.
- `bold` Display labels in bold?
- `head_label` Character, label to be used as head for the variable’s column.

**Value**

A `huxtable`.

**Examples**

```r
require(sjlabelled, quietly = TRUE)
data(diet, package = "Epi")
diet <- diet %>%
  var_labels(
    chd = "Coronary Heart Disease",
    fibre = "Fibre intake (g/day)"
  )

model_binom <- glm(chd ~ fibre, data = diet, family = binomial)
model_binom %>%
  tbl_regression(exponentiate = TRUE) %>%
  cosm_reg(bold = TRUE) %>%
  theme_pubh() %>%
  add_footnote(get_r2(model_binom), font_size = 9)

data(birthwt, package = "MASS")
birthwt <- birthwt %>%
  mutate(
    smoke = factor(smoke, labels = c("Non-smoker", "Smoker")),
```
```r
cosm_sum

race = factor(race, labels = c("White", "African American", "Other"))

var_labels(
  bwt = "Birth weight (g)",
  smoke = "Smoking status",
  race = "Race"
)

model_norm <- lm(bwt ~ smoke + race, data = birthwt)

model_norm %>%
tbl_regression() %>%
cosm_reg(bold = TRUE) %>%
theme_pubh() %>%
add_footnote(get_r2(model_norm), font_size = 9)
```

---

**cosm_sum**

Cosmetics for summary tables Adds some cosmetics to tables of descriptive statistics generated by *tbl_summary*.

### Description

Cosmetics for summary tables Adds some cosmetics to tables of descriptive statistics generated by *tbl_summary*.

### Usage

```r
cosm_sum(gt_tbl, pad = 3, bold = FALSE, head_label = "**Variable**")
```

### Arguments

- **gt_tbl**  
  A table object generated by *tbl_summary*.

- **pad**  
  Numerical, padding above and below rows.

- **bold**  
  Display labels in bold?

- **head_label**  
  Character, label to be used as head for the variable’s column.

### Details

Function *cosm_sum* adds some cosmetics to tables generated by *tbl_summary*, then converts the table as a *huxtable* and sets proper alignment.

### Value

A *huxtable*. 
Examples

require(dplyr, quietly = TRUE)

data(Oncho)
Oncho %>%
  select(-id) %>%
  tbl_summary() %>%
  cosm_sum(bold = TRUE) %>%
  theme_pubh(1)

cross_tab

Cross-tabulation.

Description

cross_tab is a wrapper to functions from package moonBook to construct tables of descriptive statistics stratified by levels of a categorical outcome.

Usage

cross_tab(
  object = NULL,
  formula = NULL,
  data = NULL,
  label = NULL,
  show.total = TRUE,
  p_val = FALSE,
  pad = 3,
  width = 0.8,
  method = 1,
  ...
)

Arguments

object When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.

formula A formula with shape: y ~ x, where y is a categorical outcome and x is the explanatory variable or a set of explanatory variables (see Details and Examples).

data A data frame where the variables in the formula can be found.

label A character, label to be used for the outcome (for non-labelled data).

show.total Logical, show column with totals?

p_val Logical, show p-values?

pad Numerical, padding above and bellow rows.

width Numerical, proportional width of the table.
method An integer indicating methods for continuous variables. 1 Reports means and standard deviations. 2 Reports medians and interquartile ranges.
... Additional arguments passed to mytable_sub.

Details

Function cross_tab is a relatively simple wrapper to function mytable of package moonBook. Its main purpose is to construct contingency tables but it can also be used to report a table with descriptives for all variables as long as they are still stratified by the outcome. Please see examples to see how to list explanatory variables. For categorical explanatory variables, the function reports column percentages. If data is labelled with sjlabelled, the label of the outcome (dependent) variable is used to name the outcome; this name can be changed with argument label.

Value

A huxtable with descriptive statistics stratified by levels of the outcome.

See Also

mytable

cross_tbl  Table of descriptive statistics by categorical variable.

description

cross_tbl is a wrapper to function from package tbl_summary that constructs tables of descriptive statistics stratified by levels of a categorical outcome.

Usage

cross_tbl(
  data,
  by,  
  head_label = " ",
  bold = TRUE,
  show_total = TRUE,
  p_val = FALSE,
  pad = 3,
  method = 2,
  ... )
Arguments

- **data**: A data frame where the variables in the formula can be found.
- **by**: The quoted name of the categorical variable (factor) used for the stratification.
- **head_label**: Character, label to be used as head for the variable’s column.
- **bold**: Display labels in bold?
- **show_total**: Logical, show column with totals?
- **p_val**: Logical, show p-values?
- **pad**: Numerical, padding above and bellow rows.
- **method**: An integer indicating methods for continuous variables. 1 Reports means and standard deviations. 2 Reports medians and interquartile ranges.
- **...**: Additional arguments passed to `tbl_summary`.

Details

Function `cross_tbl` is a relatively simple wrapper to function `tbl_summary`. It constructs contingency tables and can also be used to report a table with descriptives for all variables stratified by one of the variables. Please see examples to see how to list variables. If data is labelled, the label of the stratifying variable is used as part of the header.

Value

A huxtable with descriptive statistics stratified by levels of the outcome.

See Also

- `tbl_summary`

Examples

```r
require(dplyr, quietly = TRUE)

# data(Oncho)

## A two by two contingency table:
Oncho %>%
  select(mf, area) %>%
  cross_tbl(by = "mf", bold = TRUE) %>%
  theme_pubh(2)

## Reporting prevalence:
Oncho %>%
  select(mf, area) %>%
  cross_tbl(by = "area", bold = TRUE) %>%
  theme_pubh(2)

## Descriptive statistics for all variables in the `Oncho` data set except `id`.
Oncho %>%
  select(-id) %>%
```
diag_test

cross_tbl(by = "mf", bold = TRUE) %>%
theme_pubh(2)

---

**diag_test**  
Diag nost tests from variables.

**Description**

diag_test is a wrap function that calls epi.tests from package epiR. It computes sensitivity, specificity and other statistics related with screening tests.

**Usage**

diag_test(object = NULL, formula = NULL, data = NULL, ...)

**Arguments**

- **object**
  - When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.

- **formula**
  - A formula with shape: outcome ~ predictor (see details).

- **data**
  - A data frame where the variables in the formula can be found.

- **...**
  - Further arguments passed to epi.tests.

**Details**

For the formula, the outcome is the gold standard and the explanatory variable is the new (screening) test. See examples.

**See Also**

- epi.tests.

