Package 'esci'

December 21, 2024

Type Package

Title Estimation Statistics with Confidence Intervals

Version 1.0.6

Description A collection of functions and 'jamovi' module for the estimation approach to inferential statistics, the approach which emphasizes effect sizes, interval estimates, and meta-analysis. Nearly all functions are based on 'statpsych' and 'metafor'. This package is still under active development, and breaking changes are likely, especially with the plot and hypothesis test functions. Data sets are included for all examples from Cumming & Calin-Jageman (2024) <ISBN:9780367531508>.

URL `https://github.com/rcalinjageman/esci/`, `https://rcalinjageman.github.io/esci/`

BugReports `https://github.com/rcalinjageman/esci/issues/`

License GPL-3

Encoding UTF-8

LazyData true

Imports jmvcore (>= 0.8.5), R6, rlang, methods, stats, multcomp, sadists, statpsych, metafor, ggplot2, ggdist, ggtext, ggbeeswarm, glue, Rdpack, stringr, mathjaxr, legendry (>= 0.2)

RoxygenNote 7.3.2

RdMacros Rdpack, mathjaxr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 2.10)

NeedsCompilation no

Author Robert Calin-Jageman [aut, cre, cph]

Maintainer Robert Calin-Jageman <`rcalinjageman@dom.edu`>

Repository CRAN

Date/Publication 2024-12-21 16:00:02 UTC

Contents

`CI_diamond_ratio` *Estimate the diamond ratio for a meta-analytic effect, a measure of heterogeneity*

Description

`CI_diamond_ratio` returns the diamond ratio and CI for a meta-analytic effect, the ratio of the random-effects CI width to the fixed-effects CI width. The diamond ratio is a measure of effect-size heterogeneity.

Usage

```
CI_diamond_ratio(RE, FE, vi, conf_level = 0.95)
```

Arguments

`RE`	metafor object with random effects result
`FE`	metafor object with fixed effects result
`vi`	vector of effect size variances
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.

Details

Calculation of the CI is based on code provided by Maxwell Cairns (see Cairns et al., 2022). Specifically, this function implements what Cairns et al (2022) called the Sub-Q approach, which provides the best CI coverage in simulations. For comparison, this function also returns the CI produced by the bWT-DL approach (which generally has worse performance).

Value

Returns a list with 3 properties:

- diamond_ratio
- LL - lower limit of the conf_level% CI, Sub-Q approach
- UL - upper limit of the conf_level% CI, Sub-Q approach
- LL_bWT_DL - lower limit of the conf_level% CI, bWT-DL approach
- UL_bWT_DL - upper limit of the conf_level% CI, bWT-DL approach

Source

Cairns, Maxwell, Geoff Cumming, Robert Calin-Jageman, and Luke A. Prendergast. “The Diamond Ratio: A Visual Indicator of the Extent of Heterogeneity in Meta-analysis.” *British Journal of Mathematical and Statistical Psychology* 75, no. 2 (May 2022): 201–19. https://doi.org/10.1111/bmsp.12258.

Examples

```
mydata <- esci::data_mccabemichael_brain

# Use esci to obtain effect sizes and sample variances, storing only raw_data
mydata <- esci::meta_mdiff_two(
  data = mydata,
  comparison_means = "M Brain",
  comparison_ns = "n Brain",
  comparison_sds = "s Brain",
  reference_means = "M No Brain",
  reference_ns = "n No Brain",
  reference_sds = "s No Brain",
  random_effects = FALSE
)$raw_data

# Conduct fixed effects meta-analysis
FE <- metafor::rma(
  data = mydata,
  yi = effect_size,
  vi = sample_variance,
  method="FE"
)
# Conduct random effect meta-analysis
RE <- metafor::rma(
  data = mydata,
  yi = effect_size,
  vi = sample_variance,
  method="DL"
)

# Get the diamond ratio
res <- esci::CI_diamond_ratio(
  RE = RE,
  FE = FE,
```

```
  vi = mydata$sample_variance
)
```

`CI_smd_ind_contrast` *Estimate standardized mean difference (Cohen's d) for an independent groups contrast*

Description

`CI_smd_ind_contrast` returns the point estimate and confidence interval for a standardized mean difference (smd aka Cohen's d aka Hedges g). A standardized mean difference is a difference in means standardized to a standard deviation:

$$d = \frac{\psi}{s}$$

Usage

```
CI_smd_ind_contrast(
  means,
  sds,
  ns,
  contrast,
  conf_level = 0.95,
  assume_equal_variance = FALSE,
  correct_bias = TRUE
)
```

Arguments

`means`	A vector of 2 or more means
`sds`	A vector of standard deviations, same length as means
`ns`	A vector of sample sizes, same length as means
`contrast`	A vector of group weights, same length as means
`conf_level`	The confidence level for the confidence interval, in decimal form. Defaults to 0.95.
`assume_equal_variance`	Defaults to FALSE
`correct_bias`	Defaults to TRUE; attempts to correct the slight upward bias in d derived from a sample. As of 8/9/2023 - Bias correction has been added for more than 2 groups when equal variance is not assumed, based on recent updates to statpsych

Value

Returns a list with these named elements:

- effect_size - the point estimate from the sample
- lower - lower bound of the CI
- upper - upper bound of the CI
- numerator - the numerator for Cohen's d_biased; the mean difference in the contrast
- denominator - the denominator for Cohen's d_biased; if equal variance is assumed this is sd_pooled, otherwise sd_avg
- df - the degrees of freedom used for correction and CI calculation
- se - the standard error of the estimate; **warning** not totally sure about this yet
- moe - margin of error; 1/2 length of the CI
- d_biased - Cohen's d without correction applied
- properties - a list of properties for the result
  Properties
- effect_size_name - if equal variance assumed d_s, otherwise d_avg
- effect_size_name_html - html representation of d_name
- denominator_name - if equal variance assumed sd_pooled otherwise sd_avg
- denominator_name_html - html representation of denominator name
- bias_corrected - TRUE/FALSE if bias correction was applied
- message - a message explaining denominator and correction status
- message_html - html representation of message

Details

It's a bit complicated:

A standardized mean difference turns out to be complicated.

First, it has many names:

- standardized mean difference (smd)
- Cohen's *d*
- When bias in a sample d has been corrected, also called Hedge's *g*

Second, the choice of the standardizer requires thought:

- sd_pooled - used when assuming all groups have exact same variance
- sd_avg - does not require assumption of equal variance
- other possibilities, too, but not dealt with in this function

The choice of standardizer is important, so it's noted in the subscript:

- d_s – assumes equal variance, standardized to sd_pooled
- d_avg - does not assume equal variance, standardized to sd_avg

A third complication is the issue of bias: d estimated from a sample has a slight upward bias at smaller sample sizes. With total sample size > 30, this slight bias becomes fairly negligible (kind of like the small upward bias in a sample standard deviation).

This bias can be corrected when equal variance is assumed or when the design of the study is simple (2 groups). For complex designs (>2 groups) without the assumption of equal variance, there is now also an approximate approach to correcting bias from Bonett.

Corrections for bias produce a long-run reduction in average bias. Corrections for bias are approximate.

How are d and its CI calculated?:

When equal variance is assumed:

When equal variance is assumed, the standardized mean difference is d_s, defined in Kline, p. 196:

$$d_s = \frac{\psi}{sd_{pooled}}$$

where psi is defined in Kline, equation 7.8:

$$\psi = \sum_{i=1}^{a} c_i M_i$$

and where sd_pooled is defined in Kline, equation 3.11

$$sd_{pooled} = \frac{\sum_{i=1}^{a}(n_i - 1)s_i^2}{\sum_{i=1}^{a}(n_i - 1)}$$

The CI for d_s is derived from lambda-prime transformation from Lecoutre, 2007 with code adapted from Cousineau & Goulet-Pelletier, 2020. Kelley, 2007 explains the general approach for linear contrasts.

This approach to generating the CI is 'exact', meaning coverage should be as desired *if* all assumptions are met (ha!).

Correction of upward bias can be applied.

When equal variance is not assumed:

When equal variance is not assumed, the standardized mean difference is d_avg, defined in Bonett, equation 6:

$$d_{avg} = \frac{\psi}{sd_{avg}}$$

Where sd_avg is the square root of the average of the group variances, as given in Bonett, explanation of equation 6:

$$sd_{avg} = \sqrt{\frac{\sum_{i=1}^{a} s_i^2}{a}}$$

If only 2 groups:

- The CI is derived from lambda-prime transformation using df and se from Huynh, 1989 – see especially Delacre et al., 2021
- This is also an 'exact' approach, and correction can be applied

If more than 2 groups:

- CI is approximated using the approach from Bonett, 2008
- An approximate correction developed by Bonett is used

References

- Bonett D. G. (2023). statpsych: Statistical Methods for Psychologists. R package version 1.4.0. https://dgbonett.github.io/statpsych
- Bonett, D. G. (2018). R code posted to personal website (now removed). Formally at https://people.ucsc.edu/~dgbonett/
- Bonett, D. G. (2008). Confidence Intervals for Standardized Linear Contrasts of Means. *Psychological Methods*, 13(2), 99–109. doi:10.1037/1082989X.13.2.99
- Cousineau & Goulet-Pelletier (2020) `https://osf.io/preprints/psyarxiv/s2597`
- Delacre et al., 2021, `https://osf.io/preprints/psyarxiv/tu6mp/`
- Huynh, C.-L. (1989). A unified approach to the estimation of effect size in meta-analysis. NBER Working Paper Series, 58(58), 99–104.
- Kelley, K. (2007). Confidence intervals for standardized effect sizes: Theory, application, and implementation. *Journal of Statistical Software, 20(8)*, 1–24. doi:10.18637/jss.v020.i08
- Lecoutre, B. (2007). Another Look at the Confidence Intervals for the Noncentral T Distribution. Journal of Modern Applied Statistical Methods, 6(1), 107–116. doi:10.22237/jmasm/1177992600

See Also

- `estimate_mdiff_ind_contrast` for friendly version that also returns raw score effect sizes for this design

Examples

```
# Example from Kline, 2013
#  Data in Table 3.4
#  Worked out in Chapter 7
#  See p. 202, non-central approach
# With equal variance assumed and no correction, should give:
#   d_s = -0.8528028 [-2.121155, 0.4482578]

res <- esci::CI_smd_ind_contrast(
  means = c(13, 11, 15),
  sds = c(2.738613, 2.236068, 2.000000),
  ns = c(5, 5, 5),
  contrast = contrast <- c(1, 0, -1),
  conf_level = 0.95,
  assume_equal_variance = TRUE,
  correct_bias = FALSE
)


# Example from [statpsych::ci.lc.stdmean.bs()] should give:
# Estimate        SE        LL         UL
# Unweighted standardizer: -1.273964 0.3692800 -2.025039 -0.5774878
# Weighted standardizer:   -1.273964 0.3514511 -1.990095 -0.6124317
# Group 1 standardizer:    -1.273810 0.4849842 -2.343781 -0.4426775
```

```
res <- esci::CI_smd_ind_contrast(
  means = c(33.5, 37.9, 38.0, 44.1),
  sds = c(3.84, 3.84, 3.65, 4.98),
  ns = c(10,10,10,10),
  contrast = c(.5, .5, -.5, -.5),
  conf_level = 0.95,
  assume_equal_variance = FALSE,
  correct_bias = TRUE
)
```

`CI_smd_one` *Estimate standardized mean difference (Cohen's d1) for a single group*

Description

`CI_smd_one` STILL NEEDS WORK TO VERIFY APPROACH FOR SE and MoE

Usage

```
CI_smd_one(mean, sd, n, reference_mean, conf_level = 0.95, correct_bias = TRUE)
```

Arguments

`mean`	Mean for a single group for the outcome measure
`sd`	Standard deviation, > 0
`n`	Sample size, an integer > 2
`reference_mean`	defaults to 0
`conf_level`	The confidence level for the confidence interval, in decimal form. Defaults to 0.95.
`correct_bias`	Defaults to TRUE

Value

Returns a list with these named elements:

- effect_size - the point estimate from the sample
- lower - lower bound of the CI
- upper - upper bound of the CI
- numerator - the numerator for Cohen's d_biased; the mean difference in the contrast
- denominator - the denominator for Cohen's d_biased; if equal variance is assumed this is sd_pooled, otherwise sd_avg
- df - the degrees of freedom used for correction and CI calculation

- se - the standard error of the estimate; **warning** not totally sure about this yet
- moe - margin of error; 1/2 length of the CI
- d_biased - Cohen's d without correction applied
- properties - a list of properties for the result

Properties

- effect_size_name - if equal variance assumed d_s, otherwise d_avg
- effect_size_name_html - html representation of d_name
- denominator_name - if equal variance assumed sd_pooled otherwise sd_avg
- denominator_name_html - html representation of denominator name
- bias_corrected - TRUE/FALSE if bias correction was applied
- message - a message explaining denominator and correction status
- message_html - html representation of message

Examples

```
# example code
res <- esci::CI_smd_one(24.5, 3.65, 40, 20)
```

`data_altruism_happiness`

Altruism Happiness - Ch12 - from Brethel-Haurwitz and Marsh (2014)

Description

Happiness may not be important just for the person feeling it; happiness may also promote kind, altruistic behavior. Brethel-Haurwitz and Marsh (2014) examined this idea by collecting data on U.S. states. A Gallup poll in 2010 was used to measure each state's well-being index, a measure of mean happiness for the state's residents on a scale from 0 to 100. Next, a kidney donation database for 1999-2010 was used to figure out each state's rate (number of donations per 1 million people) of non-directed kidney donations-that's giving one kidney to a stranger, an extremely generous and altruistic thing to do!

Usage

```
data_altruism_happiness
```

Format

`data_altruism_happiness`**:**

A data frame with 50 rows and 6 columns:

State	factor - State where data was collected
Abbreviation	factor - State where data was collected
Well_Being_2010	numeric - State where data was collected
Well_Being_2013	numeric - State where data was collected
Kidney_Rate, per million population	numeric - State where data was collected
WB Change 2013-2010	numeric - State where data was collected

Source

doi:10.1177/0956797613516148

`data_anchor_estimate_ma`

Anchor Estimate ma - Ch9 - Many Labs replications of Jacowitz and Kahneman (1995)

Description

To what extent does the wording of a question influence one's judgment? In their classic study, Jacowitz and Kahneman (1995) asked participants to estimate how many babies are born each day in the United States. Participants were given either a low anchor (more than 100 babies/day) or a high anchor (less than 50,000 babies/day). Those who saw the low anchor estimated many fewer births/day than those who saw the high anchor, which suggests that the wording can have a profound influence. The correct answer, as it happens, is ~11,000 births/day in 2014. To investigate the extent that these results are replicable, the Many Labs project repeated this classic study at many different labs around the world. You can find the summary data for 30 of these labs in the Anchor Estimate ma data file

Usage

```
data_anchor_estimate_ma
```

Format

`data_anchor_estimate_ma`**:**

A data frame with 30 rows and 9 columns:

Location	factor
M Low	numeric
s Low	numeric
n Low	integer
M High	numeric

- **s High** numeric
- **n High** integer
- **USAorNot** factor
- **Country** factor

Source

doi:10.1027/18649335/a000178

`data_basol_badnews` *Basol badnews - Ch07 - from Basol et al. (2020)*

Description

Climate change? Vaccines? Fake news and conspiracy theories on these and numerous other issues can be highly damaging, but are thriving in this social media age. Trying to debunk a conspiracy theory by presenting facts and evidence often doesn't work, alas. Psychological inoculation, also similar to prebunking, presents a mild form of misinformation, preferably with explanation, in the hope of building resistance to real-life fake news-a sort of vaccine for fake news. The Bad News game is a spin-off from research on psychological inoculation. Basol et al. (2020) assessed the possible effectiveness of this game as a fake news vaccine. At getbadnews.com you can click 'About' for information, or just start playing the game- it's easy and maybe even fun. You encounter mock Twitter (now X) fake news messages that illustrate common strategies for making fake news memorable or believable. You make choices between messages and decide which ones to 'forward' as you try to spread fake news while building your credibility score and number of 'followers'- rather like real life for a conspiracy theorist wanting to spread the word. Compete with your friends for credibility and number of followers. Basol's online participants first saw 18 fictitious fake news tweets and rated each for reliability (accuracy, believability), and also rated their confidence in that reliability rating. Both ratings were on a 1 to 7 scale. Those in the BadNews group then played the game for 15 minutes, whereas those in the Control group played Tetris. Then all once again gave reliability and confidence ratings for the 18 tweets.

Usage

```
data_basol_badnews
```

Format

`data_basol_badnews`**:**

A data frame with 198 rows and 3 columns:

- **Diff reliability** numeric
- **Diff confidence** numeric
- **Condition** factor

Source

doi:10.5334/joc.91

`data_bem_psychic` *Bem Psychic - Ch13 - from Bem and Honorton (1994)*

Description

Daryl Bem was an experienced mentalist and research psychologist, who, a decade earlier, had been one of several outside experts invited to scrutinize the laboratory and experimental procedures of parapsychology researcher Charles Honorton. Bem not only judged them adequate, but joined the research effort and became a coauthor with Honorton. Bem and Honorton (1994) first reviewed early ganzfeld studies and described how the experimental procedure had been improved to reduce the chance that results could be influenced by various possible biases, or leakages of information from sender to receiver. For example, the randomization procedure was carried out automatically by computer, and all stimuli were presented under computer control. Bem and Honorton then presented data from studies conducted with the improved procedure. Table 13.1 presents basic data from 10 studies reported by Bem and Honorton (1994). Participants each made a single judgment, so in Pilot 1, for example, 22 participants responded, with 8 of them giving a correct response. Three pilot studies helped refine the procedures, then four studies used novice receivers. Study 5 used 20 students of music, drama, or dance as receivers, in response to suggestions that creative people might be more likely to show telepathy. Studies 6 and 7 used receivers who had participated in an earlier study. The proportion of hits expected by chance is .25, and Table 13.1 shows that all but Study 1 found proportions higher than .25.

Usage

```
data_bem_psychic
```

Format

`data_bem_psychic`**:**

A data frame with 10 rows and 5 columns:

Study factor

Participants factor

N(Trials) integer

N(Hits) integer

Proportion Hits numeric

Source

doi:10.1037/00332909.115.1.25

`data_bodywellf` *BodyWellF - Ch12 - Body Satisfaction and Well-being data for females from Figure 11.24 right panel*

Description

A subset of data_bodywell_fm, reports only for those participants who identified as female. Data is Subjective Wellbeing and Body Satisfaction.

Usage

`data_bodywellf`

Format

`data_bodywellf`**:**

A data frame with 59 rows and 2 columns:

Body Satisfaction numeric

Well-being numeric

`data_bodywellfm` *BodyWellFM - Ch12 - Body Satisfaction and Well-being data from Figure 11.1*

Description

Survey data from a convenience sample Dominican University students. Reported are measures of Subjective Wellbeing abd Body Satisfaction.

Usage

`data_bodywellfm`

Format

`data_bodywellfm`**:**

A data frame with 106 rows and 2 columns:

Body Satisfaction numeric

Well-being numeric

`data_bodywellm` *BodyWellM - Ch12 - Body Satisfaction and Well-being data for males from Figure 11.24 left panel*

Description

A subset of data_bodywell_fm, reports only for those participants who identified as male. Data is Subjective Wellbeing abd Body Satisfaction.

Usage

```
data_bodywellm
```

Format

`data_bodywellm`**:**

A data frame with 47 rows and 2 columns:

Body Satisfaction numeric

Well-being numeric

`data_campus_involvement`

Campus Involvement - Ch11 - for End-of-Chapter Exercise 11.7

Description

Clinton conducted a survey of college students to determine the extent to which subjective well-being is related to campus involvement (Campus Involvement data set on the book website). Participants completed a measure of subjective well-being (scale from 1 to 5) and a measure of campus involvement (scale from 1 to 5). Participants also reported gender (male or female) and commuter status (resident or commuter). Synthetic data simulated to mimic survey data from a class project.

Usage

```
data_campus_involvement
```

Format

`data_campus_involvement`**:**

A data frame with 113 rows and 6 columns:

ID integer

Gender factor

GPA numeric

CommuterStatus factor

SWB numeric

Campus Involvement numeric

`data_chap_8_paired_ex_8.18`
Fictitious *data from an unrealistically small HEAT study comparing scores for a single group of students before and after a workshop on climate change.*

Description

Fictitious data from an unrealistically small HEAT study comparing scores for a single group of students before and after a workshop on climate change.

Usage

```
data_chap_8_paired_ex_8.18
```

Format

`data_chap_8_paired_ex_8.18`**:**

A data frame with 8 rows and 2 columns:

Before numeric

After numeric

`data_clean_moral` *Clean moral - Ch07 - from Schnall et al. (2008), Study 1, and Johnson et al. (2014)*

Description

Some researchers claim that moral judgments are based not only on rational considerations but also on one's current emotional state. To what extent can recent emotional experiences influence moral judgments? Schnall et al. (2008) examined this question by manipulating feelings of cleanliness and purity and then observing the extent that this changes how harshly participants judge the morality of others. Inscho Study 1, Schnall et al. asked participants to complete a word scramble task with either neutral words (neutral prime) or words related to cleanliness (cleanliness prime). All students then completed a set of moral judgments about controversial scenarios: Moral judgment is the average of six items, each rated on a scale from 0 to 9, with high meaning harsh. The data from this study are in the Clean moral file, which also contains data from a replication by Johnson et al. (2014)

Usage

```
data_clean_moral
```

Format

`data_clean_moral`**:**
A data frame with 208 rows and 4 columns:

Schnall Condition factor
Schnall Moral judgment numeric
Johnson Condition factor
Johnson Moral judgment numeric

Source

doi:10.1027/18649335/a000186

`data_college_survey_1` *College survey 1 - Ch03 - for End-of-Chapter Exercise 3.3*

Description

Data from an additional survey of Dominican University students; reports various psychological and behavioral measures.

Usage

```
data_college_survey_1
```

Format

`data_college_survey_1`**:**
A data frame with 243 rows and 23 columns:

ID integer
Gender factor
Gender_Code factor
Age integer
Shool_Year factor
School_Year_Code factor
Transfer factor
Transfer_Code logical
Student_Athlete factor
Student_Athlete_Code logical
Wealth_SR numeric
GPA numeric
ACT integer
Subjective_Well_Being numeric
Positive_Affect numeric

Negative_Affect numeric
Relationship_Confidence numeric
Exercise numeric
Academic_Motivation_Intrinsic numeric
Academic_Motivation_Extrinsic numeric
Academic_Motivation_Amotivation numeric
Intelligence_Value numeric
Raven_Score numeric

`data_college_survey_2` *College survey 2 - Ch05 - for End-of-Chapter Exercise 5.4*

Description

College survey 2 - Ch05 - for End-of-Chapter Exercise 5.4

Usage

```
data_college_survey_2
```

Format

`data_college_survey_2`**:**
A data frame with 138 rows and 17 columns:

ID integer
Gender factor
Gender_Code factor
Age numeric
Wealth_SR numeric
School_Year factor
School_Year_Code factor
Transfer factor
Transfer_Code logical
GPA numeric
Subjective_Well_Being numeric
Positive_Affect numeric
Negative_Affect numeric
Academic_Engagement numeric
Religious_Meaning numeric
Health numeric
Emotion_Recognition factor

`data_damischrcj` *DamischRCJ - Ch9 - from 6 Damisch studies, and Calin-Jageman and Caldwell (2014)*

Description

DamischRCJ - Ch9 - from 6 Damisch studies, and Calin-Jageman and Caldwell (2014)

Usage

```
data_damischrcj
```

Format

`data_damischrcj`**:**

A data frame with 8 rows and 5 columns:

Study factor

Cohen's d unbiased numeric

n Control integer

n Lucky integer

Research Group factor

`data_effronraj_fakenews`

EffronRaj fakenews - Ch8 - from Effron and Raj (2020), v1.1

Description

Synthetic data meant to represent Experiment 1 of Effron & Raj, 2020. 138 U.S. adults, recruited in August 2018 on Prolific Academic, worked online. First, they saw six fake headlines four times, each time being asked to rate how interesting/engaging/ funny/well-written the headline was. This rating task simply ensured that the participants paid some attention to each headline. The stimuli were 12 actual fake-news headlines about American politics, with accompanying photographs. Half appealed to Republicans and half to Democrats. Later, 12 fake headlines were presented one at a time, a random mix of the six Old headlines-those seen before-and six New headlines not seen previously. It was stated very clearly that independent, non-partisan fact-checking had established that all the headlines were not true. Participants first rated, on a 0 (not at all) to 100 (extremely) scale, the degree to which to which they judged it unethical to publish that headline. That's the Unethicality DV. They also rated how likely they would be to share the headline if they saw it posted by an acquaintance on social media; there were three further similar ratings. Finally, they rated how accurate they believed the headline to be.

Usage

```
data_effronraj_fakenews
```

Format

`data_effronraj_fakenews`**:**

A data frame with 138 rows and 5 columns:

ID factor

UnethOld numeric

UnethNew numeric

AccurOld numeric

AccurNew numeric

Details

v1.1 – Participant 46 had an invalid negative value for UnethOld; replaced with 0.

Source

doi:10.1177/0956797619887896

`data_emotion_heartrate`

Emotion heartrate - Ch8 - from Lakens (2013)

Description

Anger is a powerful emotion. To what extent can feeling angry actually change your heart rate? To investigate, Lakens (2013) asked students to record their heart rate (in beats per minute) at rest before (baseline) and then while recalling a time of intense anger. This is a conceptual replication of a classic study by Ekman et al. (1983). Load the Emotion heartrate data set from the book website.

Usage

```
data_emotion_heartrate
```

Format

`data_emotion_heartrate`**:**

A data frame with 68 rows and 3 columns:

ID integer

HR_baseline numeric

HR_anger numeric

Source

doi:10.1109/TAFFC.2013.3

`data_exam_scores` *Exam Scores - Ch11 - for End-of-Chapter Exercise 11.2*

Description

To what extent does initial performance in a class relate to performance on a final exam? First exam and final exam scores for nine students enrolled in an introductory psychology course. Exam scores are percentages, where 0 = no answers correct and 100 = all answers correct. Data is synthetic to represent patterns found in a previous psych stats course.

Usage

```
data_exam_scores
```

Format

`data_exam_scores`:

A data frame with 9 rows and 3 columns:

StudentID factor

First Exam numeric

Final Exam numeric

`data_flag_priming_ma` *Flag Priming ma - Ch9 - Many Labs replications of Carter et al. (2011)*

Description

To what extent does being exposed to the American flag influence political attitudes? One seminal study (Carter et al., 2011) explored this issue by subtly exposing participants either to images of the American flag or to control images. Next, participants were asked about their political attitudes, using a 1-7 rating scale where high scores indicate conservative attitudes. Participants exposed to the flag were found to express substantially more conservative attitudes. The Many Labs project replicated this finding at 25 different locations in the United States.

