Package 'statsExpressions'

June 20, 2020

Type Package

Title Expressions with Statistical Details

Version 0.4.2

Maintainer Indrajeet Patil <patilindrajeet.science@gmail.com>

Description Statistical processing backend for 'ggstatsplot',
this package creates expressions with details from statistical tests.
Currently, it supports only the most common types of statistical
tests: parametric, nonparametric, robust, and Bayesian versions of
t-test/ANOVA, correlation analyses, contingency table analysis, and
meta-analysis.

License GPL-3 | file LICENSE

URL https://indrajeetpatil.github.io/statsExpressions,

https://github.com/IndrajeetPatil/statsExpressions

BugReports https://github.com/IndrajeetPatil/statsExpressions/issues

Depends R (>= 3.6.0)

Imports broomExtra, correlation (>= 0.3.0), dplyr, effectsize, ez,
ipmisc (>= 3.0.1), metafor, metaplus, rcompanion, rlang, stats,
tidyBF, tidyr, WRS2 (>= 1.1-0)

Suggests ggplot2, knitr, rmarkdown, spelling, testthat

VignetteBuilder knitr

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.1.0.9000

NeedsCompilation no

Author Indrajeet Patil [cre, aut, cph]
(<https://orcid.org/0000-0003-1995-6531>)

Repository CRAN

Date/Publication 2020-06-20 14:00:02 UTC

R topics documented:

---

bugs_long   *Tidy version of the "Bugs" dataset.*

Description

Tidy version of the "Bugs" dataset.

Usage

```
bugs_long
```

Format

A data frame with 372 rows and 6 variables

- subject. Dummy identity number for each participant.

- gender. Participant's gender (Female, Male).

- region. Region of the world the participant was from.

- education. Level of education.

- condition. Condition of the experiment the participant gave rating for (**LDLF**: low freighteningness and low disgustingness; **LFHD**: low freighteningness and high disgustingness; **HFHD**: high freighteningness and low disgustingness; **HFHD**: high freighteningness and high disgustingness).
- desire. The desire to kill an arthropod was indicated on a scale from 0 to 10.

Details

This data set, "Bugs", provides the extent to which men and women want to kill arthropods that vary in freighteningness (low, high) and disgustingness (low, high). Each participant rates their attitudes towards all anthropods. Subset of the data reported by Ryan et al. (2013).

Source

https://www.sciencedirect.com/science/article/pii/S0747563213000277

Examples

```
dim(bugs_long)
head(bugs_long)
dplyr::glimpse(bugs_long)
```

---

expr_anova_bayes   *Making expression containing Bayesian one-way ANOVA results.*

Description

Making expression containing Bayesian one-way ANOVA results.

Usage

```
expr_anova_bayes(data, x, y, paired = FALSE, bf.prior = 0.707, k = 2L, ...)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. A matrix or tables will **not** be accepted.
x	The grouping variable from the dataframe data.
y	The response (a.k.a. outcome or dependent) variable from the dataframe data.
paired	Logical that decides whether the experimental design is repeated measures/within-subjects or between-subjects. The default is FALSE.
bf.prior	A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors.
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
...	Additional arguments (currently ignored).

Value

For more details, see- https://indrajeetpatil.github.io/statsExpressions/articles/stats_
details.html

Examples

```
set.seed(123)

# between-subjects --------------------------------------
# with defaults
statsExpressions::expr_anova_bayes(
  data = ggplot2::msleep,
  x = vore,
  y = sleep_rem
)

# modifying the defaults
statsExpressions::expr_anova_bayes(
  data = ggplot2::msleep,
  x = vore,
  y = sleep_rem,
  k = 3,
  bf.prior = 0.8
)

# repeated measures --------------------------------------
statsExpressions::expr_anova_bayes(
  data = WRS2::WineTasting,
  x = Wine,
  y = Taste,
  paired = TRUE,
  k = 4
)
```

---

expr_anova_nonparametric

Making text subtitle for non-parametric ANOVA.

Description

Making text subtitle for non-parametric ANOVA.

Usage

```
expr_anova_nonparametric(
  data,
  x,
```

```
    y,
    paired = FALSE,
    conf.type = "perc",
    conf.level = 0.95,
    k = 2L,
    nboot = 100L,
    stat.title = NULL,
    ...
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. A matrix or tables will **not** be accepted.
`x`	The grouping variable from the dataframe `data`.
`y`	The response (a.k.a. outcome or dependent) variable from the dataframe `data`.
`paired`	Logical that decides whether the experimental design is repeated measures/within-subjects or between-subjects. The default is `FALSE`.
`conf.type`	A vector of character strings representing the type of intervals required. The value should be any subset of the values `"norm"`, `"basic"`, `"perc"`, `"bca"`. For more, see `?boot::boot.ci`.
`conf.level`	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (`0.95`).
`k`	Number of digits after decimal point (should be an integer) (Default: `k = 2`).
`nboot`	Number of bootstrap samples for computing confidence interval for the effect size (Default: `100`).
`stat.title`	A character describing the test being run, which will be added as a prefix in the subtitle. The default is `NULL`. An example of a `stat.title` argument will be something like `"Student's t-test: "`.
`...`	Additional arguments (currently ignored).

Details

For paired designs, the effect size is Kendall's coefficient of concordance (*W*), while for between-subjects designs, the effect size is epsilon-squared (for more, see `?rcompanion::epsilonSquared` and `?rcompanion::kendallW`).

Value

For more details, see- https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html

Examples

```
# setup
set.seed(123)
```

```
library(statsExpressions)

# -------------- within-subjects design -------------------------------

# creating the subtitle
statsExpressions::expr_anova_nonparametric(
  data = bugs_long,
  x = condition,
  y = desire,
  paired = TRUE,
  conf.level = 0.99,
  k = 2
)

# -------------- between-subjects design -------------------------------

statsExpressions::expr_anova_nonparametric(
  data = ggplot2::msleep,
  x = vore,
  y = sleep_rem,
  paired = FALSE,
  conf.level = 0.99,
  conf.type = "perc"
)
```

---

expr_anova_parametric   *Making expression containing parametric ANOVA results*

Description

The effect sizes and their confidence intervals are computed using `effectsize::eta_squared` and `effectsize::omega_squared` functions.