**Examples**

```r
## We compare the use of lung's X-rays on the screening of TB against the gold standard test.
Freq <- c(1739, 8, 51, 22)
BCG <- gl(2, 1, 4, labels = c("Negative", "Positive"))
Xray <- gl(2, 2, labels = c("Negative", "Positive"))
tb <- data.frame(Freq, BCG, Xray)
tb <- expand_df(tb)

tb %>%
diag_test(BCG ~ Xray)
```
diag_test2  
*Diagnostic tests from direct input.*

**Description**

diag_test2 is a wrap that calls epi.tests from package epiR. It computes sensitivity, specificity and other statistics related with screening tests.

**Usage**

diag_test2(aa, bb, cc, dd)

**Arguments**

- **aa**  
  Number of cases where both screening test and the gold standard are positive.

- **bb**  
  Number of cases where screening test is positive but gold standard is negative.

- **cc**  
  Number of cases where screening test is negative but gold standard is positive.

- **dd**  
  Number of cases where both screening test and the gold standard are negative.

**Details**

diag.test uses direct input variables.

**See Also**

epi.tests.

**Examples**

```r
## We compare the use of lung's X-rays on the screening of TB against the gold standard test.
diag_test2(22, 51, 8, 1739)
```

---

**estat**  
*Descriptive statistics for continuous variables.*

**Description**

estat calculates descriptives of numerical variables.

**Usage**

estat(object = NULL, formula = NULL, data = NULL, digits = 2, label = NULL)
**Arguments**

- **object**: When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.

- **formula**: A formula with shape: \( \sim x \) or \( \sim x \mid z \) (for groups).

- **data**: A data frame where the variables in the formula can be found.

- **digits**: Number of digits for rounding (default = 2).

- **label**: Label used to display the name of the variable (see examples).

**Value**

A data frame with descriptive statistics.

**See Also**

`summary`, `mytable`.

**Examples**

data(kfm, package = "ISwR")
require(sjlabelled, quietly = TRUE)
kfm <- kfm %>%
  var_labels(
    dl.milk = "Breast-milk intake (dl/day)",
    sex = "Sex",
    weight = "Child weight (kg)",
    ml.suppl = "Milk substitute (ml/day)",
    mat.weight = "Maternal weight (kg)",
    mat.height = "Maternal height (cm)"
  )

kfm %>%
estat(~ dl.milk)

estat(~ dl.milk | sex, data = kfm)

kfm %>%
estat(~ weight | sex)

---

**expand_df**

**Expand a data frame.**

**Description**

expand_df expands a data frame by a vector of frequencies.

**Usage**

```
expand_df(aggregate.data, index.var = "Freq", retain.freq = FALSE)
```
Arguments

aggregate.data A data frame.
index.var A numerical variable with the frequencies (counts).
retain.freq Logical expression indicating if frequencies should be kept.

Details

This is a generic function that resembles weighted frequencies in other statistical packages (for example, Stata). expand.df was adapted from a function developed by deprecated package epicalc (now package epiDisplay).

Value

An expanded data frame with replicates given by the frequencies.

Examples

Freq <- c(5032, 5095, 41, 204)
Mortality <- gl(2, 2, labels = c("No", "Yes"))
Calcium <- gl(2, 1, 4, labels = c("No", "Yes"))
anyca <- data.frame(Freq, Mortality, Calcium)
anyca
anyca.exp <- expand_df(anyca)
with(anyca.exp, table(Calcium, Mortality))

Fentress Migraine pain reduction.

Description

Randomised control trial on children suffering from frequent and severe migraine. Control group represents untreated children. The active treatments were either relaxation alone or relaxation with biofeedback.

Usage

Fentress

Format

A labelled tibble with 18 rows and 2 variables:

- **pain** Reduction in weekly headache activity expressed as percentage of baseline data.
- **group** Group, a factor with levels "Untreated", "Relaxation" (alone) and "Biofeedback" (relaxation and biofeedback).
freq_cont

Source


Examples

data(Fentress)

Fentress %>%
  strip_error(pain ~ group)

freq_cont

Relative and Cumulative Frequency.

Description

freq_cont tabulates a continuous variable by given classes.

Usage

freq_cont(x, bks, dg = 2)

Arguments

x A numerical (continuous) variable. Ideally, relatively long (greater than 100 observations).

bks Breaks defining the classes (see example).

dg Number of digits for rounding (default = 2).

Value

A data frame with the classes, the mid-point, the frequencies, the relative and cumulative frequencies.

Examples

data(IgM, package = "ISwR")
Ab <- data.frame(IgM)
estat(~IgM, data = Ab)
freq_cont(IgM, seq(0, 4.5, 0.5))
**Description**

`gen_bst_df` is a function that generates a data frame with confidence intervals of a continuous variable by levels of one or two categorical ones (factors).

**Usage**

```r
gen_bst_df(object = NULL, formula = NULL, data = NULL, stat = "mean", ...)
```

**Arguments**

- `object`: When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
- `formula`: A formula with shape: `y ~ x` or `y ~ x|z` where `y` is a numerical variable and both `x` and `z` are factors.
- `data`: A data frame where the variables in the `formula` can be found.
- `stat`: Statistic used for `bst`.
- `...`: Passes optional arguments to `bst`.

**Value**

A data frame with the confidence intervals by level.

**Examples**

```r
data(kfm, package = "ISwR")
require(sjlabelled, quietly = TRUE)
kfm <- kfm %>%
  var_labels(
    dl.milk = "Breast-milk intake (dl/day)",
    sex = "Sex",
    weight = "Child weight (kg)",
    ml.suppl = "Milk substitute (ml/day)",
    mat.weight = "Maternal weight (kg)",
    mat.height = "Maternal height (cm)"
  )

kfm %>%
  gen_bst_df(dl.milk ~ sex)

data(birthwt, package = "MASS")
require(dplyr, quietly = TRUE)
birthwt <- mutate(birthwt,
  smoke = factor(smoke, labels = c("Non-smoker", "Smoker")),
  Race = factor(race > 1, labels = c("White", "Non-white")))
```
birthwt <- birthwt %>%
  var_labels(
    bwt = "Birth weight (g)",
    smoke = "Smoking status"
  )

gen_bst_df(bwt ~ smoke | Race, data = birthwt)

---

**geo_mean**  
*Geometric mean.*

**Description**  
Geometric mean.

**Usage**  
`geo_mean(x)`

**Arguments**  
- `x`: A numeric variable with no negative values.

**Value**  
A scalar, the calculated geometric mean.

**Examples**  
```r
data(IgM, package = "ISwR")
Ab <- data.frame(IgM)
estat(~IgM, data = Ab)
geo_mean(IgM)
```

---

**get_r2**  
*Estimate R2 or Pseudo-R2 from regression models*

**Description**  
`get_r2` is a wrapper function that calls `r2` from package `performance`. Calculates the R2 or pseudo-R2 value for different regression model objects, returning a character object for easy printing in tables of coefficients.