Usage

```
data_flag_priming_ma
```

Format

`data_flag_priming_ma`**:**

A data frame with 25 rows and 7 columns:

Location factor

M Flag numeric

s Flag numeric

n Flag integer

M Noflag numeric

s Noflag numeric

n Noflag integer

Source

doi:10.5334/jopd.ad

`data_gender_math_iat` *Gender math IAT - Ch07 - Ithaca and SDSU replications of Nosek et al. (2002)*

Description

To what extent do men and women differ in their attitudes towards mathematics? To investigate, Nosek et al. (2002) asked male and female students to complete an Implicit Association Test (IAT)- this is a task designed to measure a participant's implicit (non-conscious) feelings towards a topic. (If you've never heard of the IAT, try it out here: tiny.cc/harvardiat) On this IAT, students were tested for negative feelings towards mathematics and art. Scores reflect the degree to which a student had more negative implicit attitudes about mathematics than art (positive score: more negative feelings about mathematics; 0: same level of negativity to both; negative score: more negative feelings about art). data_gender_math_iat has data from two labs that participated in a large-scale replication of the original study (Klein et al., 2014a, 2014b)

Usage

```
data_gender_math_iat
```

Format

`data_gender_math_iat`**:**

A data frame with 155 rows and 4 columns:

Ithaca gender factor

Ithaca IAT numeric

SDSU gender factor

SDSU IAT numeric

Source

doi:10.1027/18649335/a000178

`data_gender_math_iat_ma`

Gender math IAT ma - Ch9 - Many Labs replications of Nosek et al. (2002)

Description

In EOC Exercise 4 in Chapter 7 we encountered the classic study of Nosek et al. (2002), in which male and female participants completed an Implicit Association Test (IAT) that measured the extent of negative attitudes towards mathematics, compared with art. The study found that women, compared with men, tended to have more negative implicit attitudes towards mathematics. The Many Labs project repeated this study at locations around the world (Klein et al., 2014a, 2014b). Summary data for 30 of these labs are available in Gender math IAT ma. Higher scores indicate more implicit bias against mathematics. See also data_gender_math_iat for raw data from two specific sites from this replication effort.

Usage

```
data_gender_math_iat_ma
```

Format

`data_gender_math_iat_ma`**:**

A data frame with 30 rows and 9 columns:

Location	factor
M Male	numeric
s Male	numeric
n Male	integer
M Female	numeric
s Female	numeric
n Female	integer
USAorNot	factor
Country	factor

Source

doi:10.1027/18649335/a000178

`data_halagappa` *Halagappa - Ch14 - from Halagappa et al. (2007)*

Description

Could eating much less delay Alzheimer's? If so, that would be great news. Halagappa et al. (2007) investigated the possibility by using a mouse model, meaning they used Alzheimer-prone mice, which were genetically predisposed to develop neural degeneration typical of Alzheimer's. The researchers used six independent groups of mice, three tested in mouse middle age when 10 months old, and three in mouse old age when 17 months. At each age there was a control group of normal mice that ate freely (the NFree10 and NFree17 groups), a group of Alzheimer-prone mice that also ate freely (the AFree10 and AFree17 groups), and another Alzheimer-prone group restricted to 40% less food than normal (the ADiet10 and ADiet17 groups). Table 14.2 lists the factors that define the groups, and group labels. I'll discuss one measure of mouse cognition: the percent time spent near the target of a water maze, with higher values indicating better learning and memory. Table 14.2 reports the means and standard deviations for this measure, and group sizes.

Usage

`data_halagappa`

Format

`data_halagappa`**:**

A data frame with 6 rows and 4 columns:

Groups factor

Mean numeric

SD numeric

n integer

Source

doi:10.1016/j.nbd.2006.12.019

`data_home_prices` *Home Prices - Ch12 - for End-of-Chapter Exercise 12.2*

Description

Maybe you're thinking about buying a house after college? Regression can help you hunt for a bargain. Download the Home Prices data set. This file contains real estate listings from 1997 to 2003 in a city in California. Let's explore the extent to which the size of the home (in square meters) predicts the sale price.

Usage

`data_home_prices`

Format

`data_home_prices`**:**
A data frame with 300 rows and 8 columns:

MLS integer
Location factor
Price integer
Bedrooms integer
Bathrooms integer
Size (m2) numeric
Status factor
Status_Code logical

`data_kardas_expt_3` *Kardas Expt 3 - Ch07 - from Kardas and O'Brien (2018), Experiment 3*

Description

Suppose you want to change the battery in your phone, cook the perfect souffle, or perform a three-ball juggle. Just as numerous people do every day, you might search online to find a video that shows what to do. Suppose you watch such a video just once. First question: How well would you then predict you could perform the task? Second question: How well would you actually perform the task, the first time you tried? Now suppose you watch the video many times: Again consider the two questions. These questions were investigated in a series of studies by Kardas and O'Brien (2018). Let's first do some quick analyses of Kardas Experiments 3 and 4-let's call them Expt 3 and Expt 4-focusing on the effect of watching a video many times rather than once. In Expt 3, participants first watched a brief video of a person performing the moonwalk. The Low Exposure group watched the video once, the High Exposure group 20 times. Then participants predicted, on a 1 to 10 scale, how well they felt they would be able to perform the moonwalk themselves. Finally, they attempted a single performance of the moonwalk, which was videoed. These videos were rated, on the same 1 to 10 scale, by independent raters.

Usage

`data_kardas_expt_3`

Format

`data_kardas_expt_3`**:**
A data frame with 100 rows and 3 columns:

Exposure factor
Prediction numeric
Performance numeric

Source

doi:10.1177/0956797617740646

`data_kardas_expt_4` *Kardas Expt 4 - Ch07 - from Kardas and O'Brien (2018), Experiment 4*

Description

Suppose you want to change the battery in your phone, cook the perfect souffle, or perform a three-ball juggle. Just as numerous people do every day, you might search online to find a video that shows what to do. Suppose you watch such a video just once. First question: How well would you then predict you could perform the task? Second question: How well would you actually perform the task, the first time you tried? Now suppose you watch the video many times: Again consider the two questions. These questions were investigated in a series of studies by Kardas and O'Brien (2018). Let's first do some quick analyses of Kardas Experiments 3 and 4-let's call them Expt 3 and Expt 4-focusing on the effect of watching a video many times rather than once. Expt 4 was conducted online with participants recruited from Amazon's Mechanical Turk, who are typically more diverse than students. The online task was based on a mirror-drawing game developed by Bob and students (Cusack et al., 2015, tiny.cc/bobmirrortrace). Participants first read a description of the game and the scoring procedure. To play, you use your computer trackpad to trace a target line, as accurately and quickly as you can. The task is tricky because you can see only a mirror image of the path you are tracing with a finger on the trackpad. A running score is displayed. The final score is the percentage match between the target line and the path you traced, so scores can range from 0 to 100

Usage

```
data_kardas_expt_4
```

Format

`data_kardas_expt_4`**:**

A data frame with 270 rows and 4 columns:

Exposure factor

Prediction integer

Performance integer

Time numeric

Source

doi:10.1177/0956797617740646

`data_labels_flavor` *Labels flavor - Ch8 - from Floretta-Schiller et al. (2015)*

Description

To what extent do brand labels influence perceptions of a product? To investigate, participants were asked to participate in a taste test. All participants were actually given the same grape juice, but one glass was poured from a bottle labeled 'Organic' and the other glass from a bottle labeled 'Generic'. After each tasting (in counterbalanced order), participants were asked to rate how much they enjoyed the juice on a scale from 1 (not at all) to 10 (very much). Participants were also asked to say how much they'd be willing to pay for a large container of that juice on a scale from $1 to $10. Load the Labels flavor data set from the book website. These data were collected as part of a class project by Floretta-Schiller et al. (2015), whose work was inspired by a very clever study looking at the effects of fast-food wrappers on children's enjoyment of food (Robinson et al., 2007).

Usage

```
data_labels_flavor
```

Format

`data_labels_flavor`**:**

A data frame with 51 rows and 6 columns:

ParticipantID	integer
Enjoy_Generic	numeric
Enjoy_Organic	numeric
Pay_Generic	numeric
Pay_Organic	numeric
Suspicious	logical

`data_latimier_3groups` *Latimier 3Groups - Ch14 - 3 groups in Latimier et al. (2019)*

Description

The researchers were interested in how different study approaches might impact learning. Working in France, they created three independent groups, each comprising 95 adults. Participants worked online through seven learning modules about DNA. The Reread group worked through a module, then worked through it a second time before going on to the next module. The Quiz group worked through a module, then had to complete a quiz before going on to the next module. The Prequiz group had to work through the quiz before seeing the presentation of a module, then went on to the quiz and presentation of the next module. Participants received feedback and a brief explanation after answering each question in a quiz, and could take as long as they wished to work through each module and quiz. Seven days later, participants completed a final test. data_latimier_3groups is the full data set. To facilitate different student exercises, there are also separate data entities for each group (data_latimier_prequiz, data_latimier_reread, etc.), and for every *pair* of groups (data_latimier_quiz_prequiz, etc.).

Usage

```
data_latimier_3groups
```

Format

`data_latimier_3groups`**:**

A data frame with 285 rows and 3 columns:

Group factor

Test% numeric

Time numeric

Source

doi:10.7910/DVN/XPYPMF

`data_latimier_prequiz` *Latimier Prequiz - Ch03 - Prequiz group in Latimier et al. (2019)*

Description

Just the Prequiz group from Latimier et al., 2019 See full details in data_latimier_3_groups

Usage

```
data_latimier_prequiz
```

Format

`data_latimier_prequiz`**:**

A data frame with 95 rows and 3 columns:

Group factor

Test% numeric

TIme numeric

Source

doi:10.7910/DVN/XPYPMF

`data_latimier_quiz` *Latimier Quiz - Ch03 - Quiz group in Latimier et al. (2019)*

Description

Just the Quiz group from Latimier et al., 2019 See full details in data_latimier_3_groups

Usage

```
data_latimier_quiz
```

Format

`data_latimier_quiz`**:**
A data frame with 95 rows and 3 columns:

Group factor
Test% numeric
Time numeric

Source

doi:10.7910/DVN/XPYPMF

`data_latimier_quiz_prequiz` *Latimier Quiz Prequiz - Ch07 - Quiz and Prequiz groups in Latimier et al. (2019)*

Description

Just the Quiz (RQ) an Prequiz (QR) groups from Latimier et al., 2019 See full details in data_latimier_3_groups

Usage

```
data_latimier_quiz_prequiz
```

Format

`data_latimier_quiz_prequiz`**:**
A data frame with 190 rows and 3 columns:

Group factor
Test% numeric
Time numeric

Source

doi:10.7910/DVN/XPYPMF

`data_latimier_reread` *Latimier Reread - Ch03 - Reread group in Latimier et al. (2019)*

Description

Just the Reread group from Latimier et al., 2019 See full details in data_latimier_3_groups

Usage

```
data_latimier_reread
```

Format

`data_latimier_reread`**:**
A data frame with 95 rows and 3 columns:

Group factor
Test% numeric
Time numeric

Source

doi:10.7910/DVN/XPYPMF

`data_latimier_reread_prequiz` *Latimier Reread Prequiz - Ch07 - Reread and Prequiz groups in Latimier et al. (2019)*

Description

Just the Reread (RR) an Prequiz (QR) groups from Latimier et al., 2019 See full details in data_latimier_3_groups

Usage

```
data_latimier_reread_prequiz
```

Format

`data_latimier_reread_prequiz`**:**
A data frame with 190 rows and 3 columns:

Group factor
Test% numeric
Time numeric

Source

doi:10.7910/DVN/XPYPMF

`data_latimier_reread_quiz` *Latimier Reread Quiz - Ch07 - Reread and Quiz groups in Latimier et al. (2019)*

Description

Just the Reread Quiz groups from Latimier et al., 2019 See full details in data_latimier_3_groups

Usage

```
data_latimier_reread_quiz
```

Format

`data_latimier_reread_quiz`**:**
A data frame with 190 rows and 3 columns:

Group factor
Test% numeric
Time numeric

Source

doi:10.7910/DVN/XPYPMF

`data_macnamara_r_ma` *Macnamara r ma - Ch11 - from Macnamara et al. (2014)*

Description

Is genius born or made? Could any of us be Michael Jordan or Mozart if we worked sufficiently hard to develop the requisite skills? Meta-analysis of correlations can help answer such questions. The issue here is the extent that practice and effort may be sufficient for achieving the highest levels of expertise. Ericsson et al. (1993) argued that years of effort is what matters most: 'Many characteristics once believed to reflect innate talent are actually the result of intense practice extended for a minimum of 10 years' (p. 363). This view was enormously popularized by Malcolm Gladwell (2008), who argued in his book Outliers that 10,000 hours of focused practice is the key to achieving expertise. However, this view is now being challenged, with one important contribution being a large meta-analysis of correlations between amount of intense practice and level of achievement: Macnamara et al. (2014) combined 157 correlations reported in a wide range of fields, from sports to music and education, and found correlation of only r = .35 (.30, .39). Table 11.1 shows the 16 main correlations for music, from Macnamara et al. (2014).

Usage

```
data_macnamara_r_ma
```

Format

`data_macnamara_r_ma`:

A data frame with 16 rows and 4 columns:

Study factor

r numeric

N integer

Instrument_Type factor

Source

doi:10.1177/0956797614535810

`data_mccabemichael_brain`

McCabeMichael brain - Ch9 - from Michael et al. (2013)

Description

You’ve probably seen cross sections of the brain with brightly colored areas indicating which brain regions are most active during a particular type of cognition or emotion. Search online for fMRI (functional magnetic resonance imaging) brain scans to see such pictures and learn how they are made. They can be fascinating-are we at last able to see how thinking works? In 2008, McCabe and Castel published studies that investigated how adding a brain picture might alter judgments of the credibility of a scientific article. For one group of participants, an article was accompanied by a brain image that was irrelevant to the article. For a second, independent group, there was no image. Participants read the article, then gave a rating of the statement ’The scientific reasoning in the article made sense’. The response options were 1 (strongly disagree), 2 (disagree), 3 (agree), and 4 (strongly agree). The researchers reported that mean ratings were higher with a brain picture than without, but that the difference was small. It seemed that an irrelevant brain picture may have some, but only a small influence. The authors drew appropriately cautious conclusions, but the result quickly attracted attention and there were many media reports that greatly overstated it. At least according to the popular media, it seemed that adding a brain picture made any story convincing. Search on ’McCabe seeing is believing’, or similar, to find media reports and blog posts. Some warned readers to watch out for brain pictures, which, they said, can trick you into believing things that aren’t true. The result intrigued some New Zealander colleagues of mine who discovered that, despite its wide recognition, the finding hadn’t been replicated. They ran replication studies using the materials used by the original researchers, and found generally small ESs. I joined the team at the data analysis stage and the research was published (Michael et al., 2013). I’ll discuss here a meta-analysis of two of the original studies and eight replications by our team. The studies were sufficiently similar for meta-analysis, especially considering that all the Michael studies were designed to have many features that matched the original studies. This data set does *not* include two additional critique studies run by the Michael team. See also data_mccabemichael_brain2

Usage

```
data_mccabemichael_brain
```

Format

`data_mccabemichael_brain`**:**
A data frame with 10 rows and 9 columns:

Study name factor
M No Brain numeric
s No Brain numeric
n No Brain numeric
M Brain numeric
s Brain numeric
n Brain numeric
SimpleCritique factor
Research group factor

Source

doi:10.3758/s1342301303916

`data_mccabemichael_brain2`

McCabeMichael brain2 - Ch9 - from Michael et al. (2013)

Description

Same as data_mccabemichael_brain but includes two additional critique studies run by the Michael team.

Usage

```
data_mccabemichael_brain2
```

Format

`data_mccabemichael_brain2`**:**
A data frame with 12 rows and 9 columns:

Study name factor
M No Brain numeric
s No Brain numeric
n No Brain numeric
M Brain numeric
s Brain numeric
n Brain numeric
SimpleCritique factor
Research group factor

Source

doi:10.3758/s1342301303916

`data_meditationbrain` *MeditationBrain - Ch15 - from Holzel et al. (2011)*

Description

My example is a well-known study of mindfulness meditation by Holzel et al. (2011). People who wanted to reduce stress, and were not experienced meditators, were assigned to a Meditation (n = 16) or a Control (n = 17) group. The Meditation group participated in 8 weeks of intensive training and practice of mindfulness meditation. The researchers used a questionnaire to assess a range of emotional and cognitive variables both before (Pretest) and after (Posttest) the 8-week period. All assessment was conducted while the participants were not meditating. The study is notable for including brain imaging to assess possible changes in participants' brains from Pretest to Posttest. The researchers measured gray matter concentration, which increases in brain regions that experience higher and more frequent activation. The researchers expected that the hippocampus may be especially responsive to meditation because it has been implicated in the regulation of emotion, arousal, and general responsiveness. They therefore included in their planned analysis the assessment of any changes to gray matter concentration in the hippocampus.

Usage

```
data_meditationbrain
```

Format

`data_meditationbrain`**:**

A data frame with 33 rows and 7 columns:

Pretest numeric

Posttest numeric

Group factor

ControlPre numeric

ControlPost numeric

MeditationPre numeric

MeditationPost numeric

Source

doi:10.1007/9789400720794_9

`data_organicmoral` *OrganicMoral - Ch14 - from Eskine (2013)*

Description

To what extent might choosing organic foods make us morally smug? To investigate, Eskine (2013) asked participants to rate images of organic food, neutral (control) food, or comfort food. Next, under the guise of a different study, all participants completed a moral judgment scale in which they read different controversial scenarios and rated how morally wrong they judged them to be (scale of 1-7, high judgments mean more wrong). Table 14.7 shows summary data, which are also available in the first four variables in the OrganicMoral file. In that file you can see two further variables, which report full data-we'll come to these shortly. Here we use the summary data. After the results of Eskine (2013) were published, Moery and Calin-Jageman (2016) conducted a series of close replications. We obtained original materials from Eskine, piloted the procedure, and preregistered our sampling and analysis plan. The OSF page, osf.io/atkn7, has all the details. The data from one of these close replications are in the last two variables of the OrganicMoral file. For this replication study, group names are in the variable ReplicationGroup and moral judgments in MoralJudgment. (You may need to scroll right to see these variables.)

Usage

```
data_organicmoral
```

Format

`data_organicmoral`**:**

A data frame with 106 rows and 6 columns:

Group factor

Mean numeric

SD numeric

N integer

ReplicationGroup factor

MoralJudgmentment numeric

Source

doi:10.1177/1948550616639649

`data_penlaptop1` *% transcription scores from pen and laptop group of Meuller et al., 2014*

Description

% transcription scores from pen and laptop group of Meuller et al., 2014

Usage

```
data_penlaptop1
```

Format

`data_penlaptop1`**:**

A data frame with 65 rows and 2 columns:

condition factor

transcription numeric

Source

doi:10.1177/0956797614524581

`data_powerperformance_ma` *PowerPerformance ma - Ch9 - from Burgmer and Englich (2012), and Cusack et al. (2015)*

Description

To what extent could feeling powerful affect your performance at motor skills? To investigate, Burgmer and Englich (2012) assigned German participants to either power or control conditions and then asked them to play golf (Experiment 1) or darts (Experiment 2). They found that participants manipulated to feel powerful performed substantially better than those in the control condition. To study this finding further, Cusack et al. (2015) conducted five replications in the United States. Across these replications they tried different ways of manipulating power, different types of tasks (golf, mirror tracing, and a cognitive task), different levels of difficulty, and different types of participant pools (undergraduates and online). Summary data from all seven studies are available in PowerPerformance ma.

Usage

```
data_powerperformance_ma
```

Format

`data_powerperformance_ma`:

A data frame with 8 rows and 12 columns:

StudyName factor

Country factor

Population factor

Difficulty factor

Task factor

M Control numeric

s Control numeric

M Power numeric

s Power numeric

Cohen d unb numeric

n Control integer

n Power integer

Source

doi:10.1371/journal.pone.0140806

`data_rattanmotivation` *RattanMotivation - Ch14 - from Rattan et al. (2012)*

Description

How do you think you would react to feedback that gave encouragement and reassurance, or, instead, encouragement and challenge? Carol Dweck and her colleagues have investigated many such questions about how people respond to different types of feedback. My next example comes from Dweck's research group and illustrates data analysis that starts with the full data, rather than only summary statistics. Rattan et al. (2012) asked their college student participants to imagine they were undertaking a mathematics course and had just received a low score (65%) on the first test of the year. Participants were assigned randomly into three groups, which received different feedback along with the low score. The Comfort group received positive encouragement and also reassurance, the Challenge group received positive encouragement and also challenge, and the Control group received just the positive encouragement. Participants then responded to a range of questions about how they felt about the course and their professor. I'll discuss data for their ratings of their own motivation toward mathematics, made after they had received the feedback.

Usage

```
data_rattanmotivation
```

Format

`data_rattanmotivation`**:**

A data frame with 54 rows and 2 columns:

Group factor

Motivation numeric

Source

doi:10.1016/j.jesp.2011.12.012

`data_religionsharing` *ReligionSharing - Ch14 -* **RETRACTED DATA** *used in End-of-Chapter Exercise 14.3*

Description

To what extent is a religious upbringing related to prosocial behavior in childhood? To investigate, a large international sample of children was asked to play a game in which they were given 10 stickers but then asked if they would give some of these stickers away to another child who had not been able to be tested that day. The number of stickers donated was considered a measure of altruistic sharing. In addition, the parents of each child reported the family's religion. Summary data provided. **THIS STUDY HAS BEEN RETRACTED DUE TO AN ERRONEOUS ANALYSIS - THE DATASET WILL BE REMOVED FROM FUTURE VERSIONS OF ESCI AND THE BOOK**

Usage

```
data_religionsharing
```

Format

`data_religionsharing`**:**

A data frame with 3 rows and 4 columns:

Group factor

Mean numeric

SD numeric

N integer

Source

doi:10.1016/j.cub.2015.09.056

`data_religious_belief` *Religious belief - Ch03 - for End-of-Chapter Exercise 3.5*

Description

Let's look at some data about religious beliefs. The Religious belief file has data from a large online survey in which participants were asked to report, on a scale from 0 to 100, their belief in the existence of God. Age was also reported.

Usage

```
data_religious_belief
```

Format

`data_religious_belief`**:**

A data frame with 213 rows and 3 columns:

Response_ID character

Belief_in_God factor

Age integer

`data_selfexplain` *SelfExplain - Ch15 - from McEldoon et al. (2013)*

Description

Self-explaining is a learning strategy where students write or say their own explanations of the material they are studying. Self-explaining has generally been found to be more effective than standard studying, but it may also take more time. This raises the question of whether it's the study strategy or the extra time that benefits learning. To explore this issue, grade school children took a pretest of mathematics conceptual knowledge, studied mathematics problems, and then took a similar posttest (McEldoon et al., 2013). Participants were randomly assigned to one of two study conditions: normal study + more practice (More Practice group), or self-explaining (Self-Explain group). The first condition was intended to make time spent learning to be similar for the two groups. You can find part of the data from this study in SelfExplain, the scores being percent correct.

Usage

```
data_selfexplain
```

Format

`data_selfexplain`**:**

A data frame with 52 rows and 4 columns:

Student ID factor

Condition factor

Pretest numeric

Posttest numeric

Source

doi:10.1111/j.20448279.2012.02083.x

`data_simmonscredibility`

SimmonsCredibility - Ch14 - from Simmons and Nelson (2020)

Description

You're excited! Your company has developed a wonderful new weight-loss program, and now it's your job to develop the ad campaign. Should you choose a BeforeAfter pair of pictures, as in Figure 14.1, top panel? Or might a Progressive sequence of pictures of the same person, as in the bottom panel, be more effective? Pause, think, and discuss. Which would you choose, and why? You might think that BeforeAfter is simpler and more dramatic. On the other hand, Progressive highlights the steady improvement that you claim the program will deliver. You're probably not surprised to learn that BeforeAfter is used often and has long been a favorite of the advertising industry, whereas Progressive is used only rarely. Luca Cian and colleagues (Cian et al., 2020) were curious to know the extent to which BeforeAfter is actually more effective, appealing, and credible than Progressive, or, indeed, whether Progressive might score more highly. They reported seven studies of various aspects of that question. I'll focus on their Study 2, in which they used three independent groups to compare all three conditions illustrated in Figure 14.1. The BeforeAfterInfo condition, in the middle panel, comprises three BeforeAfter pairs, thus providing extra information about the before and after endpoints. The researchers included this condition in case any advantage of Progressive might stem simply from having more images, rather than because it illustrates a clear progressive sequence. They randomly assigned 213 participants from MTurk to one of the three groups. Participants were asked to 'imagine that you have decided to lose some weight', then saw one of the three ads for a weight loss program called MRMDiets. They then answered the question 'How would you evaluate MRMDiets?' by choosing a 1-7 response on several scales, including Unlikeable-Likable, Ineffective-Effective, and Not credible-Credible. The researchers averaged six such scores to give an overall Credibility score, on the 1-7 scale, with 7 being the most credible. Simmons and Nelson (2020) were sufficiently intrigued to carry out two substantial very close replications. With the cooperation of the original researchers, they used the same materials and procedure. They used much larger groups and preregistered their research plan, including data analysis plan. I'll focus on their first replication, in which 761 participants from MTurk were randomized to the three groups.

Usage

```
data_simmonscredibility
```

Format

`data_simmonscredibility`**:**

A data frame with 3 rows and 4 columns:

Groups factor

Mean numeric

SD numeric

n integer

Source

`http://datacolada.org/94`

`data_sleep_beauty` *Sleep Beauty - Ch11 - for End-of-Chapter Exercise 11.6*

Description

Is there really such a thing as beauty sleep? To investigate, researchers decided to examine the extent to which sleep relates to attractiveness. Each of 70 college students self-reported the amount of sleep they had the night before. In addition, a photograph was taken of each participant and rated for attractiveness on a scale from 1 to 10 by two judges of the opposite gender. The average rating score was used. You can download this data set (Sleep Beauty) from the book website.

Usage

```
data_sleep_beauty
```

Format

`data_sleep_beauty`**:**

A data frame with 70 rows and 2 columns:

Sleep (hours) numeric

Rated_Attractiveness numeric

`data_smithrecall` *SmithRecall - Ch15 - from Smith et al. (2016)*

Description

SmithRecall - Ch15 - from Smith et al. (2016)

Usage

```
data_smithrecall
```

Format

`data_smithrecall`**:**

A data frame with 120 rows and 6 columns:

ID integer

Study Technique factor

Stress Status factor

%Recalled numeric

Items Recalled numeric

Group factor

`data_stickgold` *Stickgold - Ch06 - from Stickgold et al. (2000)*

Description

Stickgold et al. (2000) found that, remarkably, performance on a visual discrimination task actually improved over the 48-96 hours after initial training, even without practice during that time. However, what if participants were sleep deprived during that period? They trained 11 participants in that new skill, then all were sleep deprived. The data were (-14.7, -10.7, -10.7, 2.2, 2.4, 4.5, 7.2, 9.6, 10, 21.3, 21.8)-or download the Stickgold data set from the book website. The data are the changes in performance scores from immediately after training to after the night without sleep: 0 represents no change, positive scores represent improvement, and negative scores represent decline. Data set courtesy of DataCrunch (tiny.cc/Stickgold)

Usage

```
data_stickgold
```

Format

`data_stickgold`**:**

A data frame with 11 rows and 3 columns:

Sleep deprived numeric

B factor

C factor

Source

https://www.statcrunch.com/app/index.html?dataid=1053539

`data_studystrategies` *StudyStrategies - Ch14 - from O'Reilly et al. (1998)*

Description

To what extent does study strategy influence learning? To investigate, psychology students were randomly assigned to three groups and asked to learn biology facts using one of three different strategies: a) Self-Explain (explaining for each fact what new knowledge is gained and how it relates to what is already known), b) Elab Interrogation (elaborative interrogation: stating for each fact why it makes sense), or c) Repetition Control (stating each fact over and over). After studying, students took a 25-point fill-the-blank test (O'Reilly et al., 1998)

Usage

```
data_studystrategies
```

Format

`data_studystrategies`**:**

A data frame with 3 rows and 10 columns:

Group factor

TestMean numeric

TestSD numeric

TestN integer

PrevKnowMean numeric

PrevKnowSD numeric

PrevKnowN integer

EaseUseMean numeric

EaseUseSD numeric

EaseUseN integer

Source

doi:10.1006/ceps.1997.0977

`data_thomason_1` *Thomason 1 - Ch11 - from Thomason 1*

Description

Summary data from an unpublished study by Neil Thomason and colleagues, who were interested in ways to enhance students' critical thinking. They were investigating argument mapping, which is a promising way to use diagrams to represent the structure of arguments. Students in their study completed an established test of critical thinking (the Pretest), then a critical thinking course based on argument mapping, then a second version of the test (the Posttest).