Usage

```
expr_anova_parametric(
  data,
  x,
  y,
  paired = FALSE,
  effsize.type = "unbiased",
  partial = TRUE,
  conf.level = 0.95,
  var.equal = FALSE,
  sphericity.correction = TRUE,
  k = 2L,
  stat.title = NULL,
  ...
)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. A matrix or tables will **not** be accepted.
x	The grouping variable from the dataframe data.
y	The response (a.k.a. outcome or dependent) variable from the dataframe data.
paired	Logical that decides whether the experimental design is repeated measures/within-subjects or between-subjects. The default is FALSE.
effsize.type	Type of effect size needed for *parametric* tests. The argument can be "biased" (equivalent to "d" for Cohen's *d* for **t-test**; "partial_eta" for partial eta-squared for **anova**) or "unbiased" (equivalent to "g" Hedge's *g* for **t-test**; "partial_omega" for partial omega-squared for **anova**)).
partial	If TRUE, return partial indices.
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (0.95).
var.equal	a logical variable indicating whether to treat the variances in the samples as equal. If TRUE, then a simple F test for the equality of means in a one-way analysis of variance is performed. If FALSE, an approximate method of Welch (1951) is used, which generalizes the commonly known 2-sample Welch test to the case of arbitrarily many samples.
sphericity.correction	
	Logical that decides whether to apply correction to account for violation of sphericity in a repeated measures design ANOVA (Default: TRUE).
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
stat.title	A character describing the test being run, which will be added as a prefix in the subtitle. The default is NULL. An example of a stat.title argument will be something like "Student's t-test: ".
...	Additional arguments (currently ignored).

Value

For more details, see- https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html

Note

For repeated measures designs (paired = TRUE), only partial omega-squared and partial eta-squared are supported.

Examples

```
# for reproducibility
set.seed(123)
library(statsExpressions)

# ------------------- between-subjects ----------------------------
```

```
# with defaults
statsExpressions::expr_anova_parametric(
  data = ggplot2::msleep,
  x = vore,
  y = sleep_rem,
  paired = FALSE,
  k = 3
)

# modifying the defaults
statsExpressions::expr_anova_parametric(
  data = ggplot2::msleep,
  x = vore,
  y = sleep_rem,
  paired = FALSE,
  effsize.type = "eta",
  partial = FALSE,
  var.equal = TRUE
)

# ------------------- repeated measures ----------------------------

statsExpressions::expr_anova_parametric(
  data = iris_long,
  x = condition,
  y = value,
  paired = TRUE,
  k = 4
)
```

---

expr_anova_robust   *Expression containing results from heteroscedastic one-way ANOVA*
                           for trimmed means

Description

Expression containing results from heteroscedastic one-way ANOVA for trimmed means

Usage

```
expr_anova_robust(
  data,
  x,
  y,
  paired = FALSE,
  tr = 0.1,
  nboot = 100,
```

```
  conf.level = 0.95,
  k = 2L,
  stat.title = NULL,
  ...
)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. A matrix or tables will **not** be accepted.
x	The grouping variable from the dataframe `data`.
y	The response (a.k.a. outcome or dependent) variable from the dataframe `data`.
paired	Logical that decides whether the experimental design is repeated measures/within-subjects or between-subjects. The default is `FALSE`.
tr	Trim level for the mean when carrying out `robust` tests. If you get error stating "Standard error cannot be computed because of Winsorized variance of 0 (e.g., due to ties). Try to decrease the trimming level.", try to play around with the value of `tr`, which is by default set to `0.1`. Lowering the value might help.
nboot	Number of bootstrap samples for computing confidence interval for the effect size (Default: `100`).
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (`0.95`).
k	Number of digits after decimal point (should be an integer) (Default: `k = 2`).
stat.title	A character describing the test being run, which will be added as a prefix in the subtitle. The default is `NULL`. An example of a `stat.title` argument will be something like `"Student's t-test: "`.
...	Additional arguments (currently ignored).

Value

For more details, see- `https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html`

Examples

```
# for reproducibility
set.seed(123)

# ---------------------- between-subjects ---------------------------

# going with the defaults
statsExpressions::expr_anova_robust(
  data = ggplot2::midwest,
  x = state,
  y = percbelowpoverty,
```

```
    paired = FALSE,
    nboot = 10
)

# changing defaults
expr_anova_robust(
  data = ggplot2::midwest,
  x = state,
  y = percollege,
  paired = FALSE,
  conf.level = 0.99,
  tr = 0.2,
  nboot = 10
)

# ---------------------- within-subjects ---------------------------

statsExpressions::expr_anova_robust(
  data = iris_long,
  x = condition,
  y = value,
  paired = TRUE,
  tr = 0.2,
  k = 3
)
```

---

expr_contingency_tab   *Making expression for contingency table and goodness of fit tests*