**Usage**  
`get_r2(model, ...)`
**Arguments**

- `model`: A statistical regression model.
- `...`: Additional arguments passed to `r2`.

**Details**

The main purpose of `get_r2` is to allow easy printing of R2 value in tables of coefficients (see examples).

**See Also**

- `r2`.

**Examples**

```r
require(dplyr, quietly = TRUE)
require(sjlabelled, quietly = TRUE)

data(birthwt, package = "MASS")
birthwt <- birthwt %>%
museate(
  smoke = factor(smoke, labels = c("Non-smoker", "Smoker")),
  race = factor(race, labels = c("White", "African American", "Other"))
) %>%
var_labels(
  bwt = "Birth weight (g)",
  smoke = "Smoking status",
  race = "Race"
)

model_norm <- lm(bwt ~ smoke + race, data = birthwt)

model_norm %>%
glm_coef(labels = model_labels(model_norm)) %>%
as_hux() %>%
set_align(everywhere, 2:3, "right") %>%
theme_pubh() %>%
add_footnote(get_r2(model_norm), font_size = 9)
```

---

**gf_star**

Annotating a plot to display differences between groups.

**Description**

`gf_star` is a function used to display differences between groups (see details).

**Usage**

`gf_star(fig, x1, y1, x2, y2, y3, legend = "*", ...)"
Arguments

fig
A gformula object.

x1
Position in x for the start of the horizontal line.

y1
Position in y for the start of the vertical line, below to the horizontal line.

x2
Position in x for the end of the horizontal line.

y2
Position in y where the horizontal line is drawn.

y3
Position in y where the text is added.

legend
Character text used for annotating the plot.

... Additional information passed to `gf_text`.

Details

This function draws an horizontal line from coordinate (x1, y2) to coordinate (x2, y2). Draws vertical lines below the horizontal line, towards data, from (x1, y1) to (x1, y2) and from (x2, y1) to (x2, y2). Finally, adds text above the horizontal line, at the mid point between x1 and x2. See examples.

Examples

data(kfm, package = "ISwR")
require(sjlabelled, quietly = TRUE)
kfm <- kfm %>%
  var_labels(
    dl.milk = "Breast-milk intake (dl/day)",
    sex = "Sex",
    weight = "Child weight (kg)",
    ml.suppl = "Milk substitute (ml/day)",
    mat.weight = "Maternal weight (kg)",
    mat.height = "Maternal height (cm)"
  )

kfm %>%
  box_plot(dl.milk ~ sex, fill = "thistle", alpha = 0.8) %>%
  gf_star(x1 = 1, y1 = 10.9, x2 = 2, y2 = 11, y3 = 11.2)

kfm %>%
  box_plot(dl.milk ~ sex, fill = "thistle", alpha = 0.8) %>%
  gf_star(1, 10.9, 2, 11, 11.4, legend = "p = 0.035", size = 2.5)

data(energy, package = "ISwR")
energy <- energy %>%
  var_labels(
    expend = "Energy expenditure (MJ/day)",
    stature = "Stature"
  )

energy %>%
  strip_error(expend ~ stature, col = "red") %>%
  gf_star(1, 13, 2, 13.2, 13.4, "**")
glm_coef displays estimates with confidence intervals and p-values from generalised linear models (see Details).

Usage

```r
glm_coef(
  model,             # A model from any of the classes listed in the details section.
  digits = 2,        # A scalar, number of digits for rounding the results (default = 2).
  alpha = 0.05,      # Significant level (default = 0.05) used to calculate confidence intervals.
  labels = NULL,     # An optional character vector with the names of the coefficients (including intercept).
  se_rob = FALSE,    # Logical, should robust errors be used to calculate confidence intervals? (default = FALSE).
  type = "cond",     # Character, either "cond" (condensed) or "ext" (extended). See details.
  exp_norm = FALSE)  # Logical, should estimates and confidence intervals should be exponentiated? (for family == "gaussian").
)
```

Arguments

- **model**: A model from any of the classes listed in the details section.
- **digits**: A scalar, number of digits for rounding the results (default = 2).
- **alpha**: Significant level (default = 0.05) used to calculate confidence intervals.
- **labels**: An optional character vector with the names of the coefficients (including intercept).
- **se_rob**: Logical, should robust errors be used to calculate confidence intervals? (default = FALSE).
- **type**: Character, either "cond" (condensed) or "ext" (extended). See details.
- **exp_norm**: Logical, should estimates and confidence intervals should be exponentiated? (for family == "gaussian").

Details

glm_coef recognises objects (models) from the following classes: `clm`, `clogit`, `coxph`, `gee`, `glm`, `glmerMod`, `lm`, `lme`, `lmerMod`, `multinom`, `negbin`, `polr` and `surveg`.

For models from logistic regression (including conditional logistic, ordinal and multinomial), Poisson or survival analysis, coefficient estimates and corresponding confidence intervals are automatically exponentiated (back-transformed).

By default, glm_coef uses naive standard errors for calculating confidence intervals but has the option of using robust standard errors instead.

glm_coef can display two different data frames depending on the option of type, for type type = "cond" (the default), glm_coef displays the standard table of coefficients with confidence intervals and p-values; for type = "ext", glm_coef displays additional statistics including standard errors.

Please read the Vignette on Regression for more details.
harm_mean

Value

A data frame with estimates, confidence intervals and p-values from glm objects.

Examples

```r
require(dplyr, quietly = TRUE)
require(sjlabelled, quietly = TRUE)

## Continuous outcome.
data(birthwt, package = "MASS")
birthwt <- birthwt %>%
  mutate(
    smoke = factor(smoke, labels = c("Non-smoker", "Smoker")),
    race = factor(race, labels = c("White", "African American", "Other"))
  ) %>%
  var_labels(
    bwt = "Birth weight (g)",
    smoke = "Smoking status",
    race = "Race"
  )

model_norm <- lm(bwt ~ smoke + race, data = birthwt)

glm_coef(model_norm, labels = model_labels(model_norm))

## Logistic regression.
data(diet, package = "Epi")
model_binom <- glm(chd ~ fibre, data = diet, family = binomial)
model_binom %>%
  glm_coef(labels = c("Constant", "Fibre intake (g/day)"))

model_binom %>%
  glm_coef(labels = c("Constant", "Fibre intake (g/day)"), type = "ext")
```

harm_mean

Harmonic mean.

Description

Harmonic mean.

Usage

harm_mean(x)

Arguments

x A numeric variable with no zero values.
Value

A scalar, the calculated harmonic mean.