Usage

```
data_thomason_1
```

Format

`data_thomason_1`**:**

A data frame with 12 rows and 3 columns:

Participant ID factor

Pretest numeric

Posttest numeric

`data_videogameaggression` *VideogameAggression - Ch15 - from Hilgard (2015)*

Description

Video games can be violent and they can also be challenging. To what extent might these factors cause aggressive behavior? To explore, Hilgard (2015) asked male participants to play one of four versions of a video game for 15 minutes. The game was customized so that it could vary in violence (shooting zombies or helping aliens) and difficulty (targets controlled by tough AI or dumb AI). After the game, players were provoked by being given an insulting evaluation by a confederate. Participants then got to decide how long the confederate should hold their hand in painfully cold ice water (between 0 and 80 seconds), and this was taken as a measure of aggressive behavior. You can find the materials and analysis plan for this study on the Open Science Framework: osf. io/cwenz. This is a simplified version of the full data set.

Usage

```
data_videogameaggression
```

Format

`data_videogameaggression`**:**

A data frame with 223 rows and 3 columns:

Violence factor

Difficulty factor

Agression numeric

Source

doi:10.1177/0956797619829688

`esci_plot_difference_axis_x`

Add a difference axis to the x axis of an esci forest plot

Description

`esci_plot_difference_axis_x` can be used to redraw the difference axis from a forest plot created with plot_meta. You must pass the plot returned from plot_meta and the effect size table containing the estimated difference.

Usage

```
esci_plot_difference_axis_x(
  myplot,
  difference_table,
  dlim = c(NA, NA),
  d_n.breaks = NULL,
  d_lab = NULL
)
```

Arguments

`myplot`	required ggplot2 plot returned from a plot_meta function
`difference_table`	required data frame from an esci-estimate that has a difference-based effect size
`dlim`	Optional 2-item vector to provide the lower and upper boundaries of the difference axis. Defaults to c(NA, NA) which is to auto-set both boundaries.
`d_n.breaks`	Optional numeric > 2 to suggest number of breaks for the difference axis; defaults to NULL in which case number of breaks is handled automatically by ggplot
`d_lab`	Optional character to serve as the label for the difference axis; defaults to NULL

`estimate_magnitude` *Estimates for a continuous variable with no grouping (single-group design)*

Description

`estimate_magnitude` is suitable for a single group design with a continuous outcome variable. It estimates the population mean and population median (raw data only) with confidence intervals. You can pass raw data or summary data.

Usage

```
estimate_magnitude(
  data = NULL,
  outcome_variable = NULL,
  mean = NULL,
  sd = NULL,
  n = NULL,
  outcome_variable_name = "My outcome variable",
  conf_level = 0.95,
  save_raw_data = TRUE
)
```

Arguments

`data`	For raw data - A data frame or tibble
`outcome_variable`	For raw data - The column name of the outcome variable, or a vector of numeric data
`mean`	For summary data - A numeric representing the mean of the outcome variable
`sd`	For summary data - A numeric > 0, standard deviation of the outcome variable
`n`	For summary data - An integer > 0, sample size of the outcome variable
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`save_raw_data`	For raw data; defaults to TRUE; set to FALSE to save memory by not returning raw data in estimate object

Details

Reach for this function in place of a one-sample *t*-test or *z*-test.

Once you generate an estimate with this function, you can visualize it with `plot_magnitude()`.

If you want to compare your sample to a known value or reference, then use `estimate_mdiff_one()`.

The estimated mean is from `statpsych::ci.mean1()` (renamed ci.mean as of statpsych 1.6).

The estimated median is from `statpsych::ci.median1()` (renamed ci.median as of statpsych 1.6)

Value

Returns an object of class esci_estimate

- **overview**
  - *outcome_variable_name* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -
  - *q1* -
  - *q3* -
  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -
- **es_mean**
  - *outcome_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
- **es_median**
  - *outcome_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -

  - *ta_LL* -
  - *ta_UL* -
- **raw_data**
  - *grouping_variable* -
  - *outcome_variable* -

Examples

```
# From raw data
data("data_penlaptop1")

estimate_from_raw <- esci::estimate_magnitude(
  data = data_penlaptop1[data_penlaptop1$condition == "Pen", ],
    outcome_variable = transcription
)

# To visualize the estimate
myplot_from_raw <- esci::plot_magnitude(
  estimate_from_raw,
  effect_size = "median"
)


# From summary data
mymean <- 24.5
mysd <- 3.65
myn <- 40

estimate_from_summary <- esci::estimate_magnitude(
  mean = mymean,
  sd = mysd,
  n = myn
)

# To visualize the estimate
myplot_from_summary <- esci::plot_magnitude(
  estimate_from_summary,
  effect_size = "mean"
)
```

`estimate_mdiff_2x2_between` *Estimates for a 2x2 between-subjects design with a continuous outcome variable*

Description

Returns object `estimate_mdiff_2x2_between` is suitable for a 2x2 between-subjects design with a continuous outcome variable. It estimates each main effect, the simple effects for the first factor, and the interaction. It can express these estimates as mean differences, standardized mean differences (Cohen's d), and as median differences (raw data only). You can pass raw data or or summary data (summary data does not return medians).

Usage

```
estimate_mdiff_2x2_between(
  data = NULL,
  outcome_variable = NULL,
  grouping_variable_A = NULL,
  grouping_variable_B = NULL,
  means = NULL,
  sds = NULL,
  ns = NULL,
  grouping_variable_A_levels = NULL,
  grouping_variable_B_levels = NULL,
  outcome_variable_name = "My outcome variable",
  grouping_variable_A_name = "A",
  grouping_variable_B_name = "A",
  conf_level = 0.95,
  assume_equal_variance = FALSE,
  save_raw_data = TRUE
)
```

Arguments

`data`	For raw data - a data frame or tibble
`outcome_variable`	For raw data - The column name of the outcome variable, or a vector of numeric data
`grouping_variable_A`	For raw data - The column name of the grouping variable, or a vector of group names, only 2 levels allowed
`grouping_variable_B`	For raw data - The column name of the grouping variable, or a vector of group names, only 2 levels allowed
`means`	For summary data - A vector of 4 means: A1B1, A1B2, A2B1, A2B2
`sds`	For summary data - A vector of 4 standard deviations, same order
`ns`	For summary data - A vector of 4 sample sizes
`grouping_variable_A_levels`	For summary data - An optional vector of 2 group labels
`grouping_variable_B_levels`	For summary data - An optional vector of 2 group labels

`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`grouping_variable_A_name`	Optional friendly name for the grouping variable. Defaults to 'A' or the grouping variable column name if a data.frame is passed.
`grouping_variable_B_name`	Optional friendly name for the grouping variable. Defaults to 'A' or the grouping variable column name if a data.frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`assume_equal_variance`	Defaults to FALSE
`save_raw_data`	For raw data; defaults to TRUE; set to FALSE to save memory by not returning raw data in estimate object

Details

Reach for this function in place of a 2x2 between-subjects ANOVA.

Once you generate an estimate with this function, you can visualize it with `plot_mdiff()` and you can visualize the interaction specifically with `plot_interaction()`. You can test hypotheses with `test_mdiff()`.

The estimated mean differences are from `statpsych::ci.2x2.mean.bs()`.

The estimated SMDs are from `statpsych::ci.2x2.stdmean.bs()`.

The estimated median differences are from `statpsych::ci.2x2.median.bs()`

Value

Returns object of class esci_estimate

- **es_mean_difference**
  - *type* -
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
  - *effect_type* -
  - *effects_complex* -
- **es_median_difference**

  - *type* -
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *ta_LL* -
  - *ta_UL* -
  - *effect_type* -
  - *effects_complex* -
- **es_smd**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *numerator* -
  - *denominator* -
  - *SE* -
  - *df* -
  - *d_biased* -
  - *effect_type* -
  - *effects_complex* -
- **overview**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -
  - *q1* -
  - *q3* -

  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -
- **raw_data**
  - *grouping_variable* -
  - *outcome_variable* -
  - *grouping_variable_A* -
  - *grouping_variable_B* -

Examples

```
data("data_videogameaggression")

estimates_from_raw <- esci::estimate_mdiff_2x2_between(
  esci::data_videogameaggression,
  Agression,
  Violence,
  Difficulty
)

# To visualize the estimated mean difference for the interaction
myplot_from_raw <- esci::plot_mdiff(
  estimates_from_raw$interaction,
  effect_size = "median"
)

# To conduct a hypothesis test on the mean difference
res_htest_from_raw <- esci::test_mdiff(
  estimates_from_raw$interaction,
  effect_size = "median"
)


# From summary data
means <- c(1.5, 1.14, 1.38, 2.22)
sds <- c(1.38, .96,1.5, 1.68)
ns <- c(26, 26, 25, 26)
grouping_variable_A_levels <- c("Evening", "Morning")
grouping_variable_B_levels <- c("Sleep", "No Sleep")

estimates_from_summary <- esci::estimate_mdiff_2x2_between(
  means = means,
  sds = sds,
  ns = ns,
  grouping_variable_A_levels = grouping_variable_A_levels,
  grouping_variable_B_levels = grouping_variable_B_levels,
  grouping_variable_A_name = "Testing Time",
  grouping_variable_B_name = "Rest",
```

```
  outcome_variable_name = "False Memory Score",
  assume_equal_variance = TRUE
)

# To visualize the estimated mean difference for the interaction
plot_mdiff_interaction <- esci::plot_mdiff(
  estimates_from_summary$interaction,
  effect_size = "mean"
)

# To visualize the interaction as a line plot
plot_interaction_line <- esci::plot_interaction(estimates_from_summary)

# Same but with fan effect representing each simple-effect CI
plot_interaction_line_CI <- esci::plot_interaction(
  estimates_from_summary,
  show_CI = TRUE
)

# To conduct a hypothesis test on the mean difference
res_htest_from_raw <- esci::test_mdiff(
  estimates_from_summary$interaction,
  effect_size = "mean"
)
```

`estimate_mdiff_2x2_mixed` *Estimates for a 2x2 mixed factorial design with a continuous outcome variable*

Description

Returns object `estimate_mdiff_2x2_mixed` is suitable for a 2x2 mixed-factorial design with a continuous outcome variable. It estimates each main effect, the simple effects for the repeated-measures factor, and the interaction. It can express these estimates as mean differences, median difference, or standardized mean differences. This function accepts raw data only.

Usage

```
estimate_mdiff_2x2_mixed(
  data,
  outcome_variable_level1,
  outcome_variable_level2,
  grouping_variable,
  outcome_variable_name = "My outcome variable",
  repeated_measures_name = "Time",
  conf_level = 0.95,
```

```
  save_raw_data = TRUE
)
```

Arguments

`data`	For raw data - a dataframe or tibble
`outcome_variable_level1`	The column name of the outcome variable for level 1 of the repeated-measures factor
`outcome_variable_level2`	The column name of the outcome variable for level 2 of the repeated-measures factor
`grouping_variable`	The column name of the grouping variable; only 2 levels allowed; must be a factor
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`repeated_measures_name`	Optional friendly name for the repeated measures factor. Defaults to 'Time'
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`save_raw_data`	For raw data; defaults to TRUE; set to FALSE to save memory by not returning raw data in estimate object

Details

Reach for this function in place of a 2x2 mixed-factorial ANOVA.

Once you generate an estimate with this function, you can visualize it with `plot_mdiff()` and you can visualize the interaction specifically with `plot_interaction()`. You can test hypotheses with `test_mdiff()`.

The estimated mean differences are from `statpsych::ci.2x2.mean.mixed()`.

Value

Returns object of class esci_estimate

- **es_mean_difference**
  - *type* -
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -

  - *df* -
  - *ta_LL* -
  - *ta_UL* -
  - *effect_type* -
  - *effects_complex* -
  - *t* -
  - *p* -
- **es_smd**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *numerator* -
  - *denominator* -
  - *SE* -
  - *df* -
  - *d_biased* -
  - *effect_type* -
  - *effects_complex* -
- **overview**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -
  - *q1* -
  - *q3* -
  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -

- **raw_data**
  - *grouping_variable* -
  - *outcome_variable* -
  - *grouping_variable_A* -
  - *grouping_variable_B* -
  - *paired* -

Examples

```
# From raw data (summary data mode not available for this function)
example_data <- data.frame(
  pretest = c(
    19, 18, 19, 20, 17, 16, 16, 10, 12,  9, 13, 15
  ),
  posttest = c(
    18, 19, 20, 17, 20, 16, 19, 16, 16, 14, 16, 18
  ),
  condition = as.factor(
    c(
      rep("Control", times = 6),
      rep("Treated", times = 6)
    )
  )
)

estimates <- esci::estimate_mdiff_2x2_mixed(
  data = example_data,
  outcome_variable_level1 = pretest,
  outcome_variable_level2 = posttest,
  grouping_variable = condition,
  repeated_measures_name = "Time"
)
# To visualize the estimated mean difference for the interaction
myplot <- esci::plot_mdiff(estimates$interaction, effect_size = "mean")

# Line-plot of the interaction with fan effect representing each simple-effect CI
plot_interaction_line_CI <- esci::plot_interaction(
  estimates,
  show_CI = TRUE
)

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_mdiff(
  estimates$interaction,
  effect_size = "mean"
 )
```

`estimate_mdiff_ind_contrast` *Estimates for a multi-group design with a continuous outcome variable*

Description

Returns object `estimate_mdiff_ind_contrast` is suitable for a multi-group design (between subjects) with a continuous outcome variable. It accepts a user-defined set of contrast weights that allows estimation of any 1-df contrast. It can express estimates as mean differences, standardized mean differences (Cohen's d) or median differences (raw data only). You can pass raw data or summary data.

Usage

```
estimate_mdiff_ind_contrast(
  data = NULL,
  outcome_variable = NULL,
  grouping_variable = NULL,
  means = NULL,
  sds = NULL,
  ns = NULL,
  contrast = NULL,
  grouping_variable_levels = NULL,
  outcome_variable_name = "My outcome variable",
  grouping_variable_name = "My grouping variable",
  conf_level = 0.95,
  assume_equal_variance = FALSE,
  save_raw_data = TRUE
)
```

Arguments

`data`	For raw data - a data frame or tibble
`outcome_variable`	For raw data - The column name of the outcome variable, or a vector of numeric data
`grouping_variable`	For raw data - The column name of the grouping variable, or a vector of group names
`means`	For summary data - A vector of 2 or more means
`sds`	For summary data - A vector of standard deviations, same length as means
`ns`	For summary data - A vector of sample sizes, same length as means
`contrast`	A vector of group weights, same length as number of groups.
`grouping_variable_levels`	For summary data - An optional vector of group labels, same length as means

`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`grouping_variable_name`	Optional friendly name for the grouping variable. Defaults to 'My grouping variable' or the grouping variable column name if a data.frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`assume_equal_variance`	Defaults to FALSE
`save_raw_data`	For raw data; defaults to TRUE; set to FALSE to save memory by not returning raw data in estimate object

Details

Reach for this function in place of a one-way ANOVA.

Once you generate an estimate with this function, you can visualize it with `plot_mdiff()` and you can test hypotheses with `test_mdiff()`.

The estimated mean differences are from `statpsych::ci.lc.mean.bs()`.

The estimated SMDs are from `CI_smd_ind_contrast()` which relies on `statpsych::ci.lc.stdmean.bs()` unless there are only 2 groups.

The estimated median differences are from `statpsych::ci.lc.median.bs()`

Value

Returns object of class esci_estimate

- **es_mean_difference**
  - *type* -
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
- **es_median_difference**
  - *type* -
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -

  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *ta_LL* -
  - *ta_UL* -
- **es_smd**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *numerator* -
  - *denominator* -
  - *SE* -
  - *df* -
  - *d_biased* -
- **overview**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -
  - *q1* -
  - *q3* -
  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -
- **raw_data**
  - *grouping_variable* -
  - *outcome_variable* -

Examples

```
# From raw data
data("data_rattanmotivation")

estimate_from_raw <- esci::estimate_mdiff_ind_contrast(
  esci::data_rattanmotivation,
  Motivation,
  Group,
  contrast = c("Challenge" = 1, "Control" = -1/2, "Comfort" = -1/2)
)

# To visualize the estimate
myplot_from_raw <- esci::plot_mdiff(
  estimate_from_raw,
  effect_size = "median"
)

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_mdiff(
  estimate_from_raw,
  effect_size = "median"
)


# From summary data
data("data_halagappa")

estimate_from_summary <- estimate_mdiff_ind_contrast(
  means = data_halagappa$Mean,
  sds = data_halagappa$SD,
  ns = data_halagappa$n,
  grouping_variable_levels = as.character(data_halagappa$Groups),
  assume_equal_variance = TRUE,
  contrast = c(
    "NFree10" = 1/3,
    "AFree10" = 1/3,
    "ADiet10" = -1/3,
    "NFree17" = -1/3,
    "AFree17" = 1/3,
    "ADiet17" = -1/3
  ),
  grouping_variable_name = "Diet",
  outcome_variable_name = "% time near target"
)

# To visualize the estimate
myplot <- esci::plot_mdiff(estimate_from_summary, effect_size = "mean")

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_mdiff(
  estimate_from_summary,
  effect_size = "mean"
```

```
)
```

`estimate_mdiff_one` *Estimates for a single-group design with a continuous outcome variable compared to a reference or population value*

Description

Returns object `estimate_mdiff_one` is suitable for a single-group design with a continuous outcome variable that is compared to a reference or population value. It can express estimates as mean differences, standardized mean differences (Cohen's d) or median differences (raw data only). You can pass raw data or summary data.

Usage

```
estimate_mdiff_one(
  data = NULL,
  outcome_variable = NULL,
  comparison_mean = NULL,
  comparison_sd = NULL,
  comparison_n = NULL,
  reference_mean = 0,
  outcome_variable_name = "My outcome variable",
  conf_level = 0.95,
  save_raw_data = TRUE
)
```

Arguments

`data`	For raw data - a data frame or tibble
`outcome_variable`	For raw data - The column name of the outcome variable, or a vector of numeric data
`comparison_mean`	For summary data, a numeric
`comparison_sd`	For summary data, numeric > 0
`comparison_n`	For summary data, a numeric integer > 0
`reference_mean`	Reference value, defaults to 0
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`save_raw_data`	For raw data; defaults to TRUE; set to FALSE to save memory by not returning raw data in estimate object

Details

Reach for this function in place of a *z*-test or one-sample *t*-test.

Once you generate an estimate with this function, you can visualize it with `plot_mdiff()` and you can test hypotheses with `test_mdiff()`.

The estimated mean differences are from `statpsych::ci.mean1()` (renamed ci.mean as of statpsych 1.6).

The estimated SMDs are from `CI_smd_one()`.

The estimated median differences are from `statpsych::ci.median1()` (renamed ci.median as of statpsych 1.6)

Value

Returns object of class esci_estimate

- **overview**
  - *outcome_variable_name* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -
  - *q1* -
  - *q3* -
  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -
- **es_mean**
  - *outcome_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -

- **es_median**
  - *outcome_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
- **raw_data**
  - *grouping_variable* -
  - *outcome_variable* -
- **es_mean_difference**
  - *outcome_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
  - *type* -
- **es_median_difference**
  - *outcome_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
  - *type* -
- **es_smd**
  - *outcome_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -

  - *numerator* -
  - *denominator* -
  - *SE* -
  - *df* -
  - *d_biased* -

Examples

```
# From raw data
data("data_penlaptop1")
estimate_from_raw <- esci::estimate_mdiff_one(
  data = data_penlaptop1[data_penlaptop1$condition == "Pen", ],
  outcome_variable = transcription,
  reference_mean = 10
)

# To visualize the mean difference estimate
myplot_from_raw <- esci::plot_mdiff(estimate_from_raw, effect_size = "mean")

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_mdiff(
  estimate_from_raw,
  effect_size = "mean",
  rope = c(-2, 2)
)


# From summary data
mymean <- 12.09
mysd <- 5.52
myn <- 103

estimate_from_summary <- esci::estimate_mdiff_one(
  comparison_mean = mymean,
  comparison_sd = mysd,
  comparison_n = myn,
  reference_mean = 12
)

# To visualize the estimate
myplot_from_sumary <- esci::plot_mdiff(
  estimate_from_summary,
  effect_size = "mean"
)

# To conduct a hypothesis test
res_htest_from_summary <- esci::test_mdiff(
  estimate_from_summary,
  effect_size = "mean",
  rope = c(-2, 2)
)
```

`estimate_mdiff_paired` *Estimates for a repeated-measures study with two measures of a continuous variable*

Description

Returns object `estimate_mdiff_paired` is suitable for a simple paired design with a continuous outcome variable. It provides estimates and CIs for the population mean difference between the repeated measures, the standardized mean difference (SMD; Cohen's d) between the repeated measures, and the median difference between the repeated measures (raw data only). You can pass raw data or summary data.

Usage

```
estimate_mdiff_paired(
  data = NULL,
  comparison_measure = NULL,
  reference_measure = NULL,
  comparison_mean = NULL,
  comparison_sd = NULL,
  reference_mean = NULL,
  reference_sd = NULL,
  n = NULL,
  correlation = NULL,
  comparison_measure_name = "Comparison measure",
  reference_measure_name = "Reference measure",
  conf_level = 0.95,
  save_raw_data = TRUE
)
```

Arguments

`data`	For raw data - a data frame or tibble
`comparison_measure`	For raw data - The column name of comparison measure of the outcome variable, or a vector of numeric data
`reference_measure`	For raw data - The column name of the reference measure of the outcome variable, or a vector of numeric data
`comparison_mean`	For summary data, a numeric
`comparison_sd`	For summary data, numeric > 0
`reference_mean`	For summary data, a numeric
`reference_sd`	For summary data, numeric > 0
`n`	For summary data, a numeric integer > 0

`correlation`	For summary data, correlation between measures, a numeric that is > -1 and < 1
`comparison_measure_name`	For summary data - An optional character label for the comparison measure. Defaults to 'Comparison measure'
`reference_measure_name`	For summary data - An optional character label for the reference measure. Defaults to 'Reference measure'
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`save_raw_data`	For raw data; defaults to TRUE; set to FALSE to save memory by not returning raw data in estimate object

Details

Reach for this function in place of a paired-samples *t*-test.

Once you generate an estimate with this function, you can visualize it with `plot_mdiff()` and you can test hypotheses with `test_mdiff()`.

The estimated mean differences are from `statpsych::ci.mean.ps()`.

The estimated SMDs are from `CI_smd_ind_contrast()`.

The estimated median differences are from `statpsych::ci.median.ps()`.

Value

Returns object of class esci_estimate

- **overview**
  - *outcome_variable_name* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -
  - *q1* -
  - *q3* -
  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -
- **es_mean_difference**

  - *type* -
  - *comparison_measure_name* -
  - *reference_measure_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
- **es_smd**
  - *comparison_measure_name* -
  - *reference_measure_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *numerator* -
  - *denominator* -
  - *SE* -
  - *d_biased* -
  - *df* -
- **es_r**
  - *x_variable_name* -
  - *y_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *n* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
- **es_median_difference**
  - *type* -
  - *comparison_measure_name* -
  - *reference_measure_name* -
  - *effect* -
  - *effect_size* -

  - *LL* -
  - *UL* -
  - *SE* -
  - *ta_LL* -
  - *ta_UL* -
- **es_mean_ratio**
  - *comparison_measure_name* -
  - *reference_measure_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *comparison_mean* -
  - *reference_mean* -
- **es_median_ratio**
  - *comparison_measure_name* -
  - *reference_measure_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *comparison_median* -
  - *reference_median* -
- **raw_data**
  - *comparison_measure* -
  - *reference_measure* -

Examples

```
# From raw data
data("data_thomason_1")

estimate_from_raw <- esci::estimate_mdiff_paired(
  data = esci::data_thomason_1,
  comparison_measure = Posttest,
  reference_measure = Pretest
)

# To visualize the estimated median difference (raw data only)
myplot_from_raw <- esci::plot_mdiff(estimate_from_raw, effect_size = "median")

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_mdiff(
  estimate_from_raw,
  effect_size = "median",
```

```
  rope = c(-2, 2)
)


sd1 <- 4.28
sd2 <- 3.4
sdiff <- 2.13

cor <- (sd1^2 + sd2^2 - sdiff^2) / (2*sd1*sd2)

estimate_from_summary <- esci::estimate_mdiff_paired(
  comparison_mean = 14.25,
  comparison_sd = 4.28,
  reference_mean = 12.88,
  reference_sd = 3.4,
  n = 16,
  correlation = 0.87072223749,
  comparison_measure_name = "After",
  reference_measure_name = "Before"
)

# To visualize the estimated mean difference
myplot_from_summary <- esci::plot_mdiff(
  estimate_from_summary,
  effect_size = "mean"
)

# To conduct a hypothesis test
res_htest_from_summary <- esci::test_mdiff(
  estimate_from_summary,
  effect_size = "mean",
  rope = c(-2, 2)
)
```

`estimate_mdiff_two` *Estimates for a two-group study with a continuous outcome variable*

Description

Returns object `estimate_mdiff_two` is suitable for a simple two-group design with a continuous outcome variable. It provides estimates and CIs for the population mean difference between the repeated measures, the standardized mean difference (SMD; Cohen's d) between the repeated measures, and the median difference between the repeated measures (raw data only). You can pass raw data or summary data.

Usage

```
estimate_mdiff_two(
```

```
  data = NULL,
  outcome_variable = NULL,
  grouping_variable = NULL,
  comparison_mean = NULL,
  comparison_sd = NULL,
  comparison_n = NULL,
  reference_mean = NULL,
  reference_sd = NULL,
  reference_n = NULL,
  grouping_variable_levels = NULL,
  outcome_variable_name = "My outcome variable",
  grouping_variable_name = "My grouping variable",
  conf_level = 0.95,
  assume_equal_variance = FALSE,
  save_raw_data = TRUE,
  switch_comparison_order = FALSE
)
```

Arguments

`data`	For raw data - a data.frame or tibble
`outcome_variable`	For raw data - The column name of the outcome variable, or a vector of numeric data
`grouping_variable`	For raw data - The column name of the grouping variable, or a vector of group names
`comparison_mean`	For summary data, a numeric
`comparison_sd`	For summary data, numeric > 0
`comparison_n`	For summary data, a numeric integer > 0
`reference_mean`	For summary data, a numeric
`reference_sd`	For summary data, numeric > 0
`reference_n`	For summary data, a numeric integer > 0
`grouping_variable_levels`	For summary data - An optional vector of 2 group labels
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`grouping_variable_name`	Optional friendly name for the grouping variable. Defaults to 'My grouping variable' or the grouping variable column name if a data.frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`assume_equal_variance`	Defaults to FALSE

`save_raw_data`	For raw data; defaults to TRUE; set to FALSE to save memory by not returning raw data in estimate object
`switch_comparison_order`	Defaults to FALSE

Details

Reach for this function in place of an independent-samples *t*-test.

Once you generate an estimate with this function, you can visualize it with `plot_mdiff()` and you can test hypotheses with `test_mdiff()`.

The estimated mean differences are from `statpsych::ci.mean2()`.

The estimated SMDs are from `CI_smd_ind_contrast()`.