Description

Making expression for contingency table and goodness of fit tests

Usage

```
expr_contingency_tab(
  data,
  x,
  y = NULL,
  counts = NULL,
  ratio = NULL,
  nboot = 100,
  paired = FALSE,
  stat.title = NULL,
  conf.level = 0.95,
  conf.type = "norm",
  bias.correct = TRUE,
  k = 2L,
```

```
  ...
)

expr_onesample_proptest(
  data,
  x,
  y = NULL,
  counts = NULL,
  ratio = NULL,
  nboot = 100,
  paired = FALSE,
  stat.title = NULL,
  conf.level = 0.95,
  conf.type = "norm",
  bias.correct = TRUE,
  k = 2L,
  ...
)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. A matrix or tables will **not** be accepted.
x	The variable to use as the **rows** in the contingency table.
y	The variable to use as the **columns** in the contingency table. Default is NULL. If NULL, one-sample proportion test (a goodness of fit test) will be run for the main variable. Otherwise an appropriate association test will be run.
counts	A string naming a variable in data containing counts, or NULL if each row represents a single observation (Default).
ratio	A vector of proportions: the expected proportions for the proportion test (should sum to 1). Default is NULL, which means the null is equal theoretical proportions across the levels of the nominal variable. This means if there are two levels this will be ratio = c(0.5,0.5) or if there are four levels this will be ratio = c(0.25,0.25,0.25,0.25), etc.
nboot	Number of bootstrap samples for computing confidence interval for the effect size (Default: 100).
paired	Logical indicating whether data came from a within-subjects or repeated measures design study (Default: FALSE). If TRUE, McNemar's test subtitle will be returned. If FALSE, Pearson's chi-square test will be returned.
stat.title	Title for the effect being investigated with the chi-square test. The default is NULL, i.e. no title will be added to describe the effect being shown. An example of a stat.title argument will be something like "main x condition" or "interaction".
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (0.95).

conf.type	A vector of character strings representing the type of intervals required. The value should be any subset of the values "norm", "basic", "perc", "bca". For more, see ?boot::boot.ci.
bias.correct	If TRUE (default), a bias correction will be applied to Cramer's *V*.
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
...	Additional arguments (currently ignored).

Details

For more details about how the effect sizes and their confidence intervals were computed, see documentation in ?rcompanion::cramerV, ?rcompanion::cramerVFit, and ?rcompanion::cohenG.

Value

Expression for contingency analysis (Pearson's chi-square test for independence for between-subjects design or McNemar's test for within-subjects design) or goodness of fit test for a single categorical variable.

References

For more details, see- https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html

Examples

```
# ----------------------- association tests ----------------------------

set.seed(123)
library(statsExpressions)

# without counts data
statsExpressions::expr_contingency_tab(
  data = mtcars,
  x = am,
  y = cyl,
  paired = FALSE,
  nboot = 15
)

# ----------------------- goodness of fit tests --------------------------

# for reproducibility
set.seed(123)

# with counts
expr_contingency_tab(
  data = as.data.frame(HairEyeColor),
  x = Eye,
  counts = Freq,
```

```
    ratio = c(0.2, 0.2, 0.3, 0.3)
  )
```

---

expr_corr_test   *Making expression for correlation analysis*

Description

Making expression for correlation analysis

Usage

```
expr_corr_test(
  data,
  x,
  y,
  beta = 0.1,
  type = "parametric",
  bf.prior = 0.707,
  conf.level = 0.95,
  k = 2L,
  stat.title = NULL,
  ...
)
```

Arguments

data	for use with `formula`, a data frame containing all the data
x	The column in `data` containing the explanatory variable to be plotted on the x-axis. Can be entered either as a character string (e.g., ″x″) or as a bare expression (e.g, x).
y	The column in `data` containing the response (outcome) variable to be plotted on the y-axis. Can be entered either as a character string (e.g., ″y″) or as a bare expression (e.g, y).
beta	bending constant (Default: 0.1). For more, see ?WRS2::pbcor.
type	Type of association between paired samples required (""parametric": Pearson's product moment correlation coefficient" or ""nonparametric": Spearman's rho" or ""robust": percentage bend correlation coefficient" or ""bayes": Bayes Factor for Pearson's *r*"). Corresponding abbreviations are also accepted: ″p″ (for parametric/pearson), ″np″ (nonparametric/spearman), ″r″ (robust), ″bf″ (for bayes factor), resp.
bf.prior	A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors.
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (0.95).

k	Number of digits after decimal point (should be an integer) (Default: k = 2).
stat.title	A character describing the test being run, which will be added as a prefix in the subtitle. The default is NULL. An example of a stat.title argument will be something like "Student's t-test: ".
...	Additional arguments (ignored).

Value

Expression containing results from correlation test with confidence intervals for the correlation coefficient estimate.

References

For more details, see - https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html

Examples

```
# for reproducibility
set.seed(123)

# without changing defaults
statsExpressions::expr_corr_test(
  data = ggplot2::midwest,
  x = area,
  y = percblack,
  type = "parametric"
)

# changing defaults
statsExpressions::expr_corr_test(
  data = ggplot2::midwest,
  x = area,
  y = percblack,
  beta = 0.2,
  type = "robust"
)
```

---

expr_meta_bayes   *Making expression containing Bayesian random-effects meta-analysis.*

Description

Making expression containing Bayesian random-effects meta-analysis.

Usage

```
expr_meta_bayes(
  data,
  d = prior("norm", c(mean = 0, sd = 0.3)),
  tau = prior("invgamma", c(shape = 1, scale = 0.15)),
  k = 2,
  messages = TRUE,
  ...
)
```

Arguments

data	A dataframe. It **must** contain columns named estimate (effect sizes or outcomes) and std.error (corresponding standard errors). These two columns will be used for yi and sei arguments in metafor::rma (for parametric analysis) or metaplus::metaplus (for robust analysis).
d	prior distribution on the average effect size d. The prior probability density function is defined via [prior](#).
tau	prior distribution on the between-study heterogeneity tau (i.e., the standard deviation of the study effect sizes dstudy in a random-effects meta-analysis. A (nonnegative) prior probability density function is defined via [prior](#).
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
messages	Deprecated. Retained only for backward compatibility.
...	Arguments passed on to [metaBMA::meta_random](#)

        labels optional: character values with study labels. Can be a character vector or the quoted or unquoted name of the variable in data

        rscale_contin scale parameter of the JZS prior for the continuous covariates.

        rscale_discrete scale parameter of the JZS prior for discrete moderators.

        centering whether continuous moderators are centered.

        logml how to estimate the log-marginal likelihood: either by numerical integration ("integrate") or by bridge sampling using MCMC/Stan samples ("stan"). To obtain high precision with logml="stan", many MCMC samples are required (e.g., logml_iter=10000,warmup=1000).

        summarize how to estimate parameter summaries (mean, median, SD, etc.): Either by numerical integration (summarize = "integrate") or based on MCMC/Stan samples (summarize = "stan").

        ci probability for the credibility/highest-density intervals.

        rel.tol relative tolerance used for numerical integration using [integrate](#). Use rel.tol=.Machine$double.eps for maximal precision (however, this might be slow).

        logml_iter number of iterations (per chain) from the posterior distribution of d and tau. The samples are used for computing the marginal likelihood of the random-effects model with bridge sampling (if logml="stan") and for obtaining parameter estimates (if summarize="stan"). Note that the argument iter=2000 controls the number of iterations for estimation of the random-effect parameters per study in random-effects meta-analysis.

silent_stan whether to suppress the Stan progress bar.

Examples