Examples

data(IgM, package = "ISwR")
Ab <- data.frame(IgM)
estat(~IgM, data = Ab)
harm_mean(IgM)

hist_norm

Histogram with Normal density curve.

Description

hist_norm constructs histograms and adds corresponding Normal density curve.

Usage

hist_norm(
  object = NULL,
  formula = NULL,
  data = NULL,
  bins = 20,
  fill = "indianred3",
  color = "black",
  alpha = 0.4,
  ...
)

Arguments

object
  When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
formula
  A formula with shape: ~ y or ~ y|x where y is a numerical variable and x is a factor.
data
  A data frame where the variables in the formula can be found.
bins
  Number of bins of the histogram.
fill
  Colour to fill the bars of the histogram.
color
  Colour used for the border of the bars.
alpha
  Opacity (0 = invisible, 1 = opaque).
  ...
  Further arguments passed to gf_dhistogram.
Hodgkin

Examples

```r
# Load required packages
require(dplyr, quietly = TRUE)
require(sjlabelled, quietly = TRUE)
data(birthwt, package = "MASS")
birthwt <- birthwt %>%
  mutate(
    smoke = factor(smoke, labels = c("Non-smoker", "Smoker")),
    Race = factor(race > 1, labels = c("White", "Non-white"))
  ) %>%
  var_labels(
    bwt = "Birth weight (g)",
    smoke = "Smoking status"
  )
birthwt %>%
  hist_norm(~ bwt, alpha = 0.7, bins = 20, fill = "cadetblue")
birthwt %>%
  hist_norm(~ bwt | smoke, alpha = 0.7, bins = 20, fill = "cadetblue")
```

Hodgkin

T-cell counts from Hodgkin’s disease patients.

Description

Number of CD4+ T-cells and CD8+ T-cells in blood samples from patients in remission from Hodgkin’s disease or in remission from disseminated malignancies.

Usage

Hodgkin

Format

A labelled tibble with 40 rows and 3 variables:

- **CD4** Concentration of CD4+ T-cells (cells / mm^3).
- **CD8** Concentration of CD8+ T-cells (cells / mm^3).
- **Group** Group, factor with levels "Non-Hodgkin" and "Hodgkin".

Source


Examples

data(Hodgkin)
require(dplyr, quietly = TRUE)
require(sjlabelled, quietly = TRUE)

Hodgkin <- Hodgkin %>%
  mutate(
    Ratio = CD4 / CD8
  ) %>%
  var_labels(
    Ratio = "CD4+ / CD8+ T-cells"
  )

estat(~ Ratio | Group, data = Hodgkin)

Hodgkin %>%
  qq_plot(~ Ratio | Group)

Hodgkin$Ratio <- Hodgkin$CD4 / Hodgkin$CD8
estat(~ Ratio | Group, data = Hodgkin)

qq_plot(~ Ratio | Group, data = Hodgkin)

---

inv_logit

*Inverse of the logit*

**Description**

*inv_logit* Calculates the inverse of the logit (probability in logistic regression)

**Usage**

`inv_logit(x)`

**Arguments**

- `x` Numerical value used to compute the inverse of the logit.

---

jack_knife

*Ranks leverage observations from Jackknife method.*

**Description**

*jack_knife* Ranks the squared differences between mean values from Jackknife analysis (arithmetic mean estimated by removing one observation at a time) and the original mean value.
Usage

jack_knife(x)

Arguments

x A numeric variable. Missing values are removed by default.

Value

Data frame with the ranked squared differences.

See Also

rank_leverage.

Examples

x <- rnorm(10, 170, 8)
xmean(x)
jack_knife(x)

x <- rnorm(100, 170, 8)
mean(x)
head(jack_knife(x))

Kirkwood

Body weight and plasma volume.

Description

Body weight and plasma volume in eight healthy men.

Usage

Kirkwood

Format

A labelled data frame with 8 rows and 3 variables:

subject Subject ID.
weight Body weight in kg.
volume Plasma volume in litres.

Source

Examples

```r
data(Kirkwood)
Kirkwood %>%
gf_point(volume ~ weight) %>%
gf_lm(col = "indianred3", interval = "confidence", fill = "indianred3")
```

---

```
knife_mean
```

### Description

`knife_mean` is an internal function. Calculates arithmetic means by removing one observation at a time.

### Usage

```r
knife_mean(x)
```

### Arguments

- `x`: A numerical variable. Missing values are removed for the mean calculation.

### Value

A vector with the mean calculations.

### Examples

```r
x <- rnorm(10, 170, 8)
x
mean(x)
knife_mean(x)
```

---

```
leverage
```

### Description

`leverage` is an internal function called by `rank_leverage`.

### Usage

```r
leverage(x)
```
logistic_gof

Arguments

x A numeric variable. Missing values are removed by default.

Details

Estimates the leverage of each observation around the arithmetic mean.

Value

Variable with corresponding leverage estimations

Examples

x <- rnorm(10, 170, 8)
mean(x)
leverage(x)
rank_leverage(x)

logistic_gof Goodness of fit for Logistic Regression.

Description

logistic_gof performs the Hosmer and Lemeshow test to test the goodness of fit of a logistic regression model. This function is part of residuals.lrm from package rms.

Usage

logistic_gof(model)

Arguments

model A logistic regression model object.

Author(s)

Frank Harell, Vanderbilt University <f.harrell@vanderbilt.edu>

References


Examples

data(diet, package = "Epi")
model <- glm(chd ~ fibre, data = diet, family = binomial)
glm_coef(model, labels = c("Constant", "Fibre intake (g/day)"))
logistic_gof(model)
**Description**

An international case-control study to test the hypothesis that breast cancer is related to the age that a woman gives childbirth.

**Usage**

Macmahon

**Format**

A labelled tibble with 185 rows and 2 variables:

- **cancer** Diagnosed with breast cancer, a factor with levels "No" and "Yes".
- **age** Age mother gives childbirth, factor with levels ",<20", "20-24", "25-29", "30-34" and ",>34".

**Source**


**Examples**

```r
require(dplyr, quietly = TRUE)
data(Macmahon)

Macmahon %>%
  select(cancer, age) %>%
cross_tbl(by = "cancer")
```

---

**mhor**

*Mantel-Haenszel odds ratio.*

**Description**

*mhor* computes odds ratios by levels of the stratum variable as well as the Mantel-Haenszel pooled odds ratio. The test for effect modification (test for interaction) is also displayed.

**Usage**

```
mhor(object = NULL, formula = NULL, data = NULL)
```
model_labels

Arguments

object When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.

formula A formula with shape: outcome ~ stratum/exposure.
data A data frame containing the variables used in formula.