The estimated median differences are from `statpsych::ci.median2()`.

Value

Returns object of class esci_estimate

- **es_mean_difference**
  - *type* -
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
- **es_median_difference**
  - *type* -
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *ta_LL* -
  - *ta_UL* -
- **es_smd**
  - *outcome_variable_name* -

  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *numerator* -
  - *denominator* -
  - *SE* -
  - *df* -
  - *d_biased* -
- **es_mean_ratio**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *comparison_mean* -
  - *reference_mean* -
- **es_median_ratio**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *comparison_median* -
  - *reference_median* -
- **overview**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -

  - *q1* -
  - *q3* -
  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -
- **raw_data**
  - *grouping_variable* -
  - *outcome_variable* -

Examples

```
# From raw data
data("data_penlaptop1")

estimate_from_raw <- esci::estimate_mdiff_two(
  data = data_penlaptop1,
  outcome_variable = transcription,
  grouping_variable = condition,
  switch_comparison_order = TRUE,
  assume_equal_variance = TRUE
)

# To visualize the estimated median difference (raw data only)
myplot_from_raw <- esci::plot_mdiff(
  estimate_from_raw,
  effect_size = "median"
)

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_mdiff(
  estimate_from_raw,
  effect_size = "median",
  rope = c(-2, 2)
)


# From summary data
estimate_from_summary <- esci::estimate_mdiff_two(
  comparison_mean = 12.09,
  comparison_sd = 5.52,
  comparison_n = 103,
  reference_mean = 6.88,
  reference_sd = 4.22,
  reference_n = 48,
  grouping_variable_levels = c("Ref-Laptop", "Comp-Pen"),
  outcome_variable_name = "% Transcription",
  grouping_variable_name = "Note-taking type",
  assume_equal_variance = TRUE
```

```
)

# To visualize the estimated mean difference
myplot <- esci::plot_mdiff(
  estimate_from_summary,
  effect_size = "mean"
)

# To conduct a hypothesis test
res_htest_from_summary <- esci::test_mdiff(
  estimate_from_summary,
  effect_size = "mean",
  rope = c(-2, 2)
)
```

`estimate_pdiff_ind_contrast` *Estimates for a multi-group study with a categorical outcome variable*

Description

Returns object `estimate_pdiff_ind_contrast` is suitable for a multi-group design (between subjects) with a categorical outcome variable. It accepts a user-defined set of contrast weights that allows estimation of any 1-df contrast. It can express estimates as a difference in proportions and as an odds ratio (2-group designs only). You can pass raw data or summary data.

Usage

```
estimate_pdiff_ind_contrast(
  data = NULL,
  outcome_variable = NULL,
  grouping_variable = NULL,
  cases = NULL,
  ns = NULL,
  contrast = NULL,
  case_label = 1,
  grouping_variable_levels = NULL,
  outcome_variable_name = "My outcome variable",
  grouping_variable_name = "My grouping variable",
  conf_level = 0.95,
  count_NA = FALSE
)
```

Arguments

`data`	For raw data - a data frame or tibble

`outcome_variable`	For raw data - The column name of the outcome variable which is a factor, or a vector that is a factor
`grouping_variable`	For raw data - The column name of the grouping variable which is a factor, or a vector that is a factor
`cases`	For summary data - A numeric vector of 2 or more event counts, each an integer >= 0
`ns`	For summary data - A numeric vector of sample sizes, same length as counts, each an integer >= corresponding event count
`contrast`	A vector of group weights, same length as number of groups.
`case_label`	An optional numeric or character label For summary data, used as the label and defaults to 'Affected'. For raw data, used to specify the level used for the proportion.
`grouping_variable_levels`	For summary data - An optional vector of group labels, same length as cases
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`grouping_variable_name`	Optional friendly name for the grouping variable. Defaults to 'My grouping variable' or the grouping variable column name if a data.frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`count_NA`	Logical to count NAs (TRUE) in total N or not (FALSE)

Details

Once you generate an estimate with this function, you can visualize it with `plot_mdiff()` and you can test hypotheses with `test_mdiff()`.

The estimated proportion differences are from `statpsych::ci.lc.prop.bs()`.

The estimated odds ratios (if returned) are from `statpsych::ci.oddsratio()`.

Value

Returns object of class esci_estimate

- **es_proportion_difference**
  - *type* -
  - *outcome_variable_name* -
  - *case_label* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -

  - *UL* -
  - *SE* -
  - *effect_size_adjusted* -
  - *ta_LL* -
  - *ta_UL* -
- **es_odds_ratio**
  - *outcome_variable_name* -
  - *case_label* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *SE* -
  - *LL* -
  - *UL* -
  - *ta_LL* -
  - *ta_UL* -
- **overview**
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *outcome_variable_name* -
  - *outcome_variable_level* -
  - *cases* -
  - *n* -
  - *P* -
  - *P_LL* -
  - *P_UL* -
  - *P_SE* -
  - *P_adjusted* -
  - *ta_LL* -
  - *ta_UL* -
- **es_phi**
  - *grouping_variable_name* -
  - *outcome_variable_name* -
  - *effect* -
  - *effect_size* -
  - *SE* -
  - *LL* -
  - *UL* -

Examples

```
# From raw data
data("data_campus_involvement")

estimate_from_raw <- esci::estimate_pdiff_ind_contrast(
  esci::data_campus_involvement,
  CommuterStatus,
  Gender,
  contrast = c("Male" = -1, "Female" = 1)
)

# To visualize the estimate
myplot_from_raw <- esci::plot_pdiff(estimate_from_raw)

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_pdiff(estimate_from_raw)


# From summary data
estimate_from_summary <- esci::estimate_pdiff_ind_contrast(
  cases = c(78, 10),
  ns = c(252, 20),
  case_label = "egocentric",
  grouping_variable_levels = c("Original", "Replication"),
  contrast = c(-1, 1),
  conf_level = 0.95
)

# To visualize the estimate
myplot_from_summary <- esci::plot_pdiff(estimate_from_summary)

# To conduct a hypothesis test
res_htest_from_summary <- esci::test_pdiff(estimate_from_summary)
```

`estimate_pdiff_one` *Estimates for a single-group design with a categorical outcome variable compared to a reference or population value.*

Description

Returns object `estimate_pdiff_one` is suitable for a single-group design (between subjects) with a categorical outcome variable. It calculates the effect sizes with respect to a reference or population proportion (default value of 0). It returns the estimated difference between the in proportion from the reference/population value. You can pass raw data or summary data.

Usage

```
estimate_pdiff_one(
  data = NULL,
  outcome_variable = NULL,
  comparison_cases = NULL,
  comparison_n = NULL,
  reference_p = 0,
  case_label = 1,
  outcome_variable_name = "My outcome variable",
  conf_level = 0.95,
  count_NA = FALSE
)
```

Arguments

`data`	For raw data - a dataframe or tibble
`outcome_variable`	For raw data - The column name of the outcome variable, which must be a factor, or a vector that is a factor
`comparison_cases`	For summary data, a numeric integer > 0
`comparison_n`	For summary data, a numeric integer >= count
`reference_p`	Reference proportion, numeric >=0 and <=1
`case_label`	An optional numeric or character label for the count level.
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`count_NA`	Logical to count NAs (TRUE) in total N or not (FALSE)

Details

Once you generate an estimate with this function, you can visualize it with `plot_pdiff()` and you can test hypotheses with `test_pdiff()`.

The estimated proportion differences are from `statpsych::ci.prop1()` (renamed ci.prop as of statpsych 1.6).

Value

Returns an object of class esci_estimate

- **overview**
  - *outcome_variable_name* -
  - *outcome_variable_level* -
  - *cases* -

  - *n* -
  - *P* -
  - *P_LL* -
  - *P_UL* -
  - *P_SE* -
  - *P_adjusted* -
  - *ta_LL* -
  - *ta_UL* -
- **es_proportion_difference**
  - *outcome_variable_name* -
  - *case_label* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *effect_size_adjusted* -
  - *ta_LL* -
  - *ta_UL* -
  - *cases* -
  - *n* -
  - *type* -

Examples

```
# From raw data
data("data_campus_involvement")

estimate_from_raw <- esci::estimate_pdiff_one(
  esci::data_campus_involvement,
  CommuterStatus,
  reference_p = 0.50
)

# To visualize the estimate
myplot_from_raw <- esci::plot_pdiff(estimate_from_raw)

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_pdiff(estimate_from_raw)


# From summary data
estimate_from_summary  <- esci::estimate_pdiff_one(
  comparison_cases = 8,
  comparison_n = 22,
  reference_p = 0.5
)
```

```
# To visualize the estimate
myplot_from_summary <- esci::plot_pdiff(estimate_from_summary)

# To conduct a hypothesis test
res_htest_from_summary <- esci::test_pdiff(estimate_from_summary)
```

`estimate_pdiff_paired` *Estimates for a repeated-measures study with two measures of a categorical variable*

Description

Returns object `estimate_pdiff_paired` is suitable for a simple paired design with a categorical outcome variable. It provides estimates and CIs for the population proportion difference between the repeated measures. You can pass raw data or summary data.

Usage

```
estimate_pdiff_paired(
  data = NULL,
  comparison_measure = NULL,
  reference_measure = NULL,
  cases_consistent = NULL,
  cases_inconsistent = NULL,
  not_cases_consistent = NULL,
  not_cases_inconsistent = NULL,
  case_label = 1,
  not_case_label = NULL,
  comparison_measure_name = "Comparison measure",
  reference_measure_name = "Reference measure",
  conf_level = 0.95,
  count_NA = FALSE
)
```

Arguments

`data`	For raw data - a data.frame or tibble
`comparison_measure`	For raw data - The comparison measure, a factor. Can be the column name of a data frame of a vector.
`reference_measure`	For raw data - The reference measure, a factor. Can be the column name of a data frame of a vector.

`cases_consistent`	Count of *cases* in measure 1 that *are* also cases at measure 2; measure 1 = 0, measure 2 = 0; cell 0_0
`cases_inconsistent`	Count of *cases* in measure 1 that are *not* cases at measure 2; measure 1 = 0, measure 2 = 1; cell 0_1
`not_cases_consistent`	Count of *not cases* in measure 1 that *are* also not cases at measure 2; measure 1 = 1, measure 2 = 1, cell 1_1
`not_cases_inconsistent`	Count of *not cases* in measure 1 that are *not* cases at measure 2; measure 1 = 1, measure 2 = 0, cell 1_0
`case_label`	An optional numeric or character label for the case level.
`not_case_label`	An optional numeric or character label for the not case level.
`comparison_measure_name`	For summary data - An optional character label for the comparison measure. Defaults to 'Comparison measure'
`reference_measure_name`	For summary data - An optional character label for the reference measure. Defaults to 'Reference measure'
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`count_NA`	Logical to count NAs (TRUE) in total N or not (FALSE)

Details

Once you generate an estimate with this function, you can visualize it with `plot_pdiff()` and you can test hypotheses with `test_pdiff()`.

The estimated proportion differences are from `statpsych::ci.prop.ps()`.

Value

Returns object of class esci_estimate

Examples

```
# From summary data
# Example 1 from Bonett & Price, 2012
estimate_from_summary <- esci::estimate_pdiff_paired(
  cases_consistent = 60,
  cases_inconsistent = 50,
  not_cases_inconsistent = 22,
  not_cases_consistent = 68,
  case_label = "Answered True",
  not_case_label = "Answered False",
  reference_measure_name = "9th grade",
  comparison_measure_name = "12th grade",
  conf_level = 0.95
```

```
)

# To visualize the estimate
myplot_from_summary <- esci::plot_pdiff(estimate_from_summary)

# To conduct a hypothesis test
res_htest_from_summary <- esci::test_pdiff(estimate_from_summary)
```

`estimate_pdiff_two` *Estimates for a two-group study with a categorical outcome variable*

Description

Returns object `estimate_pdiff_two` is suitable for a simple two-group design with a categorical outcome variable. It provides estimates and CIs for the difference in proportions between the two groups, the odds ratio, and phi. You can pass raw data or summary data.

Usage

```
estimate_pdiff_two(
  data = NULL,
  outcome_variable = NULL,
  grouping_variable = NULL,
  comparison_cases = NULL,
  comparison_n = NULL,
  reference_cases = NULL,
  reference_n = NULL,
  case_label = 1,
  not_case_label = NULL,
  grouping_variable_levels = NULL,
  outcome_variable_name = "My outcome variable",
  grouping_variable_name = "My grouping variable",
  conf_level = 0.95,
  count_NA = FALSE
)
```

Arguments

`data`	For raw data - a data frame or tibble
`outcome_variable`	For raw data - The column name of the outcome variable which is a factor, or a vector that is a factor
`grouping_variable`	For raw data - The column name of the grouping variable which is a factor, or a vector that is a factor

`comparison_cases`	For summary data, a numeric integer >= 0
`comparison_n`	For summary data, a numeric integer >= comparison_events
`reference_cases`	For summary data, a numeric integer >= 0
`reference_n`	For summary data, a numeric integer >= reference_events
`case_label`	An optional numeric or character label for the case level.
`not_case_label`	An optional numeric or character label for the not case level.
`grouping_variable_levels`	For summary data - An optional vector of 2 group labels
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`grouping_variable_name`	Optional friendly name for the grouping variable. Defaults to 'My grouping variable' or the grouping variable column name if a data.frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`count_NA`	Logical to count NAs (TRUE) in total N or not (FALSE)

Details

Once you generate an estimate with this function, you can visualize it with `plot_mdiff()` and you can test hypotheses with `test_mdiff()`.

The estimated mean differences are from `statpsych::ci.prop2()`.

The estimated odds ratio is from `statpsych::ci.oddsratio()`.

The estimated correlation (phi) is from `statpsych::ci.phi()`.

Value

Returns object of class esci_estimate

- **es_proportion_difference**
  - *type* -
  - *outcome_variable_name* -
  - *case_label* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *effect_size_adjusted* -
  - *ta_LL* -

  - *ta_UL* -
- **es_odds_ratio**
  - *outcome_variable_name* -
  - *case_label* -
  - *grouping_variable_name* -
  - *effect* -
  - *effect_size* -
  - *SE* -
  - *LL* -
  - *UL* -
  - *ta_LL* -
  - *ta_UL* -
- **overview**
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *outcome_variable_name* -
  - *outcome_variable_level* -
  - *cases* -
  - *n* -
  - *P* -
  - *P_LL* -
  - *P_UL* -
  - *P_SE* -
  - *P_adjusted* -
  - *ta_LL* -
  - *ta_UL* -
- **es_phi**
  - *grouping_variable_name* -
  - *outcome_variable_name* -
  - *effect* -
  - *effect_size* -
  - *SE* -
  - *LL* -
  - *UL* -

Examples

```
data("data_campus_involvement")

estimate_from_raw <- esci::estimate_pdiff_two(
  esci::data_campus_involvement,
  CommuterStatus,
  Gender
```

```
)

# To visualize the estimate
myplot_from_raw <- esci::plot_pdiff(estimate_from_raw)

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_pdiff(estimate_from_raw)


# From summary_data
estimate_from_summary <- esci::estimate_pdiff_two(
  comparison_cases = 10,
  comparison_n = 20,
  reference_cases = 78,
  reference_n = 252,
  grouping_variable_levels = c("Original", "Replication"),
  conf_level = 0.95
)

# To visualize the estimate
myplot_from_summary <- esci::plot_pdiff(estimate_from_summary)

#' # To conduct a hypothesis test
res_htest_from_summary <- esci::test_pdiff(estimate_from_summary)
```

`estimate_proportion` *Estimates for a categorical variable with no grouping (single-group design)*

Description

`estimate_proportion` is suitable for a single group design with a categorical outcome variable. It estimates the population proportion for the frequency of each level of the outcome variable, with confidence intervals. You can pass raw data or summary data.

Usage

```
estimate_proportion(
  data = NULL,
  outcome_variable = NULL,
  cases = NULL,
  case_label = 1,
  outcome_variable_levels = NULL,
  outcome_variable_name = "My outcome variable",
  conf_level = 0.95,
  count_NA = FALSE
)
```

Arguments

`data`	For raw data - a data frame or tibble
`outcome_variable`	For raw data - The column name of the outcome variable, which must be a factor, or a vector that is a factor
`cases`	For summary data - A vector of cases
`case_label`	A numeric or string indicating which level of the factor to estimate. Defaults to 1, meaning first level is analyzed
`outcome_variable_levels`	For summary data - optional vector of 2 characters indicating name of the count level and name of the not count level. Defaults to "Affected" and "Not Affected"
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My outcome variable' or the outcome variable column name if a data frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`count_NA`	Logical to count NAs (TRUE) in total N or not (FALSE)

Details

Once you generate an estimate with this function, you can visualize it with `plot_proportion()`.

If you want to compare your estimate to a known value or reference, then use `estimate_pdiff_one()`.

The estimated proportions are from `statpsych::ci.prop1()` (renamed ci.prop as of statpsych 1.6).

Value

Returns an object of class esci_estimate

- **overview**
  - *outcome_variable_name* -
  - *outcome_variable_level* -
  - *cases* -
  - *n* -
  - *P* -
  - *P_LL* -
  - *P_UL* -
  - *P_SE* -
  - *P_adjusted* -
  - *ta_LL* -
  - *ta_UL* -
- **es_proportion**
  - *outcome_variable_name* -
  - *case_label* -

  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *effect_size_adjusted* -
  - *ta_LL* -
  - *ta_UL* -
  - *cases* -
  - *n* -

Examples

```
# From raw data
data("data_campus_involvement")

estimate_from_raw <- esci::estimate_proportion(
  esci::data_campus_involvement,
  CommuterStatus
)


# To visualize the estimate
myplot_from_raw <- esci::plot_proportion(estimate_from_raw)


# From summary data
estimate_from_summary <- esci::estimate_proportion(
  cases = c(8, 22-8),
  outcome_variable_levels = c("Affected", "Not Affected")
)

# To visualize the estimate
myplot_from_summary<- esci::plot_proportion(estimate_from_summary)
```

`estimate_r` *Estimates the linear correlation (Pearson's r) between two continuous variables*

Description

`estimate_r` is suitable for a design with two continuous variables. It estimates the linear correlation between two variables (Pearson's r) with a confidence interval. You can pass raw data or summary data.

Usage

```
estimate_r(
  data = NULL,
  x = NULL,
  y = NULL,
  r = NULL,
  n = NULL,
  x_variable_name = "My x variable",
  y_variable_name = "My y variable",
  conf_level = 0.95,
  save_raw_data = TRUE
)
```

Arguments

`data`	For raw data - A data frame or tibble
`x`	For raw data - The column name of the outcome variable, or a vector of numeric data
`y`	For raw data - The column name of the outcome variable, or a vector of numeric data
`r`	For summary data - A pearson's r correlation coefficient
`n`	For summary data - Sample size, an integer > 0
`x_variable_name`	Optional friendly name for the x variable. Defaults to 'My x variable' or the outcome variable column name if a data frame is passed.
`y_variable_name`	Optional friendly name for the y variable. Defaults to 'My y variable' or the outcome variable column name if a data frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`save_raw_data`	For raw data; defaults to TRUE; set to FALSE to save memory by not returning raw data in estimate object

Details

Reach for this function to conduct simple linear correlation or simple linear regression.

Once you generate an estimate with this function, you can visualize it with `plot_correlation()` and you can test hypotheses with `test_correlation()`. In addition, you can use `plot_scatter()` to visualize the raw data and to conduct a regression analysis that r returns predicted Y' values from a given X value.

The estimated correlation is from `statpsych::ci.cor()`, which uses the Fisher r-to-z approach.

Value

Returns object of class esci_estimate

- **overview**
  - *outcome_variable_name* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -
  - *q1* -
  - *q3* -
  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -
- **es_r**
  - *x_variable_name* -
  - *y_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *n* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
- **regression**
  - *component* -
  - *values* -
  - *LL* -
  - *UL* -
- **raw_data**
  - *x* -
  - *y* -
  - *fit* -
  - *lwr* -
  - *upr* -

Examples

```
# From raw data
data("data_thomason_1")

estimate_from_raw <- esci::estimate_r(
  esci::data_thomason_1,
  Pretest,
  Posttest
)

# To visualize the value of r
myplot_correlation <- esci::plot_correlation(estimate_from_raw)

# To visualize the data (scatterplot) and use regression to obtain Y' from X
myplot_scatter_from_raw <- esci::plot_scatter(estimate_from_raw, predict_from_x = 10)

# To evaluate a hypothesis (interval null from -0.1 to 0.1):
res_htest_from_raw <- esci::test_correlation(
  estimate_from_raw,
  rope = c(-0.1, 0.1)
)


# From summary data
estimate_from_summary <- esci::estimate_r(r = 0.536, n = 50)

# To visualize the value of r
myplot_correlation_from_summary <- esci::plot_correlation(estimate_from_summary)

# To evaluate a hypothesis (interval null from -0.1 to 0.1):
res_htest_from_summary <- esci::test_correlation(
  estimate_from_summary,
  rope = c(-0.1, 0.1)
)
```

`estimate_rdiff_two` *Estimates the difference in correlation for a design with two groups and two continuous outcome variables*

Description

Returns object `estimate_rdiff_two` is suitable for a simple two-group design with two continuous outcome variables where you want to estimate the difference in the strength of the relationship between the two groups. It estimate the linear correlation (Pearson's r) for each group and the difference in r, along with confidence intervals. You can pass raw data or summary data.

Returns effect sizes appropriate for estimating the linear relationship between two quantitative variables

Usage

```
estimate_rdiff_two(
  data = NULL,
  x = NULL,
  y = NULL,
  grouping_variable = NULL,
  comparison_r = NULL,
  comparison_n = NULL,
  reference_r = NULL,
  reference_n = NULL,
  grouping_variable_levels = NULL,
  x_variable_name = "My x variable",
  y_variable_name = "My y variable",
  grouping_variable_name = "My grouping variable",
  conf_level = 0.95,
  save_raw_data = TRUE
)
```

Arguments

`data`	For raw data - a dataframe or tibble
`x`	For raw data - The column name of the outcome variable, or a vector of numeric data
`y`	For raw data - The column name of the outcome variable, or a vector of numeric data
`grouping_variable`	For raw data, a vector that is a factor or the name of a factor column from data
`comparison_r`	For summary data, a pearson's r correlation coefficient
`comparison_n`	For summary data - An integer > 0
`reference_r`	For summary data, a pearson's r correlation coefficient
`reference_n`	For summary data - An integer > 0
`grouping_variable_levels`	For summary data - An optional vector of 2 group labels
`x_variable_name`	Optional friendly name for the x variable. Defaults to 'My x variable' or the outcome variable column name if a data frame is passed.
`y_variable_name`	Optional friendly name for the y variable. Defaults to 'My y variable' or the outcome variable column name if a data frame is passed.
`grouping_variable_name`	Optional friendly name for the grouping variable. Defaults to 'My grouping variable' or the grouping variable column name if a data.frame is passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`save_raw_data`	For raw data; defaults to TRUE; set to FALSE to save memory by not returning raw data in estimate object

Details

Once you generate an estimate with this function, you can visualize it with `plot_rdiff()` and you can test hypotheses with `test_rdiff()`. In addition, you can use `plot_scatter()` to visualize the raw data.

The estimated single-group r values are from `statpsych::ci.cor()`.

The difference in r values is from `statpsych::ci.cor2()`.

Value

Returns object of class esci_estimate

- **overview**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -
  - *q1* -
  - *q3* -
  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -
- **es_r_difference**
  - *type* -
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *x_variable_name* -
  - *y_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *n* -

  - *df* -
  - *ta_LL* -
  - *ta_UL* -
  - *rz* -
  - *sem* -
  - *z* -
  - *p* -
- **es_r**
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *x_variable_name* -
  - *y_variable_name* -
  - *effect* -
  - *effect_size* -
  - *LL* -
  - *UL* -
  - *SE* -
  - *n* -
  - *df* -
  - *ta_LL* -
  - *ta_UL* -
- **raw_data**
  - *x* -
  - *y* -
  - *grouping_variable* -

Examples

```
# From raw data
data("data_campus_involvement")

estimate_from_raw <- esci::estimate_rdiff_two(
  esci::data_campus_involvement,
  GPA,
  SWB,
  Gender
)

# To visualize the difference in r
myplot_from_raw <- esci::plot_rdiff(estimate_from_raw)

# To visualize the data (scatterplot) by group
myplot_scatter <- esci::plot_scatter(estimate_from_raw)

# To evaluate a hypothesis (by default: point null of exaclty 0):
res_htest_from_raw <- esci::test_rdiff(
```

```
  estimate_from_raw
)


# From summary data
estimate <- esci::estimate_rdiff_two(
  comparison_r = .53,
  comparison_n = 45,
  reference_r = .41,
  reference_n = 59,
  grouping_variable_levels = c("Females", "Males"),
  x_variable_name = "Satisfaction with life",
  y_variable_name = "Body satisfaction",
  grouping_variable_name = "Gender",
  conf_level = .95
)

myplot_from_summary <- esci::plot_rdiff(estimate)

# To evaluate a hypothesis (interval null from -0.1 to 0.1):
res_htest_from_summary <- esci::test_rdiff(
  estimate,
  rope = c(-0.1, 0.1)
)
```

`geom_meta_diamond_h` *Meta-analysis diamond*

Description

`geom_meta_diamond_h` creates a horizontal meta-analytic diamond defined by an `x` value (horizontal center of diamond), `xmin` and `xmax` values (for the horizontal ends of the diamond), a `y` value (for the vertical placement of the diamond) and a `height` (for the vertical height of the diamond). Note the use of `xmin` and `xmax` allows for representation of asymmetric confidence intervals with this geom.