```
# setup
set.seed(123)
library(metaplus)

# renaming to what `statsExpressions` expects
df <- dplyr::rename(mag, estimate = yi, std.error = sei)

# making subtitle
expr_meta_bayes(
  data = df,
  k = 3,
  # additional arguments given to `metaBMA`
  iter = 5000,
  summarize = "integrate",
  control = list(adapt_delta = 0.99, max_treedepth = 15)
)
```

---

expr_meta_parametric   *Making expression with frequentist random-effects meta-analysis re-
                          sults*

Description

This analysis is carried out using the metafor package. For more, see ?metafor::rma.

Usage

```
expr_meta_parametric(
  data,
  conf.level = 0.95,
  k = 2L,
  output = "subtitle",
  caption = NULL,
  messages = TRUE,
  ...
)
```

Arguments

data   A dataframe. It **must** contain columns named estimate (effect sizes or out-
                  comes) and std.error (corresponding standard errors). These two columns
                  will be used for yi and sei arguments in metafor::rma (for parametric analy-
                  sis) or metaplus::metaplus (for robust analysis).

conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (`0.95`).
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
output	Character describing the desired output. If `"subtitle"`, a formatted subtitle with summary effect and statistical details will be returned, and if `"caption"`, expression containing details from model summary will be returned.
caption	Text to display as caption. This argument is relevant only when `output = "caption"`.
messages	Deprecated. Retained only for backward compatibility.
...	Arguments passed on to metafor::rma vi vector of length k with the corresponding sampling variances. See 'Details'. weights optional argument to specify a vector of length k with user-defined weights. See 'Details'. ai see below and the documentation of the escalc function for more details. bi see below and the documentation of the escalc function for more details. ci see below and the documentation of the escalc function for more details. di see below and the documentation of the escalc function for more details. n1i see below and the documentation of the escalc function for more details. n2i see below and the documentation of the escalc function for more details. x1i see below and the documentation of the escalc function for more details. x2i see below and the documentation of the escalc function for more details. t1i see below and the documentation of the escalc function for more details. t2i see below and the documentation of the escalc function for more details. m1i see below and the documentation of the escalc function for more details. m2i see below and the documentation of the escalc function for more details. sd1i see below and the documentation of the escalc function for more details. sd2i see below and the documentation of the escalc function for more details. xi see below and the documentation of the escalc function for more details. mi see below and the documentation of the escalc function for more details. ri see below and the documentation of the escalc function for more details. ti see below and the documentation of the escalc function for more details. sdi see below and the documentation of the escalc function for more details. r2i see below and the documentation of the escalc function for more details. ni see below and the documentation of the escalc function for more details. mods optional argument to include one or more moderators in the model. A single moderator can be given as a vector of length k specifying the values of the moderator. Multiple moderators are specified by giving a matrix with k rows and as many columns as there are moderator variables. Alternatively, a model formula can be used to specify the model. See 'Details'. measure character string indicating the type of data supplied to the function. When measure="GEN" (default), the observed effect sizes or outcomes and corresponding sampling variances (or standard errors) should be supplied to the function via the yi, vi, and sei arguments (only one of the two, vi or sei, needs to be specified). Alternatively, one can set measure to one of the effect size or outcome measures described under the documentation for the escalc function and specify the needed data via the appropriate arguments.

intercept logical indicating whether an intercept should be added to the model (the default is TRUE). Ignored when mods is a formula.

slab optional vector with labels for the k studies.

subset optional (logical or numeric) vector indicating the subset of studies that should be used for the analysis.

add see the documentation of the [escalc](#) function.

to see the documentation of the [escalc](#) function.

drop00 see the documentation of the [escalc](#) function.

vtype see the documentation of the [escalc](#) function.

method character string specifying whether a fixed- or a random/mixed-effects model should be fitted. A fixed-effects model (with or without moderators) is fitted when using method="FE". Random/mixed-effects models are fitted by setting method equal to one of the following: "DL", "HE", "SJ", "ML", "REML", "EB", "HS", or "GENQ". Default is "REML". See 'Details'.

weighted logical indicating whether weighted (default) or unweighted estimation should be used to fit the model.

test character string specifying how test statistics and confidence intervals for the fixed effects should be computed. By default (test="z"), Wald-type tests and CIs are obtained, which are based on a standard normal distribution. When test="knha", the method by Knapp and Hartung (2003) is used for adjusting test statistics and confidence intervals. See 'Details'.

digits integer specifying the number of decimal places to which the printed results should be rounded (if unspecified, the default is 4).

btt optional vector of indices specifying which coefficients to include in the omnibus test of moderators. Can also be a string to grep for. See 'Details'.

verbose logical indicating whether output should be generated on the progress of the model fitting (the default is FALSE). Can also be an integer. Values > 1 generate more verbose output. See 'Note'.

control optional list of control values for the iterative estimation algorithms. If unspecified, default values are defined inside the function. See 'Note'.

Examples