Value

Odds ratios with 95% outcome by levels of stratum. The Mantel-Haenszel pooled OR and the test for effect modification is also reported.

See Also

mh

Examples

data(oswego, package = "epitools")
require(dplyr, quietly = TRUE)
require(sjlabelled, quietly = TRUE)
oswego <- oswego %>%
  mutate(
    ill = factor(ill, labels = c("No", "Yes")),
    sex = factor(sex, labels = c("Female", "Male")),
    chocolate.ice.cream = factor(chocolate.ice.cream, labels = c("No", "Yes"))
  ) %>%
  var_labels(
    ill = "Developed illness",
    sex = "Sex",
    chocolate.ice.cream = "Consumed chocolate ice cream"
  )

oswego %>%
  select(ill, sex, chocolate.ice.cream) %>%
  tbl_summary() %>%
  cosm_sum() %>%
  theme_pubh()

  oswego %>%
  mhor(ill ~ sex / chocolate.ice.cream)

model_labels

Using labels as coefficient names in tables of coefficients.

Description

model_labels replaces row names in glm_coef with labels from the original data frame.
Usage

model_labels(model, intercept = TRUE)

Arguments

model A generalised linear model.
intercept Logical, should the intercept be added to the list of coefficients?

Details

model_labels does not handle yet interaction terms, see examples.
Please read the Vignette on Regression for more examples.

Examples

require(dplyr, quietly = TRUE)
require(sjlabelled, quietly = TRUE)
data(birthwt, package = "MASS")
birthwt <- birthwt %>%
  mutate(
    smoke = factor(smoke, labels = c("Non-smoker", "Smoker")),
    race = factor(race, labels = c("White", "African American", "Other"))
  )

var_labels(
  bwt = "Birth weight (g)",
  smoke = "Smoking status",
  race = "Race"
)

model_norm <- lm(bwt ~ smoke + race, data = birthwt)
glm_coef(model_norm, labels = model_labels(model_norm))

model_int <- lm(formula = bwt ~ smoke * race, data = birthwt)

model_int %>%
glm_coef(labels = c(
  model_labels(model_int),
  "Smoker: African American",
  "Smoker: Other"
))

multiple Multiple comparisons with plot.

Description

multiple displays results from post-doc analysis and constructs corresponding plot.
Usage

```
multiple(
    model,  
    formula,  
    adjust = "mvt",  
    type = "response",  
    reverse = TRUE,  
    level = 0.95,  
    digits = 2,  
    ...
)
```

Arguments

- **model**: A fitted model supported by `emmeans`, such as the result of a call to `aov`, `lm`, `glm`, etc.
- **formula**: A formula with shape: `~ y` or `~ y | x` (for interactions). Where `y` is the term of the model on which comparisons are made and `x` is a term interacting with `y`.
- **adjust**: Method to adjust CIs and p-values (see details).
- **type**: Type of prediction (matching "linear.predictor", "link", or "response").
- **reverse**: Logical argument. Determines the direction of comparisons.
- **level**: Confidence interval significance level.
- **digits**: Number of digits for rounding (default = 2).
- **...**: Further arguments passed to `emmeans`.

Details

The default adjusting method is "mvt" which uses the multivariate t distribution. Other options are: "bonferroni", "holm", "hochberg", "tukey" and "none". The default option for argument `reverse` is to make reverse comparisons, i.e., against the reference level matching comparisons from `lm` and `glm`.

Value

A list with objects: `df` A data frame with adjusted p-values, `fig_ci` a plot with estimates and adjusted confidence intervals and `fig_pval` a plot comparing adjusted p-values.

See Also

`emmeans`, `pwpp`.

Examples

```
data(birthwt, package = "MASS")
birthwt$race <- factor(birthwt$race, labels = c("White", "African American", "Other"))

model_1 <- aov(bwt ~ race, data = birthwt)
```
odds_trend

Function to calculate OR using Wald CI, and plot trend.

Description

odds_trend calculates the odds ratio with confidence intervals (Wald) for different levels (three or more) of the exposure variable, constructs the corresponding plot and calculates if the trend is significant or not.

Usage

odds_trend(formula, data, angle = 45, hjust = 1, method = "wald", ...)

Arguments

- **formula**: A formula with shape: outcome ~ exposure.
- **data**: A data frame where the variables in the formula can be found.
- **angle**: Angle of for the x labels (default = 45).
- **hjust**: Horizontal adjustment for x labels (default = 1).
- **method**: Method for calculating confidence interval around odds ratio.
- **...**: Passes optional arguments to oddsratio.

Details

odds_trend is a wrap function that calls oddsratio from package epitools.

Additional methods for confidence intervals include: "midp", "fisher", and "small".

Value

A list with components df a data frame with the results and fig corresponding plot.

See Also

oddsratio.
## Examples

```r
## A cross-sectional study looked at the association between obesity and a biopsy resulting
## from mammography screening.
Freq <- c(3441, 34, 39137, 519, 20509, 280, 12149, 196, 11882, 199)
Biopsy <- gl(2, 1, 10, labels = c("No", "Yes"))
Weight <- gl(5, 2, 10, labels = c("Underweight", "Normal", "Over (11-24%)",
                               "Over (25-39%)", "Over (> 39%)"))
breast <- data.frame(Freq, Biopsy, Weight)
breast

breast <- expand_df(breast)
require(sjlabelled, quietly = TRUE)
breast <- var_labels(breast,
                     Weight = "Weight group")

odds_trend(Biopsy ~ Weight, data = breast)$df
odds_trend(Biopsy ~ Weight, data = breast)$fig
```

### Description

Study of onchocerciasis ("river blindness") in Sierra Leone, in which subjects were classified according to whether they lived in villages in savannah or rainforest area.

### Usage

Oncho

### Format

A labelled tibble with 1302 rows and 7 variables:

- **id**: Subject ID.
- **mf**: Infected with Onchocerciasis volvulus, factor with levels "Not-infected" and "Infected".
- **area**: Area of residence, factor with levels "Savannah" and "Rainforest".
- **agegrp**: Age group in years, factor with levels "5-9", "10-19", "20-39" and "40+".
- **sex**: Subject sex, factor with levels "Male" and "Female".
- **mfload**: Microfiliariae load.
- **lesions**: Severe eye lesions, factor with levels "No" and "Yes".
Source

Examples
```r
require(dplyr, quietly = TRUE)
data(Oncho)

Oncho %>%
  select(mf, agegrp) %>%
cross_tbl(by = "mf")
```

predict_inv

Given \( y \) solve for \( x \) in a simple linear model.

Description

predict_inv Calculates the value the predictor \( x \) that generates value \( y \) with a simple linear model.