Usage

```
geom_meta_diamond_h(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.
`data`	The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).
`stat`	The statistical transformation to use on the data for this layer. When using a `geom_*()` function to construct a layer, the `stat` argument can be used the override the default coupling between geoms and stats. The `stat` argument accepts the following: • A `Stat` ggproto subclass, for example `StatCount`. • A string naming the stat. To give the stat as a string, strip the function name of the `stat_` prefix. For example, to use `stat_count()`, give the stat as `"count"`. • For more information and other ways to specify the stat, see the layer stat documentation.
`position`	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following: • The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`. • For more information and other ways to specify the position, see the layer position documentation.
`...`	Other arguments passed to the geom. These are often aesthetics, used to set an aesthetic to a fixed value, like `colour = "red"` or `linewidth = 3` (see **Aesthetics**, below). ##Aesthetics ## `geom_meta_diamond_h` understands the following aesthetics (required are in bold): • `x` - The horizontal center of the diamond • `y` - The vertical placement of the diamond • `xmin` - The left-side start of the diamond • `xmax` - The right-side start of the diamond • `height` - The vertical span of the diamond

`na.rm`	If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.
`show.legend`	logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.
`inherit.aes`	If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.

Examples

```
# example code
library(ggplot2)

my_effects <- data.frame(
  effect_size = c(1, 2, 1, 0),
  UL = c(2, 3, 2, 1),
  LL = c(0, 1, 0, -1),
  y = c(1, 2, 3, 4)
)


myplot <- ggplot2::ggplot()
myplot <- myplot + geom_meta_diamond_h(
  data = my_effects,
  ggplot2::aes(
    x = effect_size,
    xmin = LL,
    xmax = UL,
    y = y
  ),
  height = 0.25,
  color = "black",
  fill = "red",
)
```

`jamovicorrelation` *Correlations: Single Group*

Description

Correlations: Single Group

Usage

```
jamovicorrelation(
  switch = "from_raw",
  data,
  x,
  y,
  r = " ",
  n = " ",
  x_variable_name = "X variable",
  y_variable_name = "Y variable",
  conf_level = 95,
  show_details = FALSE,
  do_regression = FALSE,
  show_line = FALSE,
  show_line_CI = FALSE,
  show_residuals = FALSE,
  show_PI = FALSE,
  show_mean_lines = FALSE,
  show_r = FALSE,
  plot_as_z = FALSE,
  predict_from_x = " ",
  evaluate_hypotheses = FALSE,
  null_value = "0",
  null_boundary = "0",
  alpha = 0.05,
  null_color = "#A40122",
  es_plot_width = "300",
  es_plot_height = "400",
  sp_plot_width = "650",
  sp_plot_height = "650",
  ymin = "-1",
  ymax = "1",
  ybreaks = "auto",
  ylab = "auto",
  xlab = "auto",
  axis.text.y = "14",
  axis.title.y = "15",
  axis.text.x = "14",
  axis.title.x = "15",
  error_layout = "none",
  sp_ymin = "auto",
  sp_ymax = "auto",
  sp_ybreaks = "auto",
  sp_xmin = "auto",
  sp_xmax = "auto",
  sp_xbreaks = "auto",
  sp_ylab = "auto",
  sp_xlab = "auto",
```

```
    sp_axis.text.y = "14",
    sp_axis.title.y = "15",
    sp_axis.text.x = "14",
    sp_axis.title.x = "15",
    shape_summary = "circle filled",
    color_summary = "#008DF9",
    fill_summary = "#008DF9",
    size_summary = "4",
    alpha_summary = "1",
    linetype_summary = "solid",
    color_interval = "black",
    size_interval = "3",
    alpha_interval = "1",
    alpha_error = "1",
    fill_error = "gray75",
    sp_shape_raw_reference = "circle filled",
    sp_color_raw_reference = "black",
    sp_fill_raw_reference = "#008DF9",
    sp_size_raw_reference = "3",
    sp_alpha_raw_reference = ".25",
    sp_linetype_summary_reference = "solid",
    sp_color_summary_reference = "#008DF9",
    sp_size_summary_reference = "3",
    sp_alpha_summary_reference = ".25",
    sp_linetype_PI_reference = "dotted",
    sp_color_PI_reference = "#E20134",
    sp_size_PI_reference = "2",
    sp_alpha_PI_reference = "1",
    sp_linetype_residual_reference = "solid",
    sp_color_residual_reference = "#E20134",
    sp_size_residual_reference = "1",
    sp_alpha_residual_reference = "1",
    sp_prediction_label = "5",
    sp_prediction_color = "#E20134",
    sp_linetype_CI = "solid",
    sp_color_CI = "#008DF9",
    sp_size_CI = "4",
    sp_alpha_CI = "1",
    sp_linetype_ref = "dotted",
    sp_color_ref = "gray60",
    sp_size_ref = "1",
    sp_alpha_ref = "1",
    sp_linetype_PI = "solid",
    sp_color_PI = "#E20134",
    sp_size_PI = "2",
    sp_alpha_PI = "1"
  )
```

Arguments

`switch`	.
`data`	.
`x`	.
`y`	.
`r`	.
`n`	.
`x_variable_name`	.
`y_variable_name`	.
`conf_level`	.
`show_details`	.
`do_regression`	.
`show_line`	.
`show_line_CI`	.
`show_residuals`	.
`show_PI`	.
`show_mean_lines`	.
`show_r`	.
`plot_as_z`	.
`predict_from_x`	.
`evaluate_hypotheses`	.
`null_value`	.
`null_boundary`	.
`alpha`	.
`null_color`	.
`es_plot_width`	.
`es_plot_height`	.
`sp_plot_width`	.
`sp_plot_height`	.
`ymin`	.
`ymax`	.
`ybreaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.

`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`error_layout`	.
`sp_ymin`	.
`sp_ymax`	.
`sp_ybreaks`	.
`sp_xmin`	.
`sp_xmax`	.
`sp_xbreaks`	.
`sp_ylab`	.
`sp_xlab`	.
`sp_axis.text.y`	.
`sp_axis.title.y`	.
`sp_axis.text.x`	.
`sp_axis.title.x`	.
`shape_summary`	.
`color_summary`	.
`fill_summary`	.
`size_summary`	.
`alpha_summary`	.
`linetype_summary`	.
`color_interval`	.
`size_interval`	.
`alpha_interval`	.
`alpha_error`	.
`fill_error`	.
`sp_shape_raw_reference`	.
`sp_color_raw_reference`	.
`sp_fill_raw_reference`	.
`sp_size_raw_reference`	.
`sp_alpha_raw_reference`	.

`sp_linetype_summary_reference`	.
`sp_color_summary_reference`	.
`sp_size_summary_reference`	.
`sp_alpha_summary_reference`	.
`sp_linetype_PI_reference`	.
`sp_color_PI_reference`	.
`sp_size_PI_reference`	.
`sp_alpha_PI_reference`	.
`sp_linetype_residual_reference`	.
`sp_color_residual_reference`	.
`sp_size_residual_reference`	.
`sp_alpha_residual_reference`	.
`sp_prediction_label`	.
`sp_prediction_color`	.
`sp_linetype_CI`	.
`sp_color_CI`	.
`sp_size_CI`	.
`sp_alpha_CI`	.
`sp_linetype_ref`	.
`sp_color_ref`	.
`sp_size_ref`	.
`sp_alpha_ref`	.
`sp_linetype_PI`	.
`sp_color_PI`	.
`sp_size_PI`	.
`sp_alpha_PI`	.

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$es_r`	a table
`results$regression`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$scatter_plot_warnings`	a html
`results$scatter_plots`	an image
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamovidescribe` *Describe*

Description

Describe

Usage

```
jamovidescribe(
  data,
  outcome_variable,
  show_details = FALSE,
  mark_mean = FALSE,
  mark_median = FALSE,
  mark_sd = FALSE,
  mark_quartiles = FALSE,
  mark_z_lines = FALSE,
  mark_percentile = "0",
  histogram_bins = "12",
  es_plot_width = "500",
  es_plot_height = "400",
  ymin = "auto",
  ymax = "auto",
  breaks = "auto",
```

```
  xmin = "auto",
  xmax = "auto",
  xbreaks = "auto",
  ylab = "auto",
  xlab = "auto",
  axis.text.y = "14",
  axis.title.y = "15",
  axis.text.x = "14",
  axis.title.x = "15",
  fill_regular = "#008DF9",
  fill_highlighted = "#E20134",
  color = "black"
)
```

Arguments

`data`	.
`outcome_variable`	.
`show_details`	.
`mark_mean`	.
`mark_median`	.
`mark_sd`	.
`mark_quartiles`	.
`mark_z_lines`	.
`mark_percentile`	.
`histogram_bins`	.
`es_plot_width`	.
`es_plot_height`	.
`ymin`	.
`ymax`	.
`breaks`	.
`xmin`	.
`xmax`	.
`xbreaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.

`fill_regular`	.
`fill_highlighted`	.
`color`	.

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$describe_plot_warnings`	a html
`results$describe_plot`	an image
`results$describe_dotplot`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamovimagnitude` *Means and Medians: Single Group*

Description

Means and Medians: Single Group

Usage

```
jamovimagnitude(
  switch = "from_raw",
  data,
  outcome_variable,
  mean = "",
  sd = "",
  n = "",
  outcome_variable_name = "Outcome variable",
  conf_level = 95,
  effect_size = "mean",
  show_details = FALSE,
  show_calculations = FALSE,
  evaluate_hypotheses = FALSE,
  null_value = "0",
  null_boundary = "0",
  alpha = 0.05,
```

```
  null_color = "#A40122",
  es_plot_width = "300",
  es_plot_height = "400",
  ymin = "auto",
  ymax = "auto",
  breaks = "auto",
  ylab = "auto",
  xlab = "auto",
  axis.text.y = "14",
  axis.title.y = "15",
  axis.text.x = "14",
  axis.title.x = "15",
  error_layout = "halfeye",
  error_scale = "0.20",
  error_nudge = "0.3",
  data_layout = "random",
  data_spread = "0.25",
  shape_raw = "circle filled",
  shape_summary = "circle filled",
  color_raw = "#008DF9",
  color_summary = "#008DF9",
  fill_raw = "NA",
  fill_summary = "#008DF9",
  size_raw = "2",
  size_summary = "4",
  alpha_raw = "1",
  alpha_summary = "1",
  linetype_summary = "solid",
  color_interval = "black",
  size_interval = "3",
  alpha_interval = "1",
  alpha_error = "1",
  fill_error = "gray75"
)
```

Arguments

`switch`	.
`data`	.
`outcome_variable`	.
`mean`	.
`sd`	.
`n`	.
`outcome_variable_name`	.
`conf_level`	.

`effect_size`	.
`show_details`	.
`show_calculations`	.
`evaluate_hypotheses`	.
`null_value`	.
`null_boundary`	.
`alpha`	.
`null_color`	.
`es_plot_width`	.
`es_plot_height`	.
`ymin`	.
`ymax`	.
`breaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`error_layout`	.
`error_scale`	.
`error_nudge`	.
`data_layout`	.
`data_spread`	.
`shape_raw`	.
`shape_summary`	.
`color_raw`	.
`color_summary`	.
`fill_raw`	.
`fill_summary`	.
`size_raw`	.
`size_summary`	.
`alpha_raw`	.
`alpha_summary`	.
`linetype_summary`	.

`color_interval`	.
`size_interval`	.
`alpha_interval`	.
`alpha_error`	.
`fill_error`	.

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$es_smd`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$magnitude_plot_warnings`	a html
`results$magnitude_plot`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamovimdiff2x2` *Means and Medians: 2x2 Factorial*

Description

Means and Medians: 2x2 Factorial

Usage

```
jamovimdiff2x2(
  design = "fully_between",
  switch = "from_raw",
  data,
  outcome_variable,
  grouping_variable_A,
  grouping_variable_B,
  outcome_variable_level1,
  outcome_variable_level2,
  outcome_variable_name_bs = "My outcome variable",
  grouping_variable,
  repeated_measures_name = "Time",
```

```
  outcome_variable_name = "My outcome variable",
  A1_label = "A1 level",
  A2_label = "A2 level",
  B1_label = "B1 level",
  B2_label = "B2 level",
  A_label = "Variable A",
  B_label = "Variable B",
  A1B1_mean = " ",
  A1B1_sd = " ",
  A1B1_n = " ",
  A1B2_mean = " ",
  A1B2_sd = " ",
  A1B2_n = " ",
  A2B1_mean = " ",
  A2B1_sd = " ",
  A2B1_n = " ",
  A2B2_mean = " ",
  A2B2_sd = " ",
  A2B2_n = " ",
  conf_level = 95,
  effect_size = "mean_difference",
  assume_equal_variance = TRUE,
  show_details = FALSE,
  show_interaction_plot = FALSE,
  show_CI = FALSE,
  evaluate_hypotheses = FALSE,
  null_value = "0",
  null_boundary = "0",
  rope_units = "raw",
  alpha = 0.05,
  null_color = "#A40122",
  es_plot_width = "700",
  es_plot_height = "400",
  ymin = "auto",
  ymax = "auto",
  ybreaks = "auto",
  ylab = "auto",
  xlab = "auto",
  axis.text.y = "14",
  axis.title.y = "15",
  axis.text.x = "14",
  axis.title.x = "15",
  simple_contrast_labels = FALSE,
  error_layout = "halfeye",
  error_scale = "0.25",
  error_nudge = "0.5",
  data_layout = "random",
  data_spread = "0.20",
```

```
difference_axis_units = "raw",
difference_axis_breaks = "auto",
shape_raw_reference = "circle filled",
shape_raw_comparison = "circle filled",
shape_raw_unused = "circle filled",
shape_summary_reference = "circle filled",
shape_summary_comparison = "circle filled",
shape_summary_unused = "circle filled",
shape_summary_difference = "triangle filled",
color_raw_reference = "#008DF9",
color_raw_comparison = "#009F81",
color_raw_unused = "gray65",
color_summary_reference = "#008DF9",
color_summary_comparison = "#009F81",
color_summary_unused = "gray65",
color_summary_difference = "black",
fill_raw_reference = "NA",
fill_raw_comparison = "NA",
fill_raw_unused = "NA",
fill_summary_reference = "#008DF9",
fill_summary_comparison = "#009F81",
fill_summary_unused = "gray65",
fill_summary_difference = "black",
size_raw_reference = "2",
size_raw_comparison = "2",
size_raw_unused = "1",
size_summary_reference = "4",
size_summary_comparison = "4",
size_summary_unused = "4",
size_summary_difference = "4",
alpha_raw_reference = "1",
alpha_raw_comparison = "1",
alpha_raw_unused = "1",
alpha_summary_reference = "1",
alpha_summary_comparison = "1",
alpha_summary_unused = "1",
alpha_summary_difference = "1",
linetype_summary_reference = "solid",
linetype_summary_comparison = "solid",
linetype_summary_unused = "solid",
linetype_summary_difference = "solid",
color_interval_reference = "black",
color_interval_comparison = "black",
color_interval_unused = "gray65",
color_interval_difference = "black",
size_interval_reference = "3",
size_interval_comparison = "3",
size_interval_unused = "3",
```

```
    size_interval_difference = "3",
    alpha_interval_reference = "1",
    alpha_interval_comparison = "1",
    alpha_interval_unused = "1",
    alpha_interval_difference = "1",
    alpha_error_reference = "1",
    alpha_error_comparison = "1",
    alpha_error_unused = "1",
    alpha_error_difference = "1",
    fill_error_reference = "gray75",
    fill_error_comparison = "gray75",
    fill_error_unused = "gray75",
    fill_error_difference = "gray75"
  )
```

Arguments

`design`	.
`switch`	.
`data`	.
`outcome_variable`	.
`grouping_variable_A`	.
`grouping_variable_B`	.
`outcome_variable_level1`	.
`outcome_variable_level2`	.
`outcome_variable_name_bs`	.
`grouping_variable`	.
`repeated_measures_name`	.
`outcome_variable_name`	.
`A1_label`	.
`A2_label`	.
`B1_label`	.
`B2_label`	.
`A_label`	.
`B_label`	.
`A1B1_mean`	.

`A1B1_sd`	.
`A1B1_n`	.
`A1B2_mean`	.
`A1B2_sd`	.
`A1B2_n`	.
`A2B1_mean`	.
`A2B1_sd`	.
`A2B1_n`	.
`A2B2_mean`	.
`A2B2_sd`	.
`A2B2_n`	.
`conf_level`	.
`effect_size`	.
`assume_equal_variance`	.
`show_details`	.
`show_interaction_plot`	.
`show_CI`	.
`evaluate_hypotheses`	.
`null_value`	.
`null_boundary`	.
`rope_units`	.
`alpha`	.
`null_color`	.
`es_plot_width`	.
`es_plot_height`	.
`ymin`	.
`ymax`	.
`ybreaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`simple_contrast_labels`	.

`error_layout`	.
`error_scale`	.
`error_nudge`	.
`data_layout`	.
`data_spread`	.
`difference_axis_units`	.
`difference_axis_breaks`	.
`shape_raw_reference`	.
`shape_raw_comparison`	.
`shape_raw_unused`	.
`shape_summary_reference`	.
`shape_summary_comparison`	.
`shape_summary_unused`	.
`shape_summary_difference`	.
`color_raw_reference`	.
`color_raw_comparison`	.
`color_raw_unused`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_unused`	.
`color_summary_difference`	.
`fill_raw_reference`	.
`fill_raw_comparison`	.
`fill_raw_unused`	.
`fill_summary_reference`	.
`fill_summary_comparison`	.

`fill_summary_unused`	.
`fill_summary_difference`	.
`size_raw_reference`	.
`size_raw_comparison`	.
`size_raw_unused`	.
`size_summary_reference`	.
`size_summary_comparison`	.
`size_summary_unused`	.
`size_summary_difference`	.
`alpha_raw_reference`	.
`alpha_raw_comparison`	.
`alpha_raw_unused`	.
`alpha_summary_reference`	.
`alpha_summary_comparison`	.
`alpha_summary_unused`	.
`alpha_summary_difference`	.
`linetype_summary_reference`	.
`linetype_summary_comparison`	.
`linetype_summary_unused`	.
`linetype_summary_difference`	.
`color_interval_reference`	.
`color_interval_comparison`	.
`color_interval_unused`	.
`color_interval_difference`	.

`size_interval_reference`

.

`size_interval_comparison`

.

`size_interval_unused`

.

`size_interval_difference`

.

`alpha_interval_reference`

.

`alpha_interval_comparison`

.

`alpha_interval_unused`

.

`alpha_interval_difference`

.

`alpha_error_reference`

.

`alpha_error_comparison`

.

`alpha_error_unused`

.

`alpha_error_difference`

.

`fill_error_reference`

.

`fill_error_comparison`

.

`fill_error_unused`

.

`fill_error_difference`

.

Value

A results object containing:

`results$analysis_type`	a html
`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$es_median_difference`	a table
`results$es_mean_difference`	a table
`results$es_smd`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$estimation_plot_warnings`	a html
`results$main_effect_A`	an image

`results$main_effect_B`	an image
`results$interaction`	an image
`results$interaction_plot`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamovimdiffindcontrast`

Means and Medians: Independent Groups Contrast

Description

Means and Medians: Independent Groups Contrast

Usage

```
jamovimdiffindcontrast(
  switch = "from_raw",
  data,
  outcome_variable,
  grouping_variable,
  means,
  sds,
  ns,
  grouping_variable_levels,
  outcome_variable_name = "Outcome variable",
  grouping_variable_name = "Grouping variable",
  comparison_labels = " ",
  reference_labels = " ",
  conf_level = 95,
  effect_size = "mean_difference",
  assume_equal_variance = TRUE,
  show_details = FALSE,
  evaluate_hypotheses = FALSE,
  null_value = "0",
  null_boundary = "0",
  rope_units = "raw",
  alpha = 0.05,
  null_color = "#A40122",
  es_plot_width = "550",
  es_plot_height = "450",
  ymin = "auto",
  ymax = "auto",
```

```
ybreaks = "auto",
ylab = "auto",
xlab = "auto",
axis.text.y = "14",
axis.title.y = "15",
axis.text.x = "14",
axis.title.x = "15",
simple_contrast_labels = TRUE,
error_layout = "halfeye",
error_scale = "0.25",
error_nudge = "0.5",
data_layout = "random",
data_spread = "0.20",
difference_axis_units = "raw",
difference_axis_breaks = "auto",
shape_raw_reference = "circle filled",
shape_raw_comparison = "circle filled",
shape_raw_unused = "circle filled",
shape_summary_reference = "circle filled",
shape_summary_comparison = "circle filled",
shape_summary_unused = "circle filled",
shape_summary_difference = "triangle filled",
color_raw_reference = "#008DF9",
color_raw_comparison = "#009F81",
color_raw_unused = "gray65",
color_summary_reference = "#008DF9",
color_summary_comparison = "#009F81",
color_summary_unused = "gray65",
color_summary_difference = "black",
fill_raw_reference = "NA",
fill_raw_comparison = "NA",
fill_raw_unused = "NA",
fill_summary_reference = "#008DF9",
fill_summary_comparison = "#009F81",
fill_summary_unused = "gray65",
fill_summary_difference = "black",
size_raw_reference = "2",
size_raw_comparison = "2",
size_raw_unused = "1",
size_summary_reference = "4",
size_summary_comparison = "4",
size_summary_unused = "4",
size_summary_difference = "4",
alpha_raw_reference = "1",
alpha_raw_comparison = "1",
alpha_raw_unused = "1",
alpha_summary_reference = "1",
alpha_summary_comparison = "1",
```

```
  alpha_summary_unused = "1",
  alpha_summary_difference = "1",
  linetype_summary_reference = "solid",
  linetype_summary_comparison = "solid",
  linetype_summary_unused = "solid",
  linetype_summary_difference = "solid",
  color_interval_reference = "black",
  color_interval_comparison = "black",
  color_interval_unused = "gray65",
  color_interval_difference = "black",
  size_interval_reference = "3",
  size_interval_comparison = "3",
  size_interval_unused = "3",
  size_interval_difference = "3",
  alpha_interval_reference = "1",
  alpha_interval_comparison = "1",
  alpha_interval_unused = "1",
  alpha_interval_difference = "1",
  alpha_error_reference = "1",
  alpha_error_comparison = "1",
  alpha_error_unused = "1",
  alpha_error_difference = "1",
  fill_error_reference = "gray75",
  fill_error_comparison = "gray75",
  fill_error_unused = "gray75",
  fill_error_difference = "gray75"
)
```

Arguments

`switch`	.
`data`	.
`outcome_variable`	.
`grouping_variable`	.
`means`	.
`sds`	.
`ns`	.
`grouping_variable_levels`	.
`outcome_variable_name`	.
`grouping_variable_name`	.
`comparison_labels`	.

`reference_labels`	.
`conf_level`	.
`effect_size`	.
`assume_equal_variance`	.
`show_details`	.
`evaluate_hypotheses`	.
`null_value`	.
`null_boundary`	.
`rope_units`	.
`alpha`	.
`null_color`	.
`es_plot_width`	.
`es_plot_height`	.
`ymin`	.
`ymax`	.
`ybreaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`simple_contrast_labels`	.
`error_layout`	.
`error_scale`	.
`error_nudge`	.
`data_layout`	.
`data_spread`	.
`difference_axis_units`	.
`difference_axis_breaks`	.
`shape_raw_reference`	.
`shape_raw_comparison`	.

`shape_raw_unused`

.

`shape_summary_reference`

.

`shape_summary_comparison`

.

`shape_summary_unused`

.

`shape_summary_difference`

.

`color_raw_reference`

.

`color_raw_comparison`

.

`color_raw_unused`

.

`color_summary_reference`

.

`color_summary_comparison`

.

`color_summary_unused`

.

`color_summary_difference`

.

`fill_raw_reference`

.

`fill_raw_comparison`

.

`fill_raw_unused`

.

`fill_summary_reference`

.

`fill_summary_comparison`

.

`fill_summary_unused`

.

`fill_summary_difference`

.

`size_raw_reference`

.

`size_raw_comparison`

.

`size_raw_unused`

.

`size_summary_reference`

.

`size_summary_comparison`

.

`size_summary_unused`	.
`size_summary_difference`	.
`alpha_raw_reference`	.
`alpha_raw_comparison`	.
`alpha_raw_unused`	.
`alpha_summary_reference`	.
`alpha_summary_comparison`	.
`alpha_summary_unused`	.
`alpha_summary_difference`	.
`linetype_summary_reference`	.
`linetype_summary_comparison`	.
`linetype_summary_unused`	.
`linetype_summary_difference`	.
`color_interval_reference`	.
`color_interval_comparison`	.
`color_interval_unused`	.
`color_interval_difference`	.
`size_interval_reference`	.
`size_interval_comparison`	.
`size_interval_unused`	.
`size_interval_difference`	.
`alpha_interval_reference`	.
`alpha_interval_comparison`	.
`alpha_interval_unused`	.

`alpha_interval_difference`
.

`alpha_error_reference`
.

`alpha_error_comparison`
.

`alpha_error_unused`
.

`alpha_error_difference`
.

`fill_error_reference`
.

`fill_error_comparison`
.

`fill_error_unused`
.

`fill_error_difference`
.

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$es_median_difference`	a table
`results$es_mean_difference`	a table
`results$es_smd`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an array of images

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamovimdiffpaired` *Means and Medians: Paired*

Description

Means and Medians: Paired

Usage

```
jamovimdiffpaired(
  switch = "from_raw",
  data,
  reference_measure,
  comparison_measure,
  comparison_mean = " ",
  comparison_sd = " ",
  reference_mean = " ",
  reference_sd = " ",
  n = " ",
  enter_r_or_sdiff = "enter_r",
  correlation = " ",
  sdiff = " ",
  comparison_measure_name = "Comparison measure",
  reference_measure_name = "Reference measure",
  conf_level = 95,
  effect_size = "mean_difference",
  show_ratio = FALSE,
  show_details = FALSE,
  show_calculations = FALSE,
  evaluate_hypotheses = FALSE,
  null_value = "0",
  null_boundary = "0",
  rope_units = "raw",
  alpha = 0.05,
  null_color = "#A40122",
  es_plot_width = "600",
  es_plot_height = "400",
  ymin = "auto",
  ymax = "auto",
  ybreaks = "auto",
  ylab = "auto",
  xlab = "auto",
  axis.text.y = "14",
  axis.title.y = "15",
  axis.text.x = "14",
  axis.title.x = "15",
  simple_contrast_labels = TRUE,
  error_layout = "halfeye",
  error_scale = "0.25",
  error_nudge = "0.5",
  data_layout = "random",
  data_spread = "0.20",
  difference_axis_units = "raw",
  difference_axis_breaks = "auto",
  shape_raw_reference = "circle filled",
  shape_raw_comparison = "circle filled",
```

```
  shape_raw_difference = "triangle filled",
  shape_summary_reference = "circle filled",
  shape_summary_comparison = "circle filled",
  shape_summary_difference = "triangle filled",
  color_raw_reference = "#008DF9",
  color_raw_comparison = "#008DF9",
  color_raw_difference = "#E20134",
  color_summary_reference = "#008DF9",
  color_summary_comparison = "#008DF9",
  color_summary_difference = "black",
  fill_raw_reference = "NA",
  fill_raw_comparison = "NA",
  fill_raw_difference = "NA",
  fill_summary_reference = "#008DF9",
  fill_summary_comparison = "#008DF9",
  fill_summary_difference = "black",
  size_raw_reference = "2",
  size_raw_comparison = "2",
  size_raw_difference = "2",
  size_summary_reference = "4",
  size_summary_comparison = "4",
  size_summary_difference = "4",
  alpha_raw_reference = "1",
  alpha_raw_comparison = "1",
  alpha_raw_difference = "1",
  alpha_summary_reference = "1",
  alpha_summary_comparison = "1",
  alpha_summary_difference = "1",
  linetype_summary_reference = "solid",
  linetype_summary_comparison = "solid",
  linetype_summary_difference = "solid",
  color_interval_reference = "black",
  color_interval_comparison = "black",
  color_interval_difference = "black",
  size_interval_reference = "3",
  size_interval_comparison = "3",
  size_interval_difference = "3",
  alpha_interval_reference = "1",
  alpha_interval_comparison = "1",
  alpha_interval_difference = "1",
  alpha_error_reference = "1",
  alpha_error_comparison = "1",
  alpha_error_difference = "1",
  fill_error_reference = "gray75",
  fill_error_comparison = "gray75",
  fill_error_difference = "gray75"
)
```

Arguments

`switch`	.
`data`	.
`reference_measure`	.
`comparison_measure`	.
`comparison_mean`	.
`comparison_sd`	.
`reference_mean`	.
`reference_sd`	.
`n`	.
`enter_r_or_sdiff`	.
`correlation`	.
`sdiff`	.
`comparison_measure_name`	.
`reference_measure_name`	.
`conf_level`	.
`effect_size`	.
`show_ratio`	.
`show_details`	.
`show_calculations`	.
`evaluate_hypotheses`	.
`null_value`	.
`null_boundary`	.
`rope_units`	.
`alpha`	.
`null_color`	.
`es_plot_width`	.
`es_plot_height`	.
`ymin`	.
`ymax`	.
`ybreaks`	.
`ylab`	.

`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`simple_contrast_labels`	.
`error_layout`	.
`error_scale`	.
`error_nudge`	.
`data_layout`	.
`data_spread`	.
`difference_axis_units`	.
`difference_axis_breaks`	.
`shape_raw_reference`	.
`shape_raw_comparison`	.
`shape_raw_difference`	.
`shape_summary_reference`	.
`shape_summary_comparison`	.
`shape_summary_difference`	.
`color_raw_reference`	.
`color_raw_comparison`	.
`color_raw_difference`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_difference`	.
`fill_raw_reference`	.
`fill_raw_comparison`	.

`fill_raw_difference`

.

`fill_summary_reference`

.

`fill_summary_comparison`

.

`fill_summary_difference`

.

`size_raw_reference`

.

`size_raw_comparison`

.

`size_raw_difference`

.

`size_summary_reference`

.

`size_summary_comparison`

.

`size_summary_difference`

.

`alpha_raw_reference`

.

`alpha_raw_comparison`

.

`alpha_raw_difference`

.

`alpha_summary_reference`

.

`alpha_summary_comparison`

.

`alpha_summary_difference`

.

`linetype_summary_reference`

.

`linetype_summary_comparison`

.

`linetype_summary_difference`

.

`color_interval_reference`

.

`color_interval_comparison`

.

`color_interval_difference`

.

`size_interval_reference`

.

`size_interval_comparison`

.

`size_interval_difference`	.
`alpha_interval_reference`	.
`alpha_interval_comparison`	.
`alpha_interval_difference`	.
`alpha_error_reference`	.
`alpha_error_comparison`	.
`alpha_error_difference`	.
`fill_error_reference`	.
`fill_error_comparison`	.
`fill_error_difference`	.

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$es_r`	a table
`results$es_mean_difference`	a table
`results$es_smd`	a table
`results$es_mean_ratio`	a table
`results$es_median_difference`	a table
`results$es_median_ratio`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamovimdifftwo` *Means and Medians: Two Groups*

Description

Means and Medians: Two Groups

Usage