```
# setup
set.seed(123)
library(statsExpressions)
library(metaplus)

# renaming to what `statsExpressions` expects
df <- dplyr::rename(mag, estimate = yi, std.error = sei)

# making subtitle
expr_meta_parametric(
  data = df,
  k = 3,
  messages = FALSE
)
```

```
# getting tidy data frame with coefficients
expr_meta_parametric(
  data = df,
  messages = FALSE,
  output = "tidy"
)

# making caption
expr_meta_parametric(
  data = df,
  k = 2,
  messages = FALSE,
  output = "caption"
)

# getting dataframe with model summary
expr_meta_parametric(
  data = df,
  messages = FALSE,
  output = "glance"
)
```

---

expr_meta_robust   *Making expression with frequentist random-effects robust meta-analysis results*

Description

This analysis is carried out using the `metaplus` package. For more, see `?metaplus::metaplus`.

Usage

```
expr_meta_robust(data, random = "mixture", k = 2, messages = TRUE, ...)
```

Arguments

data   A dataframe. It **must** contain columns named `estimate` (effect sizes or outcomes) and `std.error` (corresponding standard errors). These two columns will be used for `yi` and `sei` arguments in `metafor::rma` (for parametric analysis) or `metaplus::metaplus` (for robust analysis).

random   The type of random effects distribution. One of "normal","t-dist","mixture", for standard normal, t-distribution or mixture of normals respectively.

k   Number of digits after decimal point (should be an integer) (Default: k = 2).

messages   Deprecated. Retained only for backward compatibility.

...   Arguments passed on to [metaplus::metaplus](#)

              mods data frame of covariates corresponding to each study

label The label to be used for this model when plotting

plotci Should profile be plotted for each confidence interval?

justfit Should model only be fitted? If justfit=TRUE then profiling and like-lihood ratio statistics are not calculated. Useful for when bootstrapping.

slab Vector of character strings corresponding to each study.

useAGQ Use adaptive Gauss-Hermite quadrature when performing integration for t-distribution random-effects. Useful when standard errors for studies are relatively small.

quadpoints Number of quadrature points to use in adaptive Gauss-Hermite quadrature.

Examples

```
# setup
set.seed(123)
library(metaplus)

# renaming to what `statsExpressions` expects
df <- dplyr::rename(mag, estimate = yi, std.error = sei)

# creating expression (changing few defaults)
expr_meta_robust(
  data = df,
  random = "normal",
  k = 4,
  messages = TRUE
)
```

---

expr_template   *Template for subtitles with statistical details for tests*

Description

Template for subtitles with statistical details for tests

Usage

```
expr_template(
  no.parameters,
  stat.title = NULL,
  statistic.text,
  stats.df,
  effsize.text,
  effsize.df,
  n,
  conf.level = 0.95,
```

```
    k = 2L,
    k.parameter = 0L,
    k.parameter2 = 0L,
    n.text = NULL,
    ...
)
```

Arguments

no.parameters	An integer that specifies that the number of parameters for the statistical test. Can be `0` for non-parametric tests, 1 for tests based on *t*-statistic or chi-squared statistic, 2 for tests based on *F*-statistic.
stat.title	A character describing the test being run, which will be added as a prefix in the subtitle. The default is NULL. An example of a `stat.title` argument will be something like `"Student's t-test: "`.
statistic.text	A character that specifies the relevant test statistic. For example, for tests with *t*-statistic, `statistic.text = "t"`. If you want to use plotmath, you will have to quote the argument (e.g., `quote(italic("t"))`).
stats.df	A dataframe containing the following columns:
	• *statistic*: the numeric value of a statistic.
	• *parameter*: the numeric value of a parameter being modeled (often degrees of freedom for the test); note that if `no.parameters = 0L` (e.g., for non-parametric tests), this column will be irrelevant.
	• *parameter1*, *parameter2* relevant only if the statistic in question has two degrees of freedom (e.g., anova).
	• *p.value* the two-sided *p*-value associated with the observed statistic.
effsize.text	A character that specifies the relevant effect size. For example, for Cohen's *d* statistic, `effsize.text = "d"`. If you want to use plotmath, you will have to quote the argument (e.g., `quote(italic("d"))`).
effsize.df	A dataframe containing the following columns:
	• *estimate*: estimated value of the effect size.
	• *conf.low*: lower bound for effect size estimate.
	• *conf.high*: upper bound for effect size estimate.
n	An integer specifying the sample size used for the test.
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (`0.95`).
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
k.parameter, k.parameter2	
	Number of decimal places to display for the parameters (default: `0`).
n.text	A character that specifies the design, which will determine what the n stands for. For example, for repeated measures, this can be `quote(italic("n")["pairs"])`, while for independent subjects design this can be `quote(italic("n")["obs"])`. If NULL, defaults to generic `quote(italic("n"))`.
...	Currently ignored.

Examples

```
set.seed(123)

# creating a dataframe with stats results
stats_df <-
  cbind.data.frame(
    statistic = 5.494,
    parameter = 29.234,
    p.value = 0.00001
  )

# creating a dataframe with effect size results
effsize_df <-
  cbind.data.frame(
    estimate = -1.980,
    conf.low = -2.873,
    conf.high = -1.088
  )

# subtitle for *t*-statistic with Cohen's *d* as effect size
statsExpressions::expr_template(
  no.parameters = 1L,
  stats.df = stats_df,
  effsize.df = effsize_df,
  statistic.text = quote(italic("t")),
  effsize.text = quote(italic("d")),
  n = 32L,
  conf.level = 0.95,
  k = 3L,
  k.parameter = 3L
)
```

---

expr_t_bayes   *Making expression containing Bayesian* t*-test results*

Description

Making expression containing Bayesian *t*-test results

Usage

```
expr_t_bayes(data, x, y, bf.prior = 0.707, paired = FALSE, k = 2, ...)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. A matrix or tables will **not** be accepted.
x	The grouping variable from the dataframe data.
y	The response (a.k.a. outcome or dependent) variable from the dataframe data.

bf.prior	A number between `0.5` and 2 (default `0.707`), the prior width to use in calculating Bayes factors.
paired	Logical that decides whether the experimental design is repeated measures/within-subjects or between-subjects. The default is `FALSE`.
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
...	Additional arguments (currently ignored).