Usage

predict_inv(model, y)

Arguments

- **model**: A simple linear model object (class lm).
- **y**: A numerical scalar, the value of the outcome for which we want to calculate the predictor \( x \).

Value

The estimated value of the predictor.

Examples

```r
## Spectrophotometry example. Titration curve for riboflavin (nmol/ml). The sample has an absorbance
## of 1.15. Aim is to estimate the concentration of riboflavin in the sample.

Riboflavin <- seq(0, 80, 10)
OD <- 0.0125 * Riboflavin + rnorm(9, 0.6, 0.03)
titration <- data.frame(Riboflavin, OD)

require(sjlabelled, quietly = TRUE)
titration <- titration %>%
  var_labels(
    Riboflavin = "Riboflavin (nmol/ml)",
  )
```
prop_or

OD = "Optical density"

titration %>%
gf_point(OD ~ Riboflavin) %>%
gf_smooth(col = "indianred3", se = TRUE, lwd = 0.5, method = "loess")

## Model with intercept different from zero:
model <- lm(OD ~ Riboflavin, data = titration)
glm_coef(model)
predict_inv(model, 1.15)

prop_or

Proportion, \( p_1 \) from proportion \( p_2 \) and OR.

Description

prop_or is a simple function to calculate a proportion, from another proportion and the odds ratio between them.

Usage

prop_or(p2, or)

Arguments

p2	The value of a proportion in the unexposed group (p2).

or	The odds ratio of \( p_1/p_2 \).

Value

\( p_1 \), the proportion in the exposed group (p1).

Examples

flu <- matrix(c(20, 80, 220, 140), nrow = 2)
colnames(flu) <- c("Yes", "No")
rownames(flu) <- c("Vaccine", "Placebo")
flu

or <- (20 * 140) / (80 * 220)
p2 <- 80 / 220
prop_or(p2 = p2, or = or)
20 / 240
pseudo_r2

Description

Pseudo R2 (logistic regression) pseudo_r2 Calculates R2 analogues (pseudo R2) of logistic regression.

Usage

pseudo_r2(model)

Arguments

model A logistic regression model.

Details

pseudo_r2 calculates three pseudo R2 of logistic regression models: 1) Nagelkerke, @0 Cox and Snell, 3) Hosmer and Lemeshow.

Value

A data frame with the calculated pseudo R2 values.

Examples

data(Oncho)
model_oncho <- glm(mf ~ area, data = Oncho, binomial)
glm_coef(model_oncho, labels = c("Constant", "Area (rainforest/savannah)"))
pseudo_r2(model_oncho)

qq_plot

Description

qq_plot constructs quantile-quantile plots against the standard normal distribution (also known as quantile-normal plots).
qq_plot

Usage

qq_plot(
    object = NULL,
    formula = NULL,
    data = NULL,
    pch = 20,
    col = "indianred3",
    ylab = NULL,
    ...
)

Arguments

object When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.

formula A formula with shape: ~ x or ~ x|z where x is a numerical variable and z is a factor.

data A data frame where the variables in the formula can be found.

pch Point character passed to gf_qq.

col Colour of the reference line, passed to gf_line.

ylab Optional character passed as label for the y-axis.

Further arguments passed to gf_qq.

Examples

data(kfm, package = "ISwR")
require(sjlabelled, quietly = TRUE)
kfm <- kfm %>%
  var_labels(
    dl.milk = "Breast-milk intake (dl/day)",
    sex = "Sex",
    weight = "Child weight (kg)",
    ml.suppl = "Milk substitute (ml/day)",
    mat.weight = "Maternal weight (kg)",
    mat.height = "Maternal height (cm)"
  )

kfm %>%
  qq_plot(~dl.milk)

qq_plot(~ dl.milk | sex, data = kfm)
**rank_influence**

*Ranks observations based upon influence measures on models.*

**Description**

`rank_influence` calculates influence measures of each data observation on models and then ranks them.

**Usage**

```r
rank_influence(model)
```

**Arguments**

- `model`  A generalised linear model object.

**Details**

`rank_influence` is a wrap function that calls `influence.measures`, ranks observations on their significance influence on the model and displays the 10 most influential observations (if they are significant).

**See Also**

`influence.measures`.

**Examples**

```r
data(diet, package = "Epi")
model <- glm(chd ~ fibre, data = diet, family = binomial)
rank_influence(model)
```

---

**rank_leverage**

*Ranks observations by leverage.*

**Description**

`rank_leverage` ranks observations by their leverage (influence) on the arithmetic mean.

**Usage**

```r
rank_leverage(x)
```

**Arguments**

- `x`  A numeric variable. Missing values are removed by default.
Value
A data frame ranking observations by their leverage around the mean.

See Also
jack_knife.

Examples
```r
x <- rnorm(10, 170, 8)
x
mean(x)
rank_leverage(x)

x <- rnorm(100, 170, 8)
mean(x)
head(rank_leverage(x))
```

---

**Description**
reference_range estimates the reference range (reference interval) of a numerical variable.

**Usage**
```
reference_range(avg, std)
```

**Arguments**
- `avg` : The arithmetic mean (a scalar numerical value).
- `std` : The standard deviation (a scalar numerical value).

**Details**
The reference range assumes normality and represents the limits that would include 95 observations.

**Value**
A data frame with the reference range limits.

**Examples**
```r
x <- rnorm(100, 170, 8)
round(mean(x), 2)
round(sd(x), 2)
round(reference_range(mean(x), sd(x)), 2)
```
rel_dis  Relative Dispersion.

Description

Calculates the coefficient of variation (relative dispersion) of a variable. The relative dispersion is defined as the standard deviation over the arithmetic mean.

Usage

rel_dis(x)

Arguments

x  A numerical variable. NA's observations are removed by default.

Value

The coefficient of variation (relative dispersion).

Examples

height <- rnorm(100, 170, 8)
rel_dis(height)

Roberts  Extracorporeal membrane oxygenation in neonates.

Description

A clinical trial on the value of extracorporeal membrane oxygenation for term neonates with severe respiratory failure. RCT compares active treatment against conventional management.

Usage

Roberts

Format

A labelled tibble with 185 rows and 2 variables:

emo  Extracorporeal membrane oxygenation treatment, factor with levels "No" and "Yes".

survived  One year survival, factor with levels "No" and "Yes".
Source

Examples
```r
require(dplyr, quietly = TRUE)
data(Roberts)

Roberts %>%
  select(survived, emo) %>%
cross_tbl(by = "survived")
```

Description
A case-control study of oral contraceptives and stroke in young women with presence or absence of hypertension. Cases represent thrombotic stroke and controls are hospital controls. The group of no hypertension includes normal blood pressure (<140/90 mm Hg) and borderline hypertension (140-159/90-94 mm Hg). Hypertension group includes moderate hypertension (160-179/95-109 mm Hg) and severe hypertension (180+/110+ mm Hg). This data has been used as an example of join exposure by Rothman for measuring interactions (see examples).