```
jamovimdifftwo(
  switch = "from_raw",
  data,
  outcome_variable,
  grouping_variable,
  reference_level_name = "Reference group",
  reference_mean = " ",
  reference_sd = " ",
  reference_n = " ",
  comparison_level_name = "Comparison group",
  comparison_mean = " ",
  comparison_sd = " ",
  comparison_n = " ",
  outcome_variable_name = "Outcome variable",
  grouping_variable_name = "Grouping variable",
  conf_level = 95,
  assume_equal_variance = TRUE,
  effect_size = "mean_difference",
  show_ratio = FALSE,
  switch_comparison_order = FALSE,
  show_details = FALSE,
  show_calculations = FALSE,
  evaluate_hypotheses = FALSE,
  null_value = "0",
  null_boundary = "0",
  rope_units = "raw",
  alpha = 0.05,
  null_color = "#A40122",
  es_plot_width = "600",
  es_plot_height = "400",
  ymin = "auto",
  ymax = "auto",
  ybreaks = "auto",
  ylab = "auto",
  xlab = "auto",
  axis.text.y = "14",
  axis.title.y = "15",
  axis.text.x = "14",
```

```
  axis.title.x = "15",
  simple_contrast_labels = TRUE,
  error_layout = "halfeye",
  error_scale = "0.25",
  error_nudge = "0.5",
  data_layout = "random",
  data_spread = "0.20",
  difference_axis_units = "raw",
  difference_axis_breaks = "auto",
  shape_raw_reference = "circle filled",
  shape_raw_comparison = "circle filled",
  shape_summary_reference = "circle filled",
  shape_summary_comparison = "circle filled",
  shape_summary_difference = "triangle filled",
  color_raw_reference = "#008DF9",
  color_raw_comparison = "#009F81",
  color_summary_reference = "#008DF9",
  color_summary_comparison = "#009F81",
  color_summary_difference = "black",
  fill_raw_reference = "NA",
  fill_raw_comparison = "NA",
  fill_summary_reference = "#008DF9",
  fill_summary_comparison = "#009F81",
  fill_summary_difference = "black",
  size_raw_reference = "2",
  size_raw_comparison = "2",
  size_summary_reference = "4",
  size_summary_comparison = "4",
  size_summary_difference = "4",
  alpha_raw_reference = "1",
  alpha_raw_comparison = "1",
  alpha_summary_reference = "1",
  alpha_summary_comparison = "1",
  alpha_summary_difference = "1",
  linetype_summary_reference = "solid",
  linetype_summary_comparison = "solid",
  linetype_summary_difference = "solid",
  color_interval_reference = "black",
  color_interval_comparison = "black",
  color_interval_difference = "black",
  size_interval_reference = "3",
  size_interval_comparison = "3",
  size_interval_difference = "3",
  alpha_interval_reference = "1",
  alpha_interval_comparison = "1",
  alpha_interval_difference = "1",
  alpha_error_reference = "1",
  alpha_error_comparison = "1",
```

```
    alpha_error_difference = "1",
    fill_error_reference = "gray75",
    fill_error_comparison = "gray75",
    fill_error_difference = "gray75"
  )
```

Arguments

`switch`	.
`data`	.
`outcome_variable`	.
`grouping_variable`	.
`reference_level_name`	.
`reference_mean`	.
`reference_sd`	.
`reference_n`	.
`comparison_level_name`	.
`comparison_mean`	.
`comparison_sd`	.
`comparison_n`	.
`outcome_variable_name`	.
`grouping_variable_name`	.
`conf_level`	.
`assume_equal_variance`	.
`effect_size`	.
`show_ratio`	.
`switch_comparison_order`	.
`show_details`	.
`show_calculations`	.
`evaluate_hypotheses`	.
`null_value`	.
`null_boundary`	.
`rope_units`	.

`alpha`	.
`null_color`	.
`es_plot_width`	.
`es_plot_height`	.
`ymin`	.
`ymax`	.
`ybreaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`simple_contrast_labels`	.
`error_layout`	.
`error_scale`	.
`error_nudge`	.
`data_layout`	.
`data_spread`	.
`difference_axis_units`	.
`difference_axis_breaks`	.
`shape_raw_reference`	.
`shape_raw_comparison`	.
`shape_summary_reference`	.
`shape_summary_comparison`	.
`shape_summary_difference`	.
`color_raw_reference`	.
`color_raw_comparison`	.
`color_summary_reference`	.
`color_summary_comparison`	.

`color_summary_difference`
.

`fill_raw_reference`
.

`fill_raw_comparison`
.

`fill_summary_reference`
.

`fill_summary_comparison`
.

`fill_summary_difference`
.

`size_raw_reference`
.

`size_raw_comparison`
.

`size_summary_reference`
.

`size_summary_comparison`
.

`size_summary_difference`
.

`alpha_raw_reference`
.

`alpha_raw_comparison`
.

`alpha_summary_reference`
.

`alpha_summary_comparison`
.

`alpha_summary_difference`
.

`linetype_summary_reference`
.

`linetype_summary_comparison`
.

`linetype_summary_difference`
.

`color_interval_reference`
.

`color_interval_comparison`
.

`color_interval_difference`
.

`size_interval_reference`
.

`size_interval_comparison`
.

`size_interval_difference`
.

`alpha_interval_reference`
.

`alpha_interval_comparison`
.

`alpha_interval_difference`
.

`alpha_error_reference`
.

`alpha_error_comparison`
.

`alpha_error_difference`
.

`fill_error_reference`
.

`fill_error_comparison`
.

`fill_error_difference`
.

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$es_mean_difference`	a table
`results$es_smd`	a table
`results$es_mean_ratio`	a table
`results$es_median_difference`	a table
`results$es_median_ratio`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an array of images

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamovimetamdiff` *Meta-Analysis: Difference in Means*

Description

Meta-Analysis: Difference in Means

Usage

```
jamovimetamdiff(
  switch = "from_raw",
  data,
  comparison_means,
  comparison_sds,
  comparison_ns,
  reference_means,
  reference_sds,
  reference_ns,
  r,
  labels,
  moderator,
  d,
  dcomparison_ns,
  dreference_ns,
  dr,
  dlabels,
  dmoderator,
  conf_level = 95,
  effect_label = "My effect",
  reported_effect_size = "mean_difference",
  assume_equal_variance = TRUE,
  random_effects = "random_effects",
  include_PIs = FALSE,
  show_details = FALSE,
  es_plot_width = "600",
  es_plot_height = "750",
  size_base = "2",
  size_multiplier = "3",
  axis.text.y = "14",
  report_CIs = FALSE,
  meta_diamond_height = ".25",
  xlab = "auto",
  xmin = "auto",
  xmax = "auto",
  xbreaks = "auto",
  mark_zero = TRUE,
  axis.text.x = "14",
```

```
axis.title.x = "15",
dlab = "auto",
dmin = "auto",
dmax = "auto",
dbreaks = "auto",
shape_raw_reference = "square filled",
shape_raw_comparison = "square filled",
shape_summary_difference = "triangle filled",
shape_raw_unused = "square filled",
color_raw_reference = "#008DF9",
color_raw_comparison = "#009F81",
color_raw_unused = "gray65",
color_summary_reference = "#008DF9",
color_summary_comparison = "#009F81",
color_summary_unused = "gray75",
color_summary_difference = "black",
color_summary_overall = "black",
fill_raw_reference = "#008DF9",
fill_raw_comparison = "#009F81",
fill_raw_unused = "gray65",
fill_summary_reference = "#008DF9",
fill_summary_comparison = "#009F81",
fill_summary_unused = "gray75",
fill_summary_difference = "black",
fill_summary_overall = "black",
alpha_raw_reference = "1",
alpha_raw_comparison = "1",
alpha_raw_unused = "1",
alpha_summary_reference = "1",
alpha_summary_comparison = "1",
alpha_summary_unused = "1",
alpha_summary_difference = "1",
alpha_summary_overall = "1",
linetype_raw_reference = "solid",
linetype_raw_comparison = "solid",
linetype_summary_difference = "solid",
linetype_raw_unused = "solid",
color_interval_reference = "black",
color_interval_comparison = "black",
color_interval_difference = "black",
color_interval_unused = "black",
size_interval_reference = ".50",
size_interval_comparison = ".50",
size_interval_difference = ".50",
size_interval_unused = ".50",
alpha_interval_reference = "1",
alpha_interval_comparison = "1",
alpha_interval_difference = "1",
```

```
  alpha_interval_unused = "1"
)
```

Arguments

switch	.
data	.
comparison_means	.
comparison_sds	.
comparison_ns	.
reference_means	.
reference_sds	.
reference_ns	.
r	.
labels	.
moderator	.
d	.
dcomparison_ns	.
dreference_ns	.
dr	.
dlabels	.
dmoderator	.
conf_level	.
effect_label	.
reported_effect_size	.
assume_equal_variance	.
random_effects	.
include_PIs	.
show_details	.
es_plot_width	.
es_plot_height	.
size_base	.
size_multiplier	.
axis.text.y	.
report_CIs	.

`meta_diamond_height`	.
`xlab`	.
`xmin`	.
`xmax`	.
`xbreaks`	.
`mark_zero`	.
`axis.text.x`	.
`axis.title.x`	.
`dlab`	.
`dmin`	.
`dmax`	.
`dbreaks`	.
`shape_raw_reference`	.
`shape_raw_comparison`	.
`shape_summary_difference`	.
`shape_raw_unused`	.
`color_raw_reference`	.
`color_raw_comparison`	.
`color_raw_unused`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_unused`	.
`color_summary_difference`	.
`color_summary_overall`	.
`fill_raw_reference`	.
`fill_raw_comparison`	.
`fill_raw_unused`	.
`fill_summary_reference`	.

`fill_summary_comparison`	.
`fill_summary_unused`	.
`fill_summary_difference`	.
`fill_summary_overall`	.
`alpha_raw_reference`	.
`alpha_raw_comparison`	.
`alpha_raw_unused`	.
`alpha_summary_reference`	.
`alpha_summary_comparison`	.
`alpha_summary_unused`	.
`alpha_summary_difference`	.
`alpha_summary_overall`	.
`linetype_raw_reference`	.
`linetype_raw_comparison`	.
`linetype_summary_difference`	.
`linetype_raw_unused`	.
`color_interval_reference`	.
`color_interval_comparison`	.
`color_interval_difference`	.
`color_interval_unused`	.
`size_interval_reference`	.
`size_interval_comparison`	.
`size_interval_difference`	.
`size_interval_unused`	.

`alpha_interval_reference`	.
`alpha_interval_comparison`	.
`alpha_interval_difference`	.
`alpha_interval_unused`	.

Value

A results object containing:

`results$debug`	a html
`results$help`	a html
`results$raw_data`	a table
`results$es_meta`	a table
`results$es_heterogeneity`	a table
`results$es_meta_difference`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$raw_data$asDF`

`as.data.frame(results$raw_data)`

`jamovimetamean` *Meta-Analysis: Means*

Description

Meta-Analysis: Means

Usage

```
jamovimetamean(
  switch = "from_raw",
  data,
  means,
  sds,
  ns,
  labels,
  moderator,
  ds,
  dns,
```

```
  dlabels,
  dmoderator,
  conf_level = 95,
  effect_label = "My effect",
  reference_mean = "",
  reported_effect_size = "mean_difference",
  random_effects = "random_effects",
  include_PIs = FALSE,
  show_details = FALSE,
  es_plot_width = "600",
  es_plot_height = "750",
  size_base = "2",
  size_multiplier = "3",
  axis.text.y = "14",
  report_CIs = FALSE,
  meta_diamond_height = ".25",
  xlab = "auto",
  xmin = "auto",
  xmax = "auto",
  xbreaks = "auto",
  mark_zero = TRUE,
  axis.text.x = "14",
  axis.title.x = "15",
  dlab = "auto",
  dmin = "auto",
  dmax = "auto",
  dbreaks = "auto",
  shape_raw_reference = "square filled",
  shape_raw_comparison = "square filled",
  shape_summary_difference = "triangle filled",
  shape_raw_unused = "square filled",
  color_raw_reference = "#008DF9",
  color_raw_comparison = "#009F81",
  color_raw_unused = "gray65",
  color_summary_reference = "#008DF9",
  color_summary_comparison = "#009F81",
  color_summary_unused = "gray75",
  color_summary_difference = "black",
  color_summary_overall = "black",
  fill_raw_reference = "#008DF9",
  fill_raw_comparison = "#009F81",
  fill_raw_unused = "gray65",
  fill_summary_reference = "#008DF9",
  fill_summary_comparison = "#009F81",
  fill_summary_unused = "gray75",
  fill_summary_difference = "black",
  fill_summary_overall = "black",
  alpha_raw_reference = "1",
```

```
  alpha_raw_comparison = "1",
  alpha_raw_unused = "1",
  alpha_summary_reference = "1",
  alpha_summary_comparison = "1",
  alpha_summary_unused = "1",
  alpha_summary_difference = "1",
  alpha_summary_overall = "1",
  linetype_raw_reference = "solid",
  linetype_raw_comparison = "solid",
  linetype_summary_difference = "solid",
  linetype_raw_unused = "solid",
  color_interval_reference = "black",
  color_interval_comparison = "black",
  color_interval_difference = "black",
  color_interval_unused = "black",
  size_interval_reference = ".50",
  size_interval_comparison = ".50",
  size_interval_difference = ".50",
  size_interval_unused = ".50",
  alpha_interval_reference = "1",
  alpha_interval_comparison = "1",
  alpha_interval_difference = "1",
  alpha_interval_unused = "1"
)
```

Arguments

`switch`	.
`data`	.
`means`	.
`sds`	.
`ns`	.
`labels`	.
`moderator`	.
`ds`	.
`dns`	.
`dlabels`	.
`dmoderator`	.
`conf_level`	.
`effect_label`	.
`reference_mean`	.
`reported_effect_size`	.
`random_effects`	.

`include_PIs`	.
`show_details`	.
`es_plot_width`	.
`es_plot_height`	.
`size_base`	.
`size_multiplier`	.
`axis.text.y`	.
`report_CIs`	.
`meta_diamond_height`	.
`xlab`	.
`xmin`	.
`xmax`	.
`xbreaks`	.
`mark_zero`	.
`axis.text.x`	.
`axis.title.x`	.
`dlab`	.
`dmin`	.
`dmax`	.
`dbreaks`	.
`shape_raw_reference`	.
`shape_raw_comparison`	.
`shape_summary_difference`	.
`shape_raw_unused`	.
`color_raw_reference`	.
`color_raw_comparison`	.
`color_raw_unused`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_unused`	.

`color_summary_difference`

.

`color_summary_overall`

.

`fill_raw_reference`

.

`fill_raw_comparison`

.

`fill_raw_unused`

.

`fill_summary_reference`

.

`fill_summary_comparison`

.

`fill_summary_unused`

.

`fill_summary_difference`

.

`fill_summary_overall`

.

`alpha_raw_reference`

.

`alpha_raw_comparison`

.

`alpha_raw_unused`

.

`alpha_summary_reference`

.

`alpha_summary_comparison`

.

`alpha_summary_unused`

.

`alpha_summary_difference`

.

`alpha_summary_overall`

.

`linetype_raw_reference`

.

`linetype_raw_comparison`

.

`linetype_summary_difference`

.

`linetype_raw_unused`

.

`color_interval_reference`

.

`color_interval_comparison`

.

`color_interval_difference`
.

`color_interval_unused`
.

`size_interval_reference`
.

`size_interval_comparison`
.

`size_interval_difference`
.

`size_interval_unused`
.

`alpha_interval_reference`
.

`alpha_interval_comparison`
.

`alpha_interval_difference`
.

`alpha_interval_unused`
.

Value

A results object containing:

`results$debug`	a html
`results$help`	a html
`results$raw_data`	a table
`results$es_meta`	a table
`results$es_heterogeneity`	a table
`results$es_meta_difference`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$raw_data$asDF`

`as.data.frame(results$raw_data)`

`jamovimetapdiff` *Meta-Analysis: Difference in Proportions*

Description

Meta-Analysis: Difference in Proportions

Usage

```
jamovimetapdiff(
  data,
  reference_cases,
  reference_ns,
  comparison_cases,
  comparison_ns,
  labels,
  moderator,
  effect_label = "My effect",
  reported_effect_size = "RD",
  conf_level = 95,
  random_effects = "random_effects",
  include_PIs = FALSE,
  show_details = FALSE,
  es_plot_width = "600",
  es_plot_height = "750",
  size_base = "2",
  size_multiplier = "3",
  axis.text.y = "14",
  report_CIs = FALSE,
  meta_diamond_height = ".25",
  xlab = "auto",
  xmin = "auto",
  xmax = "auto",
  xbreaks = "auto",
  mark_zero = FALSE,
  axis.text.x = "14",
  axis.title.x = "15",
  dlab = "auto",
  dmin = "auto",
  dmax = "auto",
  dbreaks = "auto",
  shape_raw_reference = "square filled",
  shape_raw_comparison = "square filled",
  shape_summary_difference = "triangle filled",
  shape_raw_unused = "square filled",
  color_raw_reference = "#008DF9",
  color_raw_comparison = "#009F81",
```

```
  color_raw_unused = "gray65",
  color_summary_reference = "#008DF9",
  color_summary_comparison = "#009F81",
  color_summary_unused = "gray75",
  color_summary_difference = "black",
  color_summary_overall = "black",
  fill_raw_reference = "#008DF9",
  fill_raw_comparison = "#009F81",
  fill_raw_unused = "gray65",
  fill_summary_reference = "#008DF9",
  fill_summary_comparison = "#009F81",
  fill_summary_unused = "gray75",
  fill_summary_difference = "black",
  fill_summary_overall = "black",
  alpha_raw_reference = "1",
  alpha_raw_comparison = "1",
  alpha_raw_unused = "1",
  alpha_summary_reference = "1",
  alpha_summary_comparison = "1",
  alpha_summary_unused = "1",
  alpha_summary_difference = "1",
  alpha_summary_overall = "1",
  linetype_raw_reference = "solid",
  linetype_raw_comparison = "solid",
  linetype_summary_difference = "solid",
  linetype_raw_unused = "solid",
  color_interval_reference = "black",
  color_interval_comparison = "black",
  color_interval_difference = "black",
  color_interval_unused = "black",
  size_interval_reference = ".50",
  size_interval_comparison = ".50",
  size_interval_difference = ".50",
  size_interval_unused = ".50",
  alpha_interval_reference = "1",
  alpha_interval_comparison = "1",
  alpha_interval_difference = "1",
  alpha_interval_unused = "1"
)
```

Arguments

`data`	.
`reference_cases`	.
`reference_ns`	.
`comparison_cases`	.

`comparison_ns`	.
`labels`	.
`moderator`	.
`effect_label`	.
`reported_effect_size`	.
`conf_level`	.
`random_effects`	.
`include_PIs`	.
`show_details`	.
`es_plot_width`	.
`es_plot_height`	.
`size_base`	.
`size_multiplier`	.
`axis.text.y`	.
`report_CIs`	.
`meta_diamond_height`	.
`xlab`	.
`xmin`	.
`xmax`	.
`xbreaks`	.
`mark_zero`	.
`axis.text.x`	.
`axis.title.x`	.
`dlab`	.
`dmin`	.
`dmax`	.
`dbreaks`	.
`shape_raw_reference`	.
`shape_raw_comparison`	.
`shape_summary_difference`	.
`shape_raw_unused`	.
`color_raw_reference`	.

`color_raw_comparison`	.
`color_raw_unused`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_unused`	.
`color_summary_difference`	.
`color_summary_overall`	.
`fill_raw_reference`	.
`fill_raw_comparison`	.
`fill_raw_unused`	.
`fill_summary_reference`	.
`fill_summary_comparison`	.
`fill_summary_unused`	.
`fill_summary_difference`	.
`fill_summary_overall`	.
`alpha_raw_reference`	.
`alpha_raw_comparison`	.
`alpha_raw_unused`	.
`alpha_summary_reference`	.
`alpha_summary_comparison`	.
`alpha_summary_unused`	.
`alpha_summary_difference`	.
`alpha_summary_overall`	.
`linetype_raw_reference`	.

`linetype_raw_comparison`	.
`linetype_summary_difference`	.
`linetype_raw_unused`	.
`color_interval_reference`	.
`color_interval_comparison`	.
`color_interval_difference`	.
`color_interval_unused`	.
`size_interval_reference`	.
`size_interval_comparison`	.
`size_interval_difference`	.
`size_interval_unused`	.
`alpha_interval_reference`	.
`alpha_interval_comparison`	.
`alpha_interval_difference`	.
`alpha_interval_unused`	.

Value

A results object containing:

`results$debug`	a html
`results$help`	a html
`results$raw_data`	a table
`results$es_meta`	a table
`results$es_heterogeneity`	a table
`results$es_meta_difference`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$raw_data$asDF`

`as.data.frame(results$raw_data)`

`jamovimetaproportion` *Meta-Analysis: Proportions*

Description

Meta-Analysis: Proportions

Usage

```
jamovimetaproportion(
  data,
  cases,
  ns,
  labels,
  moderator,
  effect_label = "My effect",
  conf_level = 95,
  random_effects = "random_effects",
  include_PIs = FALSE,
  show_details = FALSE,
  es_plot_width = "600",
  es_plot_height = "750",
  size_base = "2",
  size_multiplier = "3",
  axis.text.y = "14",
  report_CIs = FALSE,
  meta_diamond_height = ".25",
  xlab = "auto",
  xmin = "auto",
  xmax = "auto",
  xbreaks = "auto",
  mark_zero = FALSE,
  axis.text.x = "14",
  axis.title.x = "15",
  dlab = "auto",
  dmin = "auto",
  dmax = "auto",
  dbreaks = "auto",
  shape_raw_reference = "square filled",
  shape_raw_comparison = "square filled",
  shape_summary_difference = "triangle filled",
  shape_raw_unused = "square filled",
  color_raw_reference = "#008DF9",
  color_raw_comparison = "#009F81",
  color_raw_unused = "gray65",
  color_summary_reference = "#008DF9",
  color_summary_comparison = "#009F81",
```

```
  color_summary_unused = "gray75",
  color_summary_difference = "black",
  color_summary_overall = "black",
  fill_raw_reference = "#008DF9",
  fill_raw_comparison = "#009F81",
  fill_raw_unused = "gray65",
  fill_summary_reference = "#008DF9",
  fill_summary_comparison = "#009F81",
  fill_summary_unused = "gray75",
  fill_summary_difference = "black",
  fill_summary_overall = "black",
  alpha_raw_reference = "1",
  alpha_raw_comparison = "1",
  alpha_raw_unused = "1",
  alpha_summary_reference = "1",
  alpha_summary_comparison = "1",
  alpha_summary_unused = "1",
  alpha_summary_difference = "1",
  alpha_summary_overall = "1",
  linetype_raw_reference = "solid",
  linetype_raw_comparison = "solid",
  linetype_summary_difference = "solid",
  linetype_raw_unused = "solid",
  color_interval_reference = "black",
  color_interval_comparison = "black",
  color_interval_difference = "black",
  color_interval_unused = "black",
  size_interval_reference = ".50",
  size_interval_comparison = ".50",
  size_interval_difference = ".50",
  size_interval_unused = ".50",
  alpha_interval_reference = "1",
  alpha_interval_comparison = "1",
  alpha_interval_difference = "1",
  alpha_interval_unused = "1"
)
```

Arguments

`data`	.
`cases`	.
`ns`	.
`labels`	.
`moderator`	.
`effect_label`	.
`conf_level`	.
`random_effects`	.

`include_PIs`	.
`show_details`	.
`es_plot_width`	.
`es_plot_height`	.
`size_base`	.
`size_multiplier`	.
`axis.text.y`	.
`report_CIs`	.
`meta_diamond_height`	.
`xlab`	.
`xmin`	.
`xmax`	.
`xbreaks`	.
`mark_zero`	.
`axis.text.x`	.
`axis.title.x`	.
`dlab`	.
`dmin`	.
`dmax`	.
`dbreaks`	.
`shape_raw_reference`	.
`shape_raw_comparison`	.
`shape_summary_difference`	.
`shape_raw_unused`	.
`color_raw_reference`	.
`color_raw_comparison`	.
`color_raw_unused`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_unused`	.

`color_summary_difference`
.

`color_summary_overall`
.

`fill_raw_reference`
.

`fill_raw_comparison`
.

`fill_raw_unused`
.

`fill_summary_reference`
.

`fill_summary_comparison`
.

`fill_summary_unused`
.

`fill_summary_difference`
.

`fill_summary_overall`
.

`alpha_raw_reference`
.

`alpha_raw_comparison`
.

`alpha_raw_unused`
.

`alpha_summary_reference`
.

`alpha_summary_comparison`
.

`alpha_summary_unused`
.

`alpha_summary_difference`
.

`alpha_summary_overall`
.

`linetype_raw_reference`
.

`linetype_raw_comparison`
.

`linetype_summary_difference`
.

`linetype_raw_unused`
.

`color_interval_reference`
.

`color_interval_comparison`
.

`color_interval_difference`
.

`color_interval_unused`
.

`size_interval_reference`
.

`size_interval_comparison`
.

`size_interval_difference`
.

`size_interval_unused`
.

`alpha_interval_reference`
.

`alpha_interval_comparison`
.

`alpha_interval_difference`
.

`alpha_interval_unused`
.

Value

A results object containing:

`results$debug`	a html
`results$help`	a html
`results$raw_data`	a table
`results$es_meta`	a table
`results$es_heterogeneity`	a table
`results$es_meta_difference`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$raw_data$asDF`

`as.data.frame(results$raw_data)`

`jamovimetar` *Meta-Analysis: Correlations*

Description

Meta-Analysis: Correlations

Usage