References

For more details, see- https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html

Examples

```
# for reproducibility
set.seed(123)

# ------------- between-subjects design -------------------------

statsExpressions::expr_t_bayes(
  data = mtcars,
  x = am,
  y = wt,
  paired = FALSE
)

# ------------- within-subjects design --------------------------

statsExpressions::expr_t_bayes(
  data = dplyr::filter(bugs_long, condition %in% c("LDLF", "LDHF")),
  x = condition,
  y = desire,
  paired = TRUE
)
```

---

expr_t_nonparametric *Making expression for Mann-Whitney* U-*test/Wilcoxon test results*

Description

Making expression for Mann-Whitney *U*-test/Wilcoxon test results

Usage

```
expr_t_nonparametric(
  data,
  x,
  y,
  paired = FALSE,
  k = 2L,
  conf.level = 0.95,
  conf.type = "norm",
  nboot = 100,
  stat.title = NULL,
  ...
)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. A matrix or tables will **not** be accepted.
x	The grouping variable from the dataframe data.
y	The response (a.k.a. outcome or dependent) variable from the dataframe data.
paired	Logical that decides whether the experimental design is repeated measures/within-subjects or between-subjects. The default is FALSE.
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (0.95).
conf.type	A vector of character strings representing the type of intervals required. The value should be any subset of the values "norm", "basic", "perc", "bca". For more, see ?boot::boot.ci.
nboot	Number of bootstrap samples for computing confidence interval for the effect size (Default: 100).
stat.title	A character describing the test being run, which will be added as a prefix in the subtitle. The default is NULL. An example of a stat.title argument will be something like "Student's t-test: ".
...	Additional arguments (currently ignored).

Details

For the two independent samples case, the Mann-Whitney *U*-test is calculated and *W* is reported from *stats::wilcox.test*. For the paired samples case the Wilcoxon signed rank test is run and *V* is reported.

Since there is no single commonly accepted method for reporting effect size for these tests we are computing and reporting *r* (computed as $Z/\sqrt{N}$) along with the confidence intervals associated with the estimate. Note that *N* here corresponds to total *sample size* for independent/between-subjects designs, and to total number of *pairs* (and **not** *observations*) for repeated measures/within-subjects designs.

Note: The *stats::wilcox.test* function does not follow the same convention as *stats::t.test*. The sign of the *V* test statistic will always be positive since it is **the sum of the positive signed ranks**. Therefore, *V* will vary in magnitude but not significance based solely on the order of the grouping variable. Consider manually reordering your factor levels if appropriate as shown in the second example below.

References

For more details, see- https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html

Examples

```
# for reproducibility
set.seed(123)

# -------------- between-subjects design -----------------------
# simple function call
statsExpressions::expr_t_nonparametric(
  data = sleep,
  x = group,
  y = extra
)

# creating a smaller dataset
msleep_short <- dplyr::filter(
  .data = ggplot2::msleep,
  vore %in% c("carni", "herbi")
)

# modifying few things
statsExpressions::expr_t_nonparametric(
  data = msleep_short,
  x = vore,
  y = sleep_rem,
  nboot = 200,
  conf.level = 0.99,
  conf.type = "bca"
)

# The order of the grouping factor matters when computing *V*
# Changing default alphabetical order manually
msleep_short$vore <- factor(msleep_short$vore, levels = c("herbi", "carni"))

# note the change in the reported *V* value but the identical
# value for *p* and the reversed effect size
statsExpressions::expr_t_nonparametric(
  data = msleep_short,
  x = vore,
  y = sleep_rem
```

```
)

# -------------- within-subjects design -----------------------

# using dataset included in the package
statsExpressions::expr_t_nonparametric(
  data = VR_dilemma,
  x = modality,
  y = score,
  paired = TRUE,
  conf.level = 0.90,
  conf.type = "perc",
  nboot = 200,
  k = 4
)
```

---

expr_t_onesample   *Expression for one sample* t-*test and its non-parametric and robust*
                            equivalents

Description

Expression for one sample *t*-test and its non-parametric and robust equivalents

Usage

```
expr_t_onesample(
  data,
  x,
  type = "parametric",
  test.value = 0,
  bf.prior = 0.707,
  robust.estimator = "onestep",
  effsize.type = "g",
  conf.level = 0.95,
  conf.type = "norm",
  nboot = 100,
  k = 2L,
  stat.title = NULL,
  ...
)
```

Arguments

data   A dataframe (or a tibble) from which variables specified are to be taken. A
                matrix or tables will **not** be accepted.

x   A numeric variable from the dataframe data.

type	Type of statistic expected (″parametric″ or ″nonparametric″ or ″robust″ or ″bayes″).Corresponding abbreviations are also accepted: ″p″ (for parametric), ″np″ (nonparametric), ″r″ (robust), or ″bf″resp.
test.value	A number specifying the value of the null hypothesis (Default: 0).
bf.prior	A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors.
robust.estimator	
	If type = ″robust″, a robust estimator to be used (″onestep″ (Default), ″mom″, or ″median″). For more, see ?WRS2::onesampb.
effsize.type	Type of effect size needed for *parametric* tests. The argument can be ″d″ (for Cohen's *d*) or ″g″ (for Hedge's *g*).
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (0.95).
conf.type	A vector of character strings representing the type of intervals required. The value should be any subset of the values ″norm″, ″basic″, ″perc″, ″bca″. For more, see ?boot::boot.ci.
nboot	Number of bootstrap samples for computing confidence interval for the effect size (Default: 100).
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
stat.title	A character describing the test being run, which will be added as a prefix in the subtitle. The default is NULL. An example of a stat.title argument will be something like ″Student's t-test: ″.
...	Additional arguments (currently ignored).

Value

Expression containing results from a one-sample test. The exact test and the effect size details contained will be dependent on the type argument.

References

For more details, see- https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html

Examples