Usage
Rothman

Format
A labelled tibble with 477 rows and 3 variables:

- **stroke**: Thrombotic stroke, factor with levels "No" and "Yes".
- **oc**: Current user of oral contraceptives, factor with levels "Non-user" and "User".
- **ht**: Hypertension, factor with levels "No" (<160/95 mm Hg) and "Yes".

Source
Examples

```r
require(dplyr, quietly = TRUE)
data(Rothman)

Rothman %>%
  select(stroke, oc, ht) %>%
  tbl_summary() %>%
cosm_sum()

mhor(stroke ~ ht / oc, data = Rothman)

## Model with standard interaction term:
model1 <- glm(stroke ~ ht * oc, data = Rothman, family = binomial)
glm_coef(model1)

## Model considering join exposure:
Rothman$join <- 0
Rothman$join[Rothman$oc == "Non-user" & Rothman$ht == "Yes"] <- 1
Rothman$join[Rothman$oc == "User" & Rothman$ht == "No"] <- 2
Rothman$join[Rothman$oc == "User" & Rothman$ht == "Yes"] <- 3
Rothman$join <- factor(Rothman$join, labels = c(
  "Unexposed", "Hypertension", "OC user",
  "OC and hypertension"
))

require(sjlabelled, quietly = TRUE)
Rothman$join <- set_label(Rothman$join, label = "Exposure")

Rothman %>%
  select(stroke, join) %>%
cross_tbl(by = "stroke")

model2 <- glm(stroke ~ join, data = Rothman, family = binomial)
glm_coef(model2)
```

round_pval

Rounding p-values.

Description

`round_pval` is an internal function called by `glm_coef` to round p-values from model coefficients.

Usage

`round_pval(pval)`

Arguments

- `pval` vector of p-values, numeric.
Description

A case-control study to investigate the effects of passive smoking on cancer. Passive smoking was defined as exposure to the cigarette smoke of a spouse who smoked at least one cigarette per day for at least 6 months.

Usage

Sandler

Format

A labelled tibble with 998 rows and 3 variables:

- **passive** Passive smoker, factor with levels "No" and "Yes".
- **cancer** Diagnosed with cancer, factor with levels "No" and "Yes".
- **smoke** Active smoker, factor with levels "No" and "Yes".

Source


Examples

```r
require(dplyr, quietly = TRUE)
data(Sandler)

Sandler %>%
  select(cancer, passive) %>%
  cross_tbl(by = "cancer")

mhor(cancer ~ smoke / passive, data = Sandler)
```

---

Sharples  

**Measured and self-reported weight in New Zealand.**

Description

Data on measured and self-reported weight from 40–50 year old participants in the 1989/1990 Life In New Zealand Survey.
Usage
Sharples

Format
A tibble with 343 rows and 4 variables:

**srweight**  Self-reported weight in kg.
**weight**   Measured weight in kg.
**srbmi** Body mass index calculated from self-reported weight and self-reported height in kg/m^2.
**mbmi** Body mass index calculated from measured weight and measured height in kg/m^2.

Source

Examples
Sharples %>%
  bland_altman(srweight ~ weight, transform = TRUE) %>%
  gf_labs(x = "Mean of weights (kg)", y = "Measured weight / Self-reported weight") %>%
  gf_theme(theme = sjPlot::theme_sjplot2(base_size = 9))

---

ss_jk  Sum of squares for Jackknife.

Description

ss_jk is an internal function called by jack_knife. It calculates the squared difference of a numerical variable around a given value (for example, the mean).

Usage
ss_jk(obs, stat)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obs</td>
<td>A numerical vector with no missing values (NA's).</td>
</tr>
<tr>
<td>stat</td>
<td>The value of the statistic that is used as a reference.</td>
</tr>
</tbody>
</table>

Value

The squared difference between a variable and a given value.
Examples

```r
x <- rnorm(10, 170, 8)
x
mean(x)
ss_jk(x, mean(x))
jackKnife(x)
```

---

**stats_quotes**

*Internal function to calculate descriptive statistics.*

**Description**

`stats_quotes` is an internal function called by `estat`.

**Usage**

```r
stats_quotes(x, data2, digits = 2)
```

**Arguments**

- `x`: a numeric variable
- `data2`: A data frame where `x` can be found.
- `digits`: Number of digits for rounding.

---

**strip_error**

*Strip plots with error bars.*

**Description**

`strip_error` constructs strip plots with error bars showing 95% confidence intervals around mean values.

**Usage**

```r
strip_error(
  object = NULL,
  formula = NULL,
  data = NULL,
  pch = 20,
  size = 1,
  alpha = 0.7,
  col = "indianred3",
  ...
)
```
Arguments

- **object**
  When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.

- **formula**
  A formula with shape: \( y \sim x \) or \( y \sim x \mid z \) where \( y \) is a numerical variable and both \( x \) and \( z \) are factors.

- **data**
  A data frame where the variables in the formula can be found.

- **pch**
  Point character passed to `gf_point` or `gf_jitter`.

- **size**
  Size of the symbol (pch) for representing data values.

- **alpha**
  Opacity of the symbol (0 = invisible, 1 = opaque).

- **col**
  A colour or formula used for mapping colour.

- **...**
  Additional information passed to `gf_jitter` or `gf_point`.

Examples

```r
data(energy, package = "ISwR")
require(sjlabelled, quietly = TRUE)
energy <- energy %>%
  var_labels(
    expend = "Energy expenditure (MJ/day)",
    stature = "Stature"
  )

energy %>%
  strip_error(expend ~ stature, col = "red")

t.test(expend ~ stature, data = energy)

## Adding an horizontal line to show significant difference:
energy %>%
  strip_error(expend ~ stature, col = "red") %>%
  gf_star(1, 13, 2, 13.2, 13.4, "]**")

data(birthwt, package = "MASS")
require(dplyr, quietly = TRUE)
birthwt <- birthwt %>%
  mutate(
    smoke = factor(smoke, labels = c("Non-smoker", "Smoker")),
    Race = factor(race > 1, labels = c("White", "Non-white"))
  )

birthwt %>%
  var_labels(
    bwt = "Birth weight (g)",
    smoke = "Smoking status"
  )

birthwt %>
  strip_error(bwt ~ smoke | Race, col = "darksalmon")

birthwt %>
  strip_error(bwt ~ smoke, col = ~Race) %>
```
Thall

```
gf_refine(ggsci::scale_color_jama())

birthwt %>%
  strip_error(bwt ~ smoke, pch = ~Race, col = ~Race) %>%
gf_refine(ggsci::scale_color_jama())

birthwt %>%
  strip_error(bwt ~ smoke | Race)
```

<table>
<thead>
<tr>
<th>Thall</th>
<th>RCT on the treatment of epilepsy.</th>
</tr>
</thead>
</table>

**Description**

Randomised control trial of an antiepileptic drug (prograbide), in which the number of seizures of 59 patients at baseline and other four follow-up visits were recorded.