```
jamovimetar(
  data,
  rs,
  ns,
  labels,
  moderator,
  effect_label = "My effect",
  conf_level = 95,
  random_effects = "random_effects",
  include_PIs = FALSE,
  show_details = FALSE,
  es_plot_width = "600",
  es_plot_height = "750",
  size_base = "2",
  size_multiplier = "3",
  axis.text.y = "14",
  report_CIs = FALSE,
  meta_diamond_height = ".25",
  xlab = "auto",
  xmin = "-1",
  xmax = "1",
  xbreaks = "auto",
  mark_zero = TRUE,
  axis.text.x = "14",
  axis.title.x = "15",
  dlab = "auto",
  dmin = "auto",
  dmax = "auto",
  dbreaks = "auto",
  shape_raw_reference = "square filled",
  shape_raw_comparison = "square filled",
  shape_summary_difference = "triangle filled",
  shape_raw_unused = "square filled",
  color_raw_reference = "#008DF9",
  color_raw_comparison = "#009F81",
  color_raw_unused = "gray65",
  color_summary_reference = "#008DF9",
  color_summary_comparison = "#009F81",
```

```
  color_summary_unused = "gray75",
  color_summary_difference = "black",
  color_summary_overall = "black",
  fill_raw_reference = "#008DF9",
  fill_raw_comparison = "#009F81",
  fill_raw_unused = "gray65",
  fill_summary_reference = "#008DF9",
  fill_summary_comparison = "#009F81",
  fill_summary_unused = "gray75",
  fill_summary_difference = "black",
  fill_summary_overall = "black",
  alpha_raw_reference = "1",
  alpha_raw_comparison = "1",
  alpha_raw_unused = "1",
  alpha_summary_reference = "1",
  alpha_summary_comparison = "1",
  alpha_summary_unused = "1",
  alpha_summary_difference = "1",
  alpha_summary_overall = "1",
  linetype_raw_reference = "solid",
  linetype_raw_comparison = "solid",
  linetype_summary_difference = "solid",
  linetype_raw_unused = "solid",
  color_interval_reference = "black",
  color_interval_comparison = "black",
  color_interval_difference = "black",
  color_interval_unused = "black",
  size_interval_reference = ".50",
  size_interval_comparison = ".50",
  size_interval_difference = ".50",
  size_interval_unused = ".50",
  alpha_interval_reference = "1",
  alpha_interval_comparison = "1",
  alpha_interval_difference = "1",
  alpha_interval_unused = "1"
)
```

Arguments

`data`	.
`rs`	.
`ns`	.
`labels`	.
`moderator`	.
`effect_label`	.
`conf_level`	.
`random_effects`	.

`include_PIs`	.
`show_details`	.
`es_plot_width`	.
`es_plot_height`	.
`size_base`	.
`size_multiplier`	.
`axis.text.y`	.
`report_CIs`	.
`meta_diamond_height`	.
`xlab`	.
`xmin`	.
`xmax`	.
`xbreaks`	.
`mark_zero`	.
`axis.text.x`	.
`axis.title.x`	.
`dlab`	.
`dmin`	.
`dmax`	.
`dbreaks`	.
`shape_raw_reference`	.
`shape_raw_comparison`	.
`shape_summary_difference`	.
`shape_raw_unused`	.
`color_raw_reference`	.
`color_raw_comparison`	.
`color_raw_unused`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_unused`	.

`color_summary_difference`

.

`color_summary_overall`

.

`fill_raw_reference`

.

`fill_raw_comparison`

.

`fill_raw_unused`

.

`fill_summary_reference`

.

`fill_summary_comparison`

.

`fill_summary_unused`

.

`fill_summary_difference`

.

`fill_summary_overall`

.

`alpha_raw_reference`

.

`alpha_raw_comparison`

.

`alpha_raw_unused`

.

`alpha_summary_reference`

.

`alpha_summary_comparison`

.

`alpha_summary_unused`

.

`alpha_summary_difference`

.

`alpha_summary_overall`

.

`linetype_raw_reference`

.

`linetype_raw_comparison`

.

`linetype_summary_difference`

.

`linetype_raw_unused`

.

`color_interval_reference`

.

`color_interval_comparison`

.

`color_interval_difference`
.

`color_interval_unused`
.

`size_interval_reference`
.

`size_interval_comparison`
.

`size_interval_difference`
.

`size_interval_unused`
.

`alpha_interval_reference`
.

`alpha_interval_comparison`
.

`alpha_interval_difference`
.

`alpha_interval_unused`
.

Value

A results object containing:

`results$debug`	a html
`results$help`	a html
`results$raw_data`	a table
`results$es_meta`	a table
`results$es_heterogeneity`	a table
`results$es_meta_difference`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$raw_data$asDF`

`as.data.frame(results$raw_data)`

`jamovipdiffpaired` *Proportions: Paired*

Description

Proportions: Paired

Usage

```
jamovipdiffpaired(
  switch = "from_raw",
  data,
  reference_measure,
  comparison_measure,
  cases_consistent = " ",
  cases_inconsistent = " ",
  not_cases_consistent = " ",
  not_cases_inconsistent = " ",
  case_label = "Sick",
  not_case_label = "Well",
  comparison_measure_name = "Post-test",
  reference_measure_name = "Pre-test",
  conf_level = 95,
  show_details = FALSE,
  evaluate_hypotheses = FALSE,
  null_value = "0",
  null_boundary = "0",
  alpha = 0.05,
  null_color = "#A40122",
  es_plot_width = "400",
  es_plot_height = "450",
  ymin = "auto",
  ymax = "auto",
  ybreaks = "auto",
  ylab = "auto",
  xlab = "auto",
  axis.text.y = "14",
  axis.title.y = "15",
  axis.text.x = "14",
  axis.title.x = "15",
  simple_contrast_labels = TRUE,
  error_layout = "none",
  difference_axis_breaks = "auto",
  shape_summary_reference = "circle filled",
  shape_summary_comparison = "circle filled",
  shape_summary_difference = "triangle filled",
  color_summary_reference = "#008DF9",
```

```
  color_summary_comparison = "#009F81",
  color_summary_difference = "black",
  fill_summary_reference = "#008DF9",
  fill_summary_comparison = "#009F81",
  fill_summary_difference = "black",
  size_summary_reference = "4",
  size_summary_comparison = "4",
  size_summary_difference = "4",
  alpha_summary_reference = "1",
  alpha_summary_comparison = "1",
  alpha_summary_difference = "1",
  linetype_summary_reference = "solid",
  linetype_summary_comparison = "solid",
  linetype_summary_difference = "solid"
)
```

Arguments

`switch`	.
`data`	.
`reference_measure`	.
`comparison_measure`	.
`cases_consistent`	.
`cases_inconsistent`	.
`not_cases_consistent`	.
`not_cases_inconsistent`	.
`case_label`	.
`not_case_label`	.
`comparison_measure_name`	.
`reference_measure_name`	.
`conf_level`	.
`show_details`	.
`evaluate_hypotheses`	.
`null_value`	.
`null_boundary`	.
`alpha`	.
`null_color`	.

`es_plot_width`	.
`es_plot_height`	.
`ymin`	.
`ymax`	.
`ybreaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`simple_contrast_labels`	.
`error_layout`	.
`difference_axis_breaks`	.
`shape_summary_reference`	.
`shape_summary_comparison`	.
`shape_summary_difference`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_difference`	.
`fill_summary_reference`	.
`fill_summary_comparison`	.
`fill_summary_difference`	.
`size_summary_reference`	.
`size_summary_comparison`	.
`size_summary_difference`	.
`alpha_summary_reference`	.
`alpha_summary_comparison`	.

`alpha_summary_difference`
.

`linetype_summary_reference`
.

`linetype_summary_comparison`
.

`linetype_summary_difference`
.

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$es_proportion_difference`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamovipdifftwo` *Proportions: Two Groups*

Description

Proportions: Two Groups

Usage

```
jamovipdifftwo(
  switch = "from_raw",
  data,
  outcome_variable,
  grouping_variable,
  comparison_cases = " ",
  comparison_not_cases = " ",
  reference_cases = " ",
  reference_not_cases = " ",
  case_label = "Sick",
```

```
not_case_label = "Well",
grouping_variable_level1 = "Treated",
grouping_variable_level2 = "Control",
outcome_variable_name = "Outcome variable",
grouping_variable_name = "Grouping variable",
count_NA = FALSE,
show_ratio = FALSE,
show_chi_square = FALSE,
chi_table_option = "both",
show_phi = FALSE,
conf_level = 95,
show_details = FALSE,
evaluate_hypotheses = FALSE,
null_value = "0",
null_boundary = "0",
alpha = 0.05,
null_color = "#A40122",
es_plot_width = "400",
es_plot_height = "450",
ymin = "auto",
ymax = "auto",
ybreaks = "auto",
ylab = "auto",
xlab = "auto",
axis.text.y = "14",
axis.title.y = "15",
axis.text.x = "14",
axis.title.x = "15",
simple_contrast_labels = TRUE,
error_layout = "none",
difference_axis_breaks = "auto",
shape_summary_reference = "circle filled",
shape_summary_comparison = "circle filled",
shape_summary_difference = "triangle filled",
color_summary_reference = "#008DF9",
color_summary_comparison = "#009F81",
color_summary_difference = "black",
fill_summary_reference = "#008DF9",
fill_summary_comparison = "#009F81",
fill_summary_difference = "black",
size_summary_reference = "4",
size_summary_comparison = "4",
size_summary_difference = "4",
alpha_summary_reference = "1",
alpha_summary_comparison = "1",
alpha_summary_difference = "1",
linetype_summary_reference = "solid",
linetype_summary_comparison = "solid",
```

```
  linetype_summary_difference = "solid"
)
```

Arguments

`switch`	.
`data`	.
`outcome_variable`	.
`grouping_variable`	.
`comparison_cases`	.
`comparison_not_cases`	.
`reference_cases`	.
`reference_not_cases`	.
`case_label`	.
`not_case_label`	.
`grouping_variable_level1`	.
`grouping_variable_level2`	.
`outcome_variable_name`	.
`grouping_variable_name`	.
`count_NA`	.
`show_ratio`	.
`show_chi_square`	.
`chi_table_option`	.
`show_phi`	.
`conf_level`	.
`show_details`	.
`evaluate_hypotheses`	.
`null_value`	.
`null_boundary`	.
`alpha`	.
`null_color`	.

`es_plot_width`	.
`es_plot_height`	.
`ymin`	.
`ymax`	.
`ybreaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`simple_contrast_labels`	.
`error_layout`	.
`difference_axis_breaks`	.
`shape_summary_reference`	.
`shape_summary_comparison`	.
`shape_summary_difference`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_difference`	.
`fill_summary_reference`	.
`fill_summary_comparison`	.
`fill_summary_difference`	.
`size_summary_reference`	.
`size_summary_comparison`	.
`size_summary_difference`	.
`alpha_summary_reference`	.
`alpha_summary_comparison`	.

`alpha_summary_difference`	.
`linetype_summary_reference`	.
`linetype_summary_comparison`	.
`linetype_summary_difference`	.

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$es_proportion_difference`	a table
`results$contingency_table`	a table
`results$es_phi`	a table
`results$es_odds_ratio`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an array of images

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamoviproportion` *Proportions: Single Group*

Description

Proportions: Single Group

Usage

```
jamoviproportion(
  switch = "from_raw",
  data,
  outcome_variable,
  cases = " ",
  not_cases = " ",
  case_label = "Affected",
```

```
  not_case_label = "Not Affected",
  outcome_variable_name = "My outcome variable",
  count_NA = FALSE,
  conf_level = 95,
  show_details = FALSE,
  plot_possible = FALSE,
  evaluate_hypotheses = FALSE,
  null_value = "0",
  null_boundary = "0",
  alpha = 0.05,
  null_color = "#A40122",
  es_plot_width = "300",
  es_plot_height = "400",
  ymin = "0",
  ymax = "1",
  breaks = "auto",
  ylab = "auto",
  xlab = "auto",
  axis.text.y = "14",
  axis.title.y = "15",
  axis.text.x = "14",
  axis.title.x = "15",
  error_layout = "none",
  shape_summary = "circle filled",
  color_summary = "#008DF9",
  fill_summary = "#008DF9",
  size_summary = "4",
  alpha_summary = "1",
  linetype_summary = "solid"
)
```

Arguments

`switch`	.
`data`	.
`outcome_variable`	.
`cases`	.
`not_cases`	.
`case_label`	.
`not_case_label`	.
`outcome_variable_name`	.
`count_NA`	.
`conf_level`	.
`show_details`	.

`plot_possible`	.
`evaluate_hypotheses`	.
`null_value`	.
`null_boundary`	.
`alpha`	.
`null_color`	.
`es_plot_width`	.
`es_plot_height`	.
`ymin`	.
`ymax`	.
`breaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`error_layout`	.
`shape_summary`	.
`color_summary`	.
`fill_summary`	.
`size_summary`	.
`alpha_summary`	.
`linetype_summary`	.

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$magnitude_plot_warnings`	a html
`results$estimation_plots`	an array of images

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`jamovirdifftwo` *Correlations: Two Groups*

Description

Correlations: Two Groups

Usage

```
jamovirdifftwo(
  switch = "from_raw",
  data,
  x,
  y,
  grouping_variable,
  comparison_r = " ",
  comparison_n = " ",
  reference_r = " ",
  reference_n = " ",
  x_variable_name = "X variable",
  y_variable_name = "Y variable",
  comparison_level_name = "Comparison level",
  reference_level_name = "Reference level",
  grouping_variable_name = "Grouping variable",
  conf_level = 95,
  show_details = FALSE,
  show_line = TRUE,
  show_line_CI = TRUE,
  evaluate_hypotheses = FALSE,
  null_value = "0",
  null_boundary = "0",
  alpha = 0.05,
  null_color = "#A40122",
  es_plot_width = "600",
  es_plot_height = "400",
  sp_plot_width = "650",
  sp_plot_height = "650",
  ymin = "auto",
  ymax = "auto",
  ybreaks = "auto",
  ylab = "auto",
  xlab = "auto",
  axis.text.y = "14",
  axis.title.y = "15",
  axis.text.x = "14",
  axis.title.x = "15",
  sp_ymin = "auto",
```

```
  sp_ymax = "auto",
  sp_ybreaks = "auto",
  sp_xmin = "auto",
  sp_xmax = "auto",
  sp_xbreaks = "auto",
  sp_ylab = "auto",
  sp_xlab = "auto",
  sp_axis.text.y = "14",
  sp_axis.title.y = "15",
  sp_axis.text.x = "14",
  sp_axis.title.x = "15",
  difference_axis_breaks = "auto",
  shape_summary_reference = "circle filled",
  shape_summary_comparison = "circle filled",
  shape_summary_difference = "triangle filled",
  color_summary_reference = "#008DF9",
  color_summary_comparison = "#009F81",
  color_summary_difference = "black",
  fill_summary_reference = "#008DF9",
  fill_summary_comparison = "#009F81",
  fill_summary_difference = "black",
  size_summary_reference = "4",
  size_summary_comparison = "4",
  size_summary_difference = "4",
  alpha_summary_reference = "1",
  alpha_summary_comparison = "1",
  alpha_summary_difference = "1",
  linetype_summary_reference = "solid",
  linetype_summary_comparison = "solid",
  linetype_summary_difference = "solid",
  sp_shape_raw_reference = "circle filled",
  sp_shape_raw_comparison = "circle filled",
  sp_shape_raw_unused = "circle filled",
  sp_color_raw_reference = "black",
  sp_color_raw_comparison = "black",
  sp_color_raw_unused = "black",
  sp_fill_raw_reference = "#008DF9",
  sp_fill_raw_comparison = "#009F81",
  sp_fill_raw_unused = "NA",
  sp_size_raw_reference = "3",
  sp_size_raw_comparison = "3",
  sp_size_raw_unused = "2",
  sp_alpha_raw_reference = ".25",
  sp_alpha_raw_comparison = ".25",
  sp_alpha_raw_unused = ".25",
  sp_linetype_summary_reference = "solid",
  sp_linetype_summary_comparison = "solid",
  sp_color_summary_reference = "#008DF9",
```

```
  sp_color_summary_comparison = "#009F81”,
  sp_size_summary_reference = ”2”,
  sp_size_summary_comparison = ”2”,
  sp_alpha_summary_reference = ”.25”,
  sp_alpha_summary_comparison = ”.25”
)
```

Arguments

switch	.
data	.
x	.
y	.
grouping_variable	.
comparison_r	.
comparison_n	.
reference_r	.
reference_n	.
x_variable_name	.
y_variable_name	.
comparison_level_name	.
reference_level_name	.
grouping_variable_name	.
conf_level	.
show_details	.
show_line	.
show_line_CI	.
evaluate_hypotheses	.
null_value	.
null_boundary	.
alpha	.
null_color	.
es_plot_width	.
es_plot_height	.
sp_plot_width	.

`sp_plot_height`	.
`ymin`	.
`ymax`	.
`ybreaks`	.
`ylab`	.
`xlab`	.
`axis.text.y`	.
`axis.title.y`	.
`axis.text.x`	.
`axis.title.x`	.
`sp_ymin`	.
`sp_ymax`	.
`sp_ybreaks`	.
`sp_xmin`	.
`sp_xmax`	.
`sp_xbreaks`	.
`sp_ylab`	.
`sp_xlab`	.
`sp_axis.text.y`	.
`sp_axis.title.y`	.
`sp_axis.text.x`	.
`sp_axis.title.x`	.
`difference_axis_breaks`	.
`shape_summary_reference`	.
`shape_summary_comparison`	.
`shape_summary_difference`	.
`color_summary_reference`	.
`color_summary_comparison`	.
`color_summary_difference`	.
`fill_summary_reference`	.
`fill_summary_comparison`	.

`fill_summary_difference`
.

`size_summary_reference`
.

`size_summary_comparison`
.

`size_summary_difference`
.

`alpha_summary_reference`
.

`alpha_summary_comparison`
.

`alpha_summary_difference`
.

`linetype_summary_reference`
.

`linetype_summary_comparison`
.

`linetype_summary_difference`
.

`sp_shape_raw_reference`
.

`sp_shape_raw_comparison`
.

`sp_shape_raw_unused`
.

`sp_color_raw_reference`
.

`sp_color_raw_comparison`
.

`sp_color_raw_unused`
.

`sp_fill_raw_reference`
.

`sp_fill_raw_comparison`
.

`sp_fill_raw_unused`
.

`sp_size_raw_reference`
.

`sp_size_raw_comparison`
.

`sp_size_raw_unused`
.

`sp_alpha_raw_reference`
.

`sp_alpha_raw_comparison`
.

`sp_alpha_raw_unused`
: .

`sp_linetype_summary_reference`
: .

`sp_linetype_summary_comparison`
: .

`sp_color_summary_reference`
: .

`sp_color_summary_comparison`
: .

`sp_size_summary_reference`
: .

`sp_size_summary_comparison`
: .

`sp_alpha_summary_reference`
: .

`sp_alpha_summary_comparison`
: .

Value

A results object containing:

`results$debug`	a preformatted
`results$help`	a html
`results$overview`	a table
`results$es_r`	a table
`results$es_r_difference`	a table
`results$point_null`	a table
`results$interval_null`	a table
`results$scatter_plot_warnings`	a html
`results$scatter_plots`	an image
`results$estimation_plot_warnings`	a html
`results$estimation_plots`	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

`results$overview$asDF`

`as.data.frame(results$overview)`

`meta_any` *Estimate any meta effect.*

Description

`meta_any` is suitable for synthesizing any effect size across multiple studies. You must provide the effect size for each study and the predicted sampling variance for each study.

Usage

```
meta_any(
  data,
  yi,
  vi,
  labels = NULL,
  moderator = NULL,
  contrast = NULL,
  effect_label = "My effect",
  effect_size_name = "Effect size",
  moderator_variable_name = "My moderator",
  random_effects = TRUE,
  conf_level = 0.95
)
```

Arguments

`data`	A data frame or tibble with columns
`yi`	Name a column in data containing the effect size for each study
`vi`	Name of a column in data containing the expected sampling variance for each study
`labels`	Name of a column in data containing a label for each study
`moderator`	Optional name of a column in data containing a factor as a categorical moderator
`contrast`	Optional vector specifying a contrast analysis for the categorical moderator. Only define if a moderator is defined; vector length should match number of levels in the moderator
`effect_label`	Optional human-friendly name for the effect being synthesized; defaults to 'My effect'
`effect_size_name`	Optional human-friendly name of the effect size being synthesized; defaults to 'Effect size'
`moderator_variable_name`	Optional human-friendly name of the moderator, if defined; If not passed but a moderator is defined, will be set to the quoted name of the moderator column or 'My moderator'

`random_effects`	Use TRUE to obtain a random effect meta-analysis (usually recommended); FALSE for fixed effect.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.

Details

#' Once you generate an estimate with this function, you can visualize it with `plot_meta()`.

The meta-analytic effect size, confidence interval and heterogeneity estimates all come from `metafor::rma()`.

The diamond ratio and its confidence interval come from `CI_diamond_ratio()`.

Value

An esci-estimate object; a list of data frames and properties. Returned tables include:

- **es_meta** - A data frame of meta-analytic effect sizes. If a moderator was defined, there is an additional row for each level of the moderator.
  - *effect_label* - Study label
  - *effect_size* - Effect size
  - *LL* - Lower bound of conf_level% confidence interval
  - *UL* - Upper bound of conf_level% confidence interval
  - *SE* - Expected standard error
  - *k* - Number of studies
  - *diamond_ratio* - ratio of random to fixed effects meta-analytic effect sizes
  - *diamond_ratio_LL* - lower bound of conf_level% confidence interval for diamond ratio
  - *diamond_ratio_UL* - upper bound of conf_level% confidence interval for diamond ratio
  - *I2* - I2 measure of heterogeneity
  - *I2_LL* - Lower bound of conf_level% confidence interval for I2
  - *I2_UL* - upper bound of conf_level% confidence interval for I2
  - *PI_LL* - lower bound of conf_level% of prediction interval
  - *PI_UL* - upper bound of conf_level% of prediction interval
  - *p* - p value for the meta-analytic effect size, based on null of exactly 0
  - *width - width of the effect-size confidence interval
  - *FE_effect_size* - effect size of the fixed-effects model (regardless of if fixed effects was selected
  - *RE_effect_size* - effect size of the random-effects model (regardless of if random effects was selected
  - *FE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
  - *RE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
- **es_heterogeneity** - A data frame of of heterogeneity values and conf_level% CIs for the meta-analytic effect size. If a moderator was defined also reports heterogeneity estimates for each level of the moderator.
  - *effect_label* - study label

  - *moderator_variable_name* - if moderator passed, gives name of the moderator
  - *moderator_level* - 'Overall' and each level of moderator, if passed
  - *measure* - Name of the measure of heterogeneity
  - *estimate* - Value of the heterogeneity estimate
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
- **raw_data** - A data from with one row for each study that was passed
  - *label* - study label
  - *effect_size* - effect size
  - *weight* - study weight in the meta analysis
  - *sample_variance* - expected level of sampling variation
  - *SE* - expected standard error
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
  - *mean* - used to calculate study p value; this is the d value entered for the study
  - *sd* - use to calculate study p value; set to 1 for each study
  - *n* - study sample size
  - *p* - p value for the study, based on null of exactly 0

Examples

```
#' # Data set -- see Introduction to the New Statistics, 2nd edition
data("data_mccabemichael_brain")

# Fixed effect, 95% CI
esizes <- esci::meta_mean(
  data = esci::data_mccabemichael_brain,
  means = "M Brain",
  sds = "s Brain",
  ns = "n Brain",
  labels = "Study name",
  random_effects = FALSE
)$raw_data

estimate <- esci::meta_any(
  data = esizes,
  yi = effect_size,
  vi = sample_variance,
  labels = label,
  effect_size_name = "Mean",
  random_effects = FALSE
)

myplot_forest <- esci::plot_meta(estimate)
```

`meta_d1` *Estimate a meta-analytic Cohen's d1 across multiple studies*

Description

`meta_d1` is suitable for synthesizing across multiple single-group studies with a continuous outcome variable, but where the outcome is not measured on the same scale in all studies

Usage

```
meta_d1(
  data,
  ds,
  ns,
  labels = NULL,
  moderator = NULL,
  contrast = NULL,
  effect_label = "My effect",
  random_effects = TRUE,
  conf_level = 0.95
)
```

Arguments

`data`	A data frame or tibble
`ds`	Set of bias-adjusted cohen's d1 values, 1 for each study
`ns`	Set of sample sizes, positive integers, 1 for each study
`labels`	Optional set of labels, 1 for each study
`moderator`	Optional factor as a categorical moderator; should have k > 2 per group
`contrast`	Optional vector specifying a contrast between moderator levels
`effect_label`	Optional character providing a human-friendly label for the effect
`random_effects`	Boolean; TRUE for a random effects model; otherwise fixed effects
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.

Details

Once you generate an estimate with this function, you can visualize it with `plot_meta()`.

Each study's effect size should be expressed as Cohen's d1: (mean - reference) / sd.

And the d1 values should all be corrected for bias. The function `CI_smd_one()` can assist with converting raw data from each study to d1_unbiased.

The meta-analytic effect size, confidence interval and heterogeneity estimates all come from `metafor::rma()`.

The diamond ratio and its confidence interval come from `CI_diamond_ratio()`.

Value

An esci-estimate object; a list of data frames and properties. Returned tables include:

- **es_meta** - A data frame of meta-analytic effect sizes. If a moderator was defined, there is an additional row for each level of the moderator.
  - *effect_label* - Study label
  - *effect_size* - Effect size
  - *LL* - Lower bound of conf_level% confidence interval
  - *UL* - Upper bound of conf_level% confidence interval
  - *SE* - Expected standard error
  - *k* - Number of studies
  - *diamond_ratio* - ratio of random to fixed effects meta-analytic effect sizes
  - *diamond_ratio_LL* - lower bound of conf_level% confidence interval for diamond ratio
  - *diamond_ratio_UL* - upper bound of conf_level% confidence interval for diamond ratio
  - *I2* - I2 measure of heterogeneity
  - *I2_LL* - Lower bound of conf_level% confidence interval for I2
  - *I2_UL* - upper bound of conf_level% confidence interval for I2
  - *PI_LL* - lower bound of conf_level% of prediction interval
  - *PI_UL* - upper bound of conf_level% of prediction interval
  - *p* - p value for the meta-analytic effect size, based on null of exactly 0
  - *width - width of the effect-size confidence interval
  - *FE_effect_size* - effect size of the fixed-effects model (regardless of if fixed effects was selected
  - *RE_effect_size* - effect size of the random-effects model (regardless of if random effects was selected
  - *FE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
  - *RE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
- **es_heterogeneity** - A data frame of of heterogeneity values and conf_level% CIs for the meta-analytic effect size. If a moderator was defined also reports heterogeneity estimates for each level of the moderator.
  - *effect_label* - study label
  - *moderator_variable_name* - if moderator passed, gives name of the moderator
  - *moderator_level* - 'Overall' and each level of moderator, if passed
  - *measure* - Name of the measure of heterogeneity
  - *estimate* - Value of the heterogeneity estimate
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
- **raw_data** - A data from with one row for each study that was passed
  - *label* - study label
  - *effect_size* - effect size
  - *weight* - study weight in the meta analysis

  - *sample_variance* - expected level of sampling variation
  - *SE* - expected standard error
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
  - *mean* - used to calculate study p value; this is the d value entered for the study
  - *sd* - use to calculate study p value; set to 1 for each study
  - *n* - study sample size
  - *p* - p value for the study, based on null of exactly 0

Examples

```
# example code
  original_7 <- data.frame(
study_name = c(
  "Aden (1993)" ,
  "Buggs (1995)" ,
  "Crazed (1999)" ,
  "Dudley (2003)" ,
  "Evers (2005)" ,
  "Fox (2009)",
  "Mine (2011)"
),
rt_mean = c(
  454 ,
  317 ,
  430 ,
  525 ,
  479 ,
  387,
  531
),
rt_sd = c(
  142 ,
  158 ,
  137 ,
  260 ,
  144 ,
  165,
  233
),
rt_n = c(
  24 ,
  7 ,
  20 ,
  8 ,
 14 ,
 13,
  18
),
subset = as.factor(
  c(
```