```
# for reproducibility
set.seed(123)

# --------------------- parametric -------------------------------------

statsExpressions::expr_t_onesample(
  data = ggplot2::msleep,
  x = brainwt,
  test.value = 0.275,
  type = "parametric"
```

```
)

# --------------------- non-parametric ----------------------------------

statsExpressions::expr_t_onesample(
  data = ggplot2::msleep,
  x = brainwt,
  test.value = 0.275,
  type = "nonparametric"
)

# --------------------- robust -------------------------------------------

statsExpressions::expr_t_onesample(
  data = ggplot2::msleep,
  x = brainwt,
  test.value = 0.275,
  type = "robust"
)

# --------------------- Bayes Factor -------------------------------------

statsExpressions::expr_t_onesample(
  data = ggplot2::msleep,
  x = brainwt,
  test.value = 0.275,
  type = "bayes",
  bf.prior = 0.8
)
```

---

expr_t_parametric   *Making expression containing* t-*test results*

Description

Making expression containing *t*-test results

Usage

```
expr_t_parametric(
  data,
  x,
  y,
  paired = FALSE,
  effsize.type = "g",
  conf.level = 0.95,
  var.equal = FALSE,
  k = 2,
```

```
    stat.title = NULL,
    ...
)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. A matrix or tables will **not** be accepted.
x	The grouping variable from the dataframe data.
y	The response (a.k.a. outcome or dependent) variable from the dataframe data.
paired	Logical that decides whether the experimental design is repeated measures/within-subjects or between-subjects. The default is FALSE.
effsize.type	Type of effect size needed for *parametric* tests. The argument can be "d" (for Cohen's *d*) or "g" (for Hedge's *g*).
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (0.95).
var.equal	a logical variable indicating whether to treat the variances in the samples as equal. If TRUE, then a simple F test for the equality of means in a one-way analysis of variance is performed. If FALSE, an approximate method of Welch (1951) is used, which generalizes the commonly known 2-sample Welch test to the case of arbitrarily many samples.
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
stat.title	A character describing the test being run, which will be added as a prefix in the subtitle. The default is NULL. An example of a stat.title argument will be something like "Student's t-test: ".
...	Additional arguments (currently ignored).

Details

Cohen's *d* is calculated in the traditional fashion as the difference between means or mean minus *mu* divided by the estimated standardized deviation. By default Hedge's correction is applied (*N*-3)/(*N*-2.25) to produce *g*. For independent samples *t*-test, there are two possibilities implemented. If the *t*-test did not make a homogeneity of variance assumption, (the Welch test), the variance term will mirror the Welch test, otherwise a pooled and weighted estimate is used. If a paired samples *t*-test was requested, then effect size desired is based on the standard deviation of the differences.

The computation of the confidence intervals defaults to a use of non-central Student-*t* distributions.

When computing confidence intervals the variance of the effect size *d* or *g* is computed using the conversion formula reported in Cooper et al. (2009)

- `((n1+n2)/(n1*n2) + .5*d^2/df) * ((n1+n2)/df)` (independent samples)
- `sqrt(((1 / n) + (d^2 / n)) * 2 * (1 -r))` (paired case)

Value

Expression containing details from results of a two-sample test and effect size plus confidence intervals.

References

For more details, see- https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html

Examples

```
# for reproducibility
set.seed(123)

# creating a smaller dataset
msleep_short <- dplyr::filter(ggplot2::msleep, vore %in% c("carni", "herbi"))

# with defaults
statsExpressions::expr_t_parametric(
  data = msleep_short,
  x = vore,
  y = sleep_rem
)

# changing defaults (getting expression as output)
statsExpressions::expr_t_parametric(
  data = msleep_short,
  x = vore,
  y = sleep_rem,
  var.equal = TRUE,
  effsize.type = "d"
)
```

---

expr_t_robust   *Expression containing results from a robust* t-*test*

Description

Expression containing results from a robust *t*-test

Usage

```
expr_t_robust(
  data,
  x,
  y,
  tr = 0.1,
  paired = FALSE,
  conf.level = 0.95,
  nboot = 100,
  k = 2L,
  stat.title = NULL,
  ...
)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. A matrix or tables will **not** be accepted.
x	The grouping variable from the dataframe data.
y	The response (a.k.a. outcome or dependent) variable from the dataframe data.
tr	Trim level for the mean when carrying out robust tests. If you get error stating "Standard error cannot be computed because of Winsorized variance of 0 (e.g., due to ties). Try to decrease the trimming level.", try to play around with the value of tr, which is by default set to 0.1. Lowering the value might help.
paired	Logical that decides whether the experimental design is repeated measures/within-subjects or between-subjects. The default is FALSE.
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% lower and upper confidence intervals (0.95).
nboot	Number of bootstrap samples for computing confidence interval for the effect size (Default: 100).
k	Number of digits after decimal point (should be an integer) (Default: k = 2).
stat.title	A character describing the test being run, which will be added as a prefix in the subtitle. The default is NULL. An example of a stat.title argument will be something like "Student's t-test: ".
...	Additional arguments (currently ignored).

References

For more details, see- https://indrajeetpatil.github.io/statsExpressions/articles/stats_details.html

Examples

```
# for reproducibility
set.seed(123)