**Usage**

Thall

**Format**

A tibble with 59 rows and 8 variables:

- **id** Subject ID.
- **treat** Treatment, factor with levels "Control" and "Prograbide".
- **base** Number of seizures at baseline.
- **age** Age in years at baseline.
- **y1** Number of seizures at year one follow-up.
- **y2** Number of seizures at year two follow-up.
- **y3** Number of seizures at year three follow-up.
- **y4** Number of seizures at year four follow-up.

**Source**


Examples

```r
data(Thall)

c1 <- cbind(Thall[, c(1:5)], count = Thall$y1)[, c(1:4, 6)]
c2 <- cbind(Thall[, c(1:4, 6)], count = Thall$y2)[, c(1:4, 6)]
c3 <- cbind(Thall[, c(1:4, 7)], count = Thall$y3)[, c(1:4, 6)]
c4 <- cbind(Thall[, c(1:4, 8)], count = Thall$y3)[, c(1:4, 6)]
epilepsy <- rbind(c1, c2, c3, c4)

require(lme4, quietly = TRUE)
model_glmer <- glmer(count ~ treat + base + I(age - mean(age, na.rm = TRUE)) +
(1 | id), data = epilepsy, family = poisson)

glm_coef(model_glmer, labels = c("Treatment (Prograbide/Control)",
"Baseline count", "Age (years)"
))
```

---

**theme_pubh**

A theme for huxtables This function quickly set a default style for a huxtable.

**Description**

A theme for huxtables This function quickly set a default style for a huxtable.

**Usage**

```r
theme_pubh(ht, rw = 1)
```

**Arguments**

- **ht** A huxtable object.
- **rw** A numeric vector with the rows on which a bottom border is desired.

**Details**

theme_pubh is a variation of theme_article with the added flexibility of adding a bottom border line at desired row numbers.

**Examples**

```r
require(dplyr, quietly = TRUE)
data(Oncho)

Oncho %>%
  select(area, mf) %>%
cross_tbl(by = "area") %>%
  theme_pubh(2)
```
```r
data(Bernard)
t1 <- estat(~ apache | fate, data = Bernard)
t2 <- estat(~ o2del | fate, data = Bernard)
rbind(t1, t2) %>%
as_hux() %>%
theme_pubh(c(1, 3))
```

---

### Tuzson

**Peak knee velocity in walking at flexion and extension.**

---

**Description**

Data of peak knee velocity in walking at flexion and extension in studies about functional performance in cerebral palsy.

**Usage**

Tuzson

**Format**

A labelled tibble with 18 rows and 2 variables:

- **flexion**  Peak knee velocity in gait: flexion (degree/s).
- **extension**  Peak knee velocity in gait: extension (degree/s).

**Source**


**Examples**

```r
data(Tuzson)
Tuzson %>%
gf_point(flexion ~ extension)

cor.test(~ flexion + extension, data = Tuzson)
```
Description


Usage

Vanderpump

Format

A labelled tibble with 1314 rows and 3 variables:

- **vstatus** Vitality status, factor with levels "Alive" and "Death".
- **smoker** Smoking status, factor with levels "Non-smoker" and "Smoker".
- **agegrp** Age group, factor with levels "18-44", "45-64" and "64+".

Source


Examples

```r
require(dplyr, quietly = TRUE)
data(Vanderpump)

Vanderpump %>%
  select(vstatus, agegrp, smoker) %>%
  tbl_strata(
    strata = smoker,
    .tbl_fun = ~ .x %>%
      tbl_summary(by = agegrp)
  ) %>%
cosm_sum(bold = TRUE, head_label = "")

mhor(vstatus ~ agegrp / smoker, data = Vanderpump)
```
Index

* datasets
  Bernard, 5
  Brenner, 9
  Fentress, 22
  Hodgkin, 31
  Kirkwood, 33
  Macmahon, 36
  Oncho, 41
  Roberts, 48
  Rothman, 49
  Sandler, 51
  Sharples, 51
  Thall, 55
  Tuzson, 57
  Vanderpump, 58

  expand_df, 21
  Fentress, 22
  freq_cont, 23
  gen_bst_df, 24
  geo_mean, 10, 25
  get_r2, 25
  gf_boxplot, 8
  gf_dhistogram, 30
  gf_jitter, 54
  gf_line, 45
  gf_point, 7, 54
  gf_qq, 45
  gf_star, 26
  gf_summary, 5
  gf_text, 27
  glm_coef, 28
  harm_mean, 29
  hist_norm, 30
  Hodgkin, 31
  huxtable, 14, 15
  influence.measures, 46
  inv_logit, 32
  jack_knife, 32, 47, 52
  Kirkwood, 33
  knife_mean, 34
  leverage, 34
  logistic_gof, 35
  Macmahon, 36
  mean, 10
  median, 10
  mh, 37
  mhor, 36
  model_labels, 37

axis_labs, 3
bar_error, 4
Bernard, 5
bland_altman, 6
box_plot, 8
Brenner, 9
bst, 10, 24
chisq.fisher, 10
coeff_det, 11
contingency, 12
contingency2, 13
cosm_reg, 14
cosm_sum, 15
cross_tab, 16
cross_tbl, 17
diag_test, 19
diag_test2, 20
diag_test3
emmeans, 39
epi.2by2, 12, 13
epi.tests, 19, 20
estat, 20
multiple, 38
mytable, 17, 21
mytable_sub, 17

odds_trend, 40
oddsratio, 40
Oncho, 41

predict_inv, 42
prop_or, 43
pseudo_r2, 44
pwpp, 39

qq_plot, 44

r2, 26
rank_influence, 46
rank_leverage, 33, 34, 46
reference_range, 47
rel_dis, 48
Roberts, 48
Rothman, 49
round_pval, 50

Sandler, 51
Sharples, 51
ss_jk, 52
stats_quotes, 53
strip_error, 53
summary, 21

tbl_regression, 14
tbl_summary, 15, 17, 18
Thall, 55
theme_pubh, 56
Tuzson, 57

Vanderpump, 58