# between-subjects design ----------------------------------------

# with defaults
statsExpressions::expr_t_robust(
  data = sleep,
  x = group,
  y = extra
)

# changing defaults
statsExpressions::expr_t_robust(
  data = ToothGrowth,
  x = supp,
  y = len,
  nboot = 10,
```

```
  k = 3,
  tr = 0.2
)

# within-subjects design ---------------------------------------------
statsExpressions::expr_t_robust(
  data = dplyr::filter(bugs_long, condition %in% c("LDLF", "LDHF")),
  x = condition,
  y = desire,
  paired = TRUE,
  nboot = 25
)
```

iris_long	*Edgar Anderson's Iris Data in long format.*

Description

Edgar Anderson's Iris Data in long format.

Usage

```
iris_long
```

Format

A data frame with 600 rows and 5 variables

- id. Dummy identity number for each flower (150 flowers in total).
- Species. The species are *Iris setosa*, *versicolor*, and *virginica.*
- condition. Factor giving a detailed description of the attribute (Four levels: `"Petal.Length"`, `"Petal.Width"`, `"Sepal.Length"`, `"Sepal.Width"`).
- attribute. What attribute is being measured (`"Sepal"` or `"Pepal"`).
- measure. What aspect of the attribute is being measured (`"Length"` or `"Width"`).
- value. Value of the measurement.

Details

This famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are Iris setosa, versicolor, and virginica.

This is a modified dataset from `datasets` package.

Examples

```
dim(iris_long)
head(iris_long)
dplyr::glimpse(iris_long)
```

---

movies_long   *Movie information and user ratings from IMDB.com (long format).*

Description

Movie information and user ratings from IMDB.com (long format).

Usage

```
movies_long
```

Format

A data frame with 1,579 rows and 8 variables

- title. Title of the movie.
- year. Year of release.
- budget. Total budget (if known) in US dollars
- length. Length in minutes.
- rating. Average IMDB user rating.
- votes. Number of IMDB users who rated this movie.
- mpaa. MPAA rating.
- genre. Different genres of movies (action, animation, comedy, drama, documentary, romance, short).

Details

Modified dataset from ggplot2movies package.

The internet movie database, http://imdb.com/, is a website devoted to collecting movie data supplied by studios and fans. It claims to be the biggest movie database on the web and is run by amazon. More about information imdb.com can be found online, http://imdb.com/help/show_leaf?about, including information about the data collection process, http://imdb.com/help/show_leaf?infosource.

Movies were are identical to those selected for inclusion in movies_wide but this dataset has been constructed such that every movie appears in one and only one genre category.

Source

https://CRAN.R-project.org/package=ggplot2movies

Examples

```
dim(movies_long)
head(movies_long)
dplyr::glimpse(movies_long)
```

---

movies_wide   *Movie information and user ratings from IMDB.com (wide format).*

Description

Movie information and user ratings from IMDB.com (wide format).

Usage

```
movies_wide
```

Format

A data frame with 1,579 rows and 13 variables

- title. Title of the movie.
- year. Year of release.
- budget. Total budget in millions of US dollars
- length. Length in minutes.
- rating. Average IMDB user rating.
- votes. Number of IMDB users who rated this movie.
- mpaa. MPAA rating.
- action, animation, comedy, drama, documentary, romance, short. Binary variables representing if movie was classified as belonging to that genre.
- NumGenre. The number of different genres a film was classified in an integer between one and four

Details

Modified dataset from `ggplot2movies` package.

The internet movie database, http://imdb.com/, is a website devoted to collecting movie data supplied by studios and fans. It claims to be the biggest movie database on the web and is run by amazon. More information about imdb.com can be found online, http://imdb.com/help/show_leaf?about, including information about the data collection process, http://imdb.com/help/show_leaf?infosource.

Movies were selected for inclusion if they had a known length and had been rated by at least one imdb user. Small categories such as documentaries and NC-17 movies were removed.

Source

https://CRAN.R-project.org/package=ggplot2movies

Examples

```
dim(movies_wide)
head(movies_wide)
dplyr::glimpse(movies_wide)
```

Titanic_full   *Titanic dataset.*

Description

Titanic dataset.

Usage

```
Titanic_full
```

Format

A data frame with 2201 rows and 5 variables

- id. Dummy identity number for each person.
- Class. 1st, 2nd, 3rd, Crew.
- Sex. Male, Female.
- Age. Child, Adult.
- Survived. No, Yes.

Details

This data set provides information on the fate of passengers on the fatal maiden voyage of the ocean liner 'Titanic', summarized according to economic status (class), sex, age and survival.

This is a modified dataset from `datasets` package.

Examples

```
dim(Titanic_full)
head(Titanic_full)
dplyr::glimpse(Titanic_full)
```

VR_dilemma   *Virtual reality moral dilemmas.*

Description

Virtual reality moral dilemmas.

Usage

```
VR_dilemma
```

Format

A data frame with 68 rows and 4 variables

- id. Dummy identity number for each participant.
- order. The order in which the participants completed the two sessions: `"text_first"` (0) or `"text_second"` (1).
- modality. Describes how the moral dilemmas were presented to the participants: either in text format (`"text"`) or in Virtual Reality (`"vr"`).
- score. Proportion of "utilitarian" decisions. In other words, of the 4 decisions, how many affirmative were responses. Range: 0 (all utilitarian) - 1 (none utilitarian).

Details

Dataset from a study where participants completed identical moral dilemmas in two different sessions held on separate days: in one session, they read text description of the scenario, while in another session they completed the same scenarios in Virtual Reality (videos: https://www.youtube.com/watch?v=ebdU3HhhYs8). The study investigated if there was a discrepancy between how people judged the same scenarios while reading them in text versus experiencing them in virtual reality.

Source

https://psyarxiv.com/ry3ap/

Examples

```
dim(VR_dilemma)
head(VR_dilemma)
dplyr::glimpse(VR_dilemma)
```

Index