```
    "90s",
   "90s",
    "90s",
    "00s",
    "00s",
    "00s",
    "00s"
  )
),
d1_unbiased = c(
  3.091587,
  1.742751,
  3.012857,
  1.793487,
  3.130074,
  2.195209,
  2.17667
)
)


# Fixed effect, 95% CI
estimate <- esci::meta_d1(
  original_7,
  d1_unbiased,
  rt_n,
  study_name,
  random_effects = FALSE
)

# Forest plot
myplot_forest <- esci::plot_meta(estimate)


# Add a moderator
estimate_moderator <- esci::meta_d1(
  data = original_7,
  ds = d1_unbiased,
  ns = rt_n,
  moderator = subset,
  labels = study_name,
  random_effects = FALSE
)

# Forest plot
myplot_forest_moderator <- esci::plot_meta(estimate_moderator)
```

`meta_d2` *Estimate meta-analytic standardized mean difference across multiple two group studies (all paired, all independent, or a mix).*

Description

`meta_d2` is suitable for synthesizing across multiple two-group studies (paired or independent) with a continuous outcome measure but where not all studies are measured on the same scale, and instead the magnitude of difference for each study is expressed as d_s or d_avg.

Usage

```
meta_d2(
  data,
  ds,
  comparison_ns,
  reference_ns,
  r = NULL,
  labels = NULL,
  moderator = NULL,
  contrast = NULL,
  effect_label = "My effect",
  assume_equal_variance = FALSE,
  random_effects = TRUE,
  conf_level = 0.95
)
```

Arguments

`data`	A data frame or tibble
`ds`	Set of bias-adjusted cohen's d_s or d_avg values, 1 for each study
`comparison_ns`	Set of comparison_group sample sizes, positive integers, 1 for each study
`reference_ns`	Set of reference_groups sample sizes, positive integers, 1 for each study
`r`	optional correlation between measures for w-s studies, NA otherwise
`labels`	Optional set of labels, 1 for each study
`moderator`	Optional factor as a categorical moderator; should have k > 2 per group
`contrast`	Optional vector specifying a contrast between moderator levels
`effect_label`	Optional character providing a human-friendly label for the effect
`assume_equal_variance`	Defaults to FALSE
`random_effects`	Boolean; TRUE for a random effects model; otherwise fixed effects
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.

Details

Once you generate an estimate with this function, you can visualize it with `plot_meta()`.

Each study's effect size should be expressed as: Cohen's d_s: (comparison_mean - reference_mean) / sd_pooled or Cohen_'s d_avg: (comparison_mean - reference_mean) / sd_avg

To enter d_s, set assume_equal_variance to TRUE To enter d_avg, set assume_equal_variance to FALSE

And the d values should all be corrected for bias. The function `CI_smd_ind_contrast()` can assist with converting raw data from each study to d_s or d_avg with bias correction. It also has more details on calculation of these forms of d and their CIs.

The meta-analytic effect size, confidence interval and heterogeneity estimates all come from `metafor::rma()`.

The diamond ratio and its confidence interval come from `CI_diamond_ratio()`.

Value

An esci-estimate object; a list of data frames and properties. Returned tables include:

- **es_meta** - A data frame of meta-analytic effect sizes. If a moderator was defined, there is an additional row for each level of the moderator.
    - *effect_label* - Study label
    - *effect_size* - Effect size
    - *LL* - Lower bound of conf_level% confidence interval
    - *UL* - Upper bound of conf_level% confidence interval
    - *SE* - Expected standard error
    - *k* - Number of studies
    - *diamond_ratio* - ratio of random to fixed effects meta-analytic effect sizes
    - *diamond_ratio_LL* - lower bound of conf_level% confidence interval for diamond ratio
    - *diamond_ratio_UL* - upper bound of conf_level% confidence interval for diamond ratio
    - *I2* - I2 measure of heterogeneity
    - *I2_LL* - Lower bound of conf_level% confidence interval for I2
    - *I2_UL* - upper bound of conf_level% confidence interval for I2
    - *PI_LL* - lower bound of conf_level% of prediction interval
    - *PI_UL* - upper bound of conf_level% of prediction interval
    - *p* - p value for the meta-analytic effect size, based on null of exactly 0
    - *width - width of the effect-size confidence interval
    - *FE_effect_size* - effect size of the fixed-effects model (regardless of if fixed effects was selected
    - *RE_effect_size* - effect size of the random-effects model (regardless of if random effects was selected
    - *FE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
    - *RE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio

- **es_heterogeneity** - A data frame of of heterogeneity values and conf_level% CIs for the meta-analytic effect size. If a moderator was defined also reports heterogeneity estimates for each level of the moderator.
  - *effect_label* - study label
  - *moderator_variable_name* - if moderator passed, gives name of the moderator
  - *moderator_level* - 'Overall' and each level of moderator, if passed
  - *measure* - Name of the measure of heterogeneity
  - *estimate* - Value of the heterogeneity estimate
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
- **raw_data** - A data from with one row for each study that was passed
  - *label* - study label
  - *effect_size* - effect size
  - *weight* - study weight in the meta analysis
  - *sample_variance* - expected level of sampling variation
  - *SE* - expected standard error
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
  - *mean* - used to calculate study p value; this is the d value entered for the study
  - *sd* - use to calculate study p value; set to 1 for each study
  - *n* - study sample size
  - *p* - p value for the study, based on null of exactly 0

Examples

```
# Data set -- see Introduction to the New Statistics, 1st edition
data("data_damischrcj")

# Meta-analysis, random effects, assuming equal variance, no moderator
estimate <- esci::meta_d2(
  data = esci::data_damischrcj,
  ds = "Cohen's d unbiased",
  comparison_ns = "n Control",
  reference_ns = "n Lucky",
  labels = Study,
  assume_equal_variance = TRUE,
  random_effects = TRUE
)

# Forest plot
myplot_forest <- esci::plot_meta(estimate)


# Add a categorical moderator
estimate_moderator <- esci::meta_d2(
  data = esci::data_damischrcj,
  ds = "Cohen's d unbiased",
```

```
  comparison_ns = "n Control",
  reference_ns = "n Lucky",
  labels = "Study",
  moderator = "Research Group",
  assume_equal_variance = TRUE,
  random_effects = TRUE
)

# Forest plot
myplot_forest_moderator <- esci::plot_meta(estimate_moderator)
```

`meta_mdiff_two` *Estimate meta-analytic difference in means across multiple two-group studies.*

Description

`meta_mdiff_two` is suitable for synthesizing across multiple two-group studies (paired or independent) with a continuous outcome measure. It takes in raw data from each study. If all studies used the same measurement scale, a meta-analytic raw-score difference can be returned. If studies used different scales, a standardized mean difference can be returned. Studies can be all paired, all independent, or a mix. Equal variance can be assumed, or not. If standardized mean difference is the output, it is d_s when equal variance is assumed and d_avg when equal variance is not assumed.

Usage

```
meta_mdiff_two(
  data,
  comparison_means,
  comparison_sds,
  comparison_ns,
  reference_means,
  reference_sds,
  reference_ns,
  r = NULL,
  labels = NULL,
  moderator = NULL,
  contrast = NULL,
  effect_label = "My effect",
  reported_effect_size = c("mean_difference", "smd_unbiased", "smd"),
  assume_equal_variance = FALSE,
  random_effects = TRUE,
  conf_level = 0.95
)
```

Arguments

`data`	A data frame or tibble
`comparison_means`	Set of comparison_group means, 1 per study
`comparison_sds`	Set of comparison_group standard deviations, 1 per study, all > 0
`comparison_ns`	Set of comparison_group sample sizes, positive integers, 1 for each study
`reference_means`	Set of reference_group means, 1 per study
`reference_sds`	Set of comparison_group standard deviations, 1 per study, all > 0
`reference_ns`	Set of reference_group sample sizes, positive integers, 1 for each study
`r`	Optional correlation between measures for w-s studies, NA otherwise
`labels`	An optional collection of study labels
`moderator`	An optional factor to analyze as a categorical moderator, must have k > 2 per groups
`contrast`	An optional contrast to estimate between moderator levels; express as a vector of contrast weights with 1 weight per moderator level.
`effect_label`	Optional character giving a human-friendly name of the effect being synthesized
`reported_effect_size`	Character specifying effect size to return: Must be one of 'mean_difference', 'smd_unbiased' (to return an unbiased Cohen's d_s or d_avg) or 'smd' (to return d_s or d_avg without correction for bias). Defaults to mean_difference.
`assume_equal_variance`	Defaults to FALSE
`random_effects`	TRUE for random effect model; FALSE for fixed effects
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.

Details

Once you generate an estimate with this function, you can visualize it with `plot_meta()`.

The meta-analytic effect size, confidence interval and heterogeneity estimates all come from `metafor::rma()`.

The diamond ratio and its confidence interval come from `CI_diamond_ratio()`.

If reported_effect_size is smd_unbiased or smd the conversion to Cohen's d is handled by `CI_smd_ind_contrast()`.

Value

An esci-estimate object; a list of data frames and properties. Returned tables include:

- **es_meta** - A data frame of meta-analytic effect sizes. If a moderator was defined, there is an additional row for each level of the moderator.
  - *effect_label* - Study label
  - *effect_size* - Effect size
  - *LL* - Lower bound of conf_level% confidence interval

  - *UL* - Upper bound of conf_level% confidence interval
  - *SE* - Expected standard error
  - *k* - Number of studies
  - *diamond_ratio* - ratio of random to fixed effects meta-analytic effect sizes
  - *diamond_ratio_LL* - lower bound of conf_level% confidence interval for diamond ratio
  - *diamond_ratio_UL* - upper bound of conf_level% confidence interval for diamond ratio
  - *I2* - I2 measure of heterogeneity
  - *I2_LL* - Lower bound of conf_level% confidence interval for I2
  - *I2_UL* - upper bound of conf_level% confidence interval for I2
  - *PI_LL* - lower bound of conf_level% of prediction interval
  - *PI_UL* - upper bound of conf_level% of prediction interval
  - *p* - p value for the meta-analytic effect size, based on null of exactly 0
  - *width - width of the effect-size confidence interval
  - *FE_effect_size* - effect size of the fixed-effects model (regardless of if fixed effects was selected
  - *RE_effect_size* - effect size of the random-effects model (regardless of if random effects was selected
  - *FE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
  - *RE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
- **es_heterogeneity** - A data frame of of heterogeneity values and conf_level% CIs for the meta-analytic effect size. If a moderator was defined also reports heterogeneity estimates for each level of the moderator.
  - *effect_label* - study label
  - *moderator_variable_name* - if moderator passed, gives name of the moderator
  - *moderator_level* - 'Overall' and each level of moderator, if passed
  - *measure* - Name of the measure of heterogeneity
  - *estimate* - Value of the heterogeneity estimate
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
- **raw_data** - A data from with one row for each study that was passed
  - *label* - study label
  - *effect_size* - effect size
  - *weight* - study weight in the meta analysis
  - *sample_variance* - expected level of sampling variation
  - *SE* - expected standard error
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
  - *mean* - used to calculate study p value; this is the d value entered for the study
  - *sd* - use to calculate study p value; set to 1 for each study
  - *n* - study sample size
  - *p* - p value for the study, based on null of exactly 0

Examples

```
# Data set -- see Introduction to the New Statistics, 2nd edition
data("data_mccabemichael_brain")

# Meta-analysis: random effects, no moderator
estimate <- esci::meta_mdiff_two(
  data = esci::data_mccabemichael_brain,
  comparison_means = "M Brain",
  comparison_sds = "s Brain",
  comparison_ns = "n Brain",
  reference_means = "M No Brain",
  reference_sds = "s No Brain",
  reference_ns = "n No Brain",
  labels = "Study name",
  effect_label = "Brain Photo Rating - No Brain Photo Rating",
  assume_equal_variance = TRUE,
  random_effects = TRUE
)
# Forest plot
myplot_forest <- esci::plot_meta(estimate)


# Meta-analysis: random effects, moderator
estimate_moderator <- esci::meta_mdiff_two(
  data = esci::data_mccabemichael_brain,
  comparison_means = "M Brain",
  comparison_sds = "s Brain",
  comparison_ns = "n Brain",
  reference_means = "M No Brain",
  reference_sds = "s No Brain",
  reference_ns = "n No Brain",
  labels = "Study name",
  moderator = "Research group",
  effect_label = "Brain Photo Rating - No Brain Photo Rating",
  assume_equal_variance = TRUE,
  random_effects = TRUE
)
# Forest plot
myplot_forest_moderator <- esci::plot_meta(estimate_moderator)


# Meta-analysis: random effects, moderator, output d_s
estimate_moderator_d <- esci::meta_mdiff_two(
  data = esci::data_mccabemichael_brain,
  comparison_means = "M Brain",
  comparison_sds = "s Brain",
  comparison_ns = "n Brain",
  reference_means = "M No Brain",
  reference_sds = "s No Brain",
  reference_ns = "n No Brain",
  labels = "Study name",
  moderator = "Research group",
```

```
  effect_label = "Brain Photo Rating - No Brain Photo Rating",
  assume_equal_variance = TRUE,
  random_effects = TRUE
)
# Forest plot
myplot_forest_moderator_d <- esci::plot_meta(estimate_moderator_d)
```

`meta_mean` *Estimate a meta-analytic mean across multiple single-group studies.*

Description

`meta_mean` is suitable for synthesizing across multiple single-group studies with a continuous outcome variable when all studies are measured on the same scale.

Usage

```
meta_mean(
  data,
  means,
  sds,
  ns,
  labels = NULL,
  moderator = NULL,
  contrast = NULL,
  effect_label = "My effect",
  reference_mean = 0,
  reported_effect_size = c("mean_difference", "smd_unbiased", "smd"),
  random_effects = TRUE,
  conf_level = 0.95
)
```

Arguments

`data`	A dataframe or tibble
`means`	A collection of study means, 1 per study
`sds`	A collection of study standard deviations, 1 per study, all >0
`ns`	A collection of sample sizes, 1 per study, all integers > 2
`labels`	An optional collection of study labels
`moderator`	An optional factor to analyze as a categorical moderator, must have k > 2 per groups
`contrast`	An optional contrast to estimate between moderator levels; express as a vector of contrast weights with 1 weight per moderator level.
`effect_label`	Optional character giving a human-friendly name of the effect being synthesized

`reference_mean`	Optional reference mean, defaults to 0
`reported_effect_size`	Character specifying effect size to return; Must be one of 'mean_difference', 'smd_unbiased' (to return an unbiased Cohen's d1) or 'smd' (to return Cohen's d1 without correction for bias)
`random_effects`	TRUE for random effect model; FALSE for fixed effects
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.

Details

The meta-analytic effect size, confidence interval and heterogeneity estimates all come from `metafor::rma()`.

The diamond ratio and its confidence interval come from `CI_diamond_ratio()`.

If reported_effect_size is smd_unbiased or smd the conversion to d1 is handled by `CI_smd_one()`.

Value

An esci-estimate object; a list of data frames and properties. Returned tables include:

- **es_meta** - A data frame of meta-analytic effect sizes. If a moderator was defined, there is an additional row for each level of the moderator.
  - *effect_label* - Study label
  - *effect_size* - Effect size
  - *LL* - Lower bound of conf_level% confidence interval
  - *UL* - Upper bound of conf_level% confidence interval
  - *SE* - Expected standard error
  - *k* - Number of studies
  - *diamond_ratio* - ratio of random to fixed effects meta-analytic effect sizes
  - *diamond_ratio_LL* - lower bound of conf_level% confidence interval for diamond ratio
  - *diamond_ratio_UL* - upper bound of conf_level% confidence interval for diamond ratio
  - *I2* - I2 measure of heterogeneity
  - *I2_LL* - Lower bound of conf_level% confidence interval for I2
  - *I2_UL* - upper bound of conf_level% confidence interval for I2
  - *PI_LL* - lower bound of conf_level% of prediction interval
  - *PI_UL* - upper bound of conf_level% of prediction interval
  - *p* - p value for the meta-analytic effect size, based on null of exactly 0
  - *width - width of the effect-size confidence interval
  - *FE_effect_size* - effect size of the fixed-effects model (regardless of if fixed effects was selected
  - *RE_effect_size* - effect size of the random-effects model (regardless of if random effects was selected
  - *FE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
  - *RE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio

- **es_heterogeneity** - A data frame of of heterogeneity values and conf_level% CIs for the meta-analytic effect size. If a moderator was defined also reports heterogeneity estimates for each level of the moderator.
  - *effect_label* - study label
  - *moderator_variable_name* - if moderator passed, gives name of the moderator
  - *moderator_level* - 'Overall' and each level of moderator, if passed
  - *measure* - Name of the measure of heterogeneity
  - *estimate* - Value of the heterogeneity estimate
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
- **raw_data** - A data from with one row for each study that was passed
  - *label* - study label
  - *effect_size* - effect size
  - *weight* - study weight in the meta analysis
  - *sample_variance* - expected level of sampling variation
  - *SE* - expected standard error
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
  - *mean* - used to calculate study p value; this is the d value entered for the study
  - *sd* - use to calculate study p value; set to 1 for each study
  - *n* - study sample size
  - *p* - p value for the study, based on null of exactly 0

Examples

```
# Data set -- see Introduction to the New Statistics, 2nd edition
data("data_mccabemichael_brain")

# Fixed effect, 95% CI
estimate <- esci::meta_mean(
  data = esci::data_mccabemichael_brain,
  means = "M Brain",
  sds = "s Brain",
  ns = "n Brain",
  labels = "Study name",
  random_effects = FALSE
)

myplot_forest <- esci::plot_meta(estimate)


# Add a moderator, report cohen's d1
estimate_moderator_d <- esci::meta_mean(
  data = esci::data_mccabemichael_brain,
  means = "M Brain",
  sds = "s Brain",
  ns = "n Brain",
```

```
  labels = "Study name",
  moderator = "Research group",
  reported_effect_size = "smd_unbiased",
  random_effects = FALSE
)

# Forest plot
myplot_forest_moderator_d <- esci::plot_meta(estimate_moderator_d)
```

`meta_pdiff_two` *Estimate meta-analytic difference in proportions over multiple studies with two independent groups and a categorical outcome variable.*

Description

`meta_pdiff_two` is suitable for synthesizing across multiple two-group studies with a categorical outcome variable. It takes as input the the number of cases/events in the comparison and reference groups as well as the total number of samples in the comparison and reference groups.

Usage

```
meta_pdiff_two(
  data,
  comparison_cases,
  comparison_ns,
  reference_cases,
  reference_ns,
  labels = NULL,
  moderator = NULL,
  contrast = NULL,
  effect_label = "My effect",
  reported_effect_size = c("RD", "RR", "OR", "AS", "PETO"),
  random_effects = TRUE,
  conf_level = 0.95
)
```

Arguments

`data`	A dataframe or tibble
`comparison_cases`	A collection of case/event counts for the comparison groups, 1 per study, all integers >= 0
`comparison_ns`	A collection of sample sizes for the comparison groups, 1 per study, all integers > 2

`reference_cases`	A collection of case/event counts for the reference groups, 1 per study, all integers >= 0
`reference_ns`	A collection of sample sizes for the reference groups, 1 per study, all integers > 2
`labels`	An optional collection of study labels
`moderator`	An optional factor to analyze as a categorical moderator, must have k > 2 per groups
`contrast`	An optional contrast to estimate between moderator levels; express as a vector of contrast weights with 1 weight per moderator level.
`effect_label`	Optional character giving a human-friendly name of the effect being synthesized
`reported_effect_size`	Character specifying effect size to return: Must be one of 'RD' (risk difference, default), 'RR' (log risk ratio), 'OR' (log odds ratio), 'AS' (arcsine square root transformed risk difference), or 'PETO' (log odds ratio estimated using Peto's method). See `metafor::escalc()` for details.
`random_effects`	TRUE for random effect model; FALSE for fixed effects
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.

Details

Once you generate an estimate with this function, you can visualize it with `plot_meta()`.

The meta-analytic effect size, confidence interval and heterogeneity estimates all come from `metafor::rma()`.

The conversion of events into suitable effect sizes is handled by `metafor::escalc()`

Value

An esci-estimate object; a list of data frames and properties. Returned tables include:

- **es_meta** - A data frame of meta-analytic effect sizes. If a moderator was defined, there is an additional row for each level of the moderator.
  - *effect_label* - Study label
  - *effect_size* - Effect size
  - *LL* - Lower bound of conf_level% confidence interval
  - *UL* - Upper bound of conf_level% confidence interval
  - *SE* - Expected standard error
  - *k* - Number of studies
  - *diamond_ratio* - ratio of random to fixed effects meta-analytic effect sizes
  - *diamond_ratio_LL* - lower bound of conf_level% confidence interval for diamond ratio
  - *diamond_ratio_UL* - upper bound of conf_level% confidence interval for diamond ratio
  - *I2* - I2 measure of heterogeneity
  - *I2_LL* - Lower bound of conf_level% confidence interval for I2
  - *I2_UL* - upper bound of conf_level% confidence interval for I2

  - *PI_LL* - lower bound of conf_level% of prediction interval
  - *PI_UL* - upper bound of conf_level% of prediction interval
  - *p* - p value for the meta-analytic effect size, based on null of exactly 0
  - *width - width of the effect-size confidence interval
  - *FE_effect_size* - effect size of the fixed-effects model (regardless of if fixed effects was selected
  - *RE_effect_size* - effect size of the random-effects model (regardless of if random effects was selected
  - *FE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
  - *RE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
- **es_heterogeneity** - A data frame of of heterogeneity values and conf_level% CIs for the meta-analytic effect size. If a moderator was defined also reports heterogeneity estimates for each level of the moderator.
  - *effect_label* - study label
  - *moderator_variable_name* - if moderator passed, gives name of the moderator
  - *moderator_level* - 'Overall' and each level of moderator, if passed
  - *measure* - Name of the measure of heterogeneity
  - *estimate* - Value of the heterogeneity estimate
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
- **raw_data** - A data from with one row for each study that was passed
  - *label* - study label
  - *effect_size* - effect size
  - *weight* - study weight in the meta analysis
  - *sample_variance* - expected level of sampling variation
  - *SE* - expected standard error
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
  - *mean* - used to calculate study p value; this is the d value entered for the study
  - *sd* - use to calculate study p value; set to 1 for each study
  - *n* - study sample size
  - *p* - p value for the study, based on null of exactly 0

Examples

```
# Data set: Replications of power on egocentric behavior
esci_meta_pdiff_two <- data.frame(
  studies = c(
    "Online",
    "Original",
    "Online Pilot",
    "Exact replication"
  ),
```

```
    control_egocentric = c(
      33,
      4,
      4,
      7
    ),
    control_sample_size = c(
     101,
      33,
      10,
      53
    ),
    power_egocentric = c(
      48,
      8,
      4,
      11
    ),
    power_sample_size = c(
      105,
      24,
      12,
      56
    ),
    setting = as.factor(
      c(
        "Online",
       "In-Person",
        "Online",
        "In-Person"
      )
    )
  )

  # Meta-analysis, risk difference as effect size
  estimate <- esci::meta_pdiff_two(
    esci_meta_pdiff_two,
    power_egocentric,
    power_sample_size,
    control_egocentric,
    control_sample_size,
    studies,
    reported_effect_size = "RD"
  )

  # Forest plot
  myplot_forst <- esci::plot_meta(estimate)


  # Add a categorical moderator (setting)
  estimate_moderator <- esci::meta_pdiff_two(
    esci_meta_pdiff_two,
    power_egocentric,
```

```
  power_sample_size,
  control_egocentric,
  control_sample_size,
  studies,
  moderator = setting,
  reported_effect_size = "RD"
)

# Forest plot
myplot_forest_moderator <- esci::plot_meta(estimate_moderator)
```

`meta_proportion` *Estimate a meta-analytic proportion of outcomes over multiple studies with a categorical outcome variable.*

Description

`meta_proportion` is suitable for synthesizing across multiple studies with a categorical outcome variable. It takes as input the number of cases/events and the number of samples in each study.

Usage

```
meta_proportion(
  data,
  cases,
  ns,
  labels = NULL,
  moderator = NULL,
  contrast = NULL,
  effect_label = "My effect",
  random_effects = TRUE,
  conf_level = 0.95
)
```

Arguments

`data`	A dataframe or tibble
`cases`	A collection of cases/event counts, 1 per study, all integers, all > 0
`ns`	A collection of sample sizes, 1 per study, all integers > 2
`labels`	An optional collection of study labels
`moderator`	An optional factor to analyze as a categorical moderator, must have k > 2 per groups
`contrast`	An optional contrast to estimate between moderator levels; express as a vector of contrast weights with 1 weight per moderator level.

`effect_label`	Optional character giving a human-friendly name of the effect being synthesized
`random_effects`	TRUE for random effect model; FALSE for fixed effects
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.

Details

Once you generate an estimate with this function, you can visualize it with `plot_meta()`.

The meta-analytic effect size, confidence interval and heterogeneity estimates all come from `metafor::rma()`.

Value

An esci-estimate object; a list of data frames and properties. Returned tables include:

- **es_meta** - A data frame of meta-analytic effect sizes. If a moderator was defined, there is an additional row for each level of the moderator.
    - *effect_label* - Study label
    - *effect_size* - Effect size
    - *LL* - Lower bound of conf_level% confidence interval
    - *UL* - Upper bound of conf_level% confidence interval
    - *SE* - Expected standard error
    - *k* - Number of studies
    - *diamond_ratio* - ratio of random to fixed effects meta-analytic effect sizes
    - *diamond_ratio_LL* - lower bound of conf_level% confidence interval for diamond ratio
    - *diamond_ratio_UL* - upper bound of conf_level% confidence interval for diamond ratio
    - *I2* - I2 measure of heterogeneity
    - *I2_LL* - Lower bound of conf_level% confidence interval for I2
    - *I2_UL* - upper bound of conf_level% confidence interval for I2
    - *PI_LL* - lower bound of conf_level% of prediction interval
    - *PI_UL* - upper bound of conf_level% of prediction interval
    - *p* - p value for the meta-analytic effect size, based on null of exactly 0
    - *width - width of the effect-size confidence interval
    - *FE_effect_size* - effect size of the fixed-effects model (regardless of if fixed effects was selected
    - *RE_effect_size* - effect size of the random-effects model (regardless of if random effects was selected
    - *FE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
    - *RE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
- **es_heterogeneity** - A data frame of of heterogeneity values and conf_level% CIs for the meta-analytic effect size. If a moderator was defined also reports heterogeneity estimates for each level of the moderator.
    - *effect_label* - study label

  - *moderator_variable_name* - if moderator passed, gives name of the moderator
  - *moderator_level* - ’Overall’ and each level of moderator, if passed
  - *measure* - Name of the measure of heterogeneity
  - *estimate* - Value of the heterogeneity estimate
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
- **raw_data** - A data from with one row for each study that was passed
  - *label* - study label
  - *effect_size* - effect size
  - *weight* - study weight in the meta analysis
  - *sample_variance* - expected level of sampling variation
  - *SE* - expected standard error
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
  - *mean* - used to calculate study p value; this is the d value entered for the study
  - *sd* - use to calculate study p value; set to 1 for each study
  - *n* - study sample size
  - *p* - p value for the study, based on null of exactly 0

Examples

```
# Data set: Replications of power on egocentric behavior
esci_meta_pdiff_two <- data.frame(
  studies = c(
    "Online",
    "Original",
    "Online Pilot",
    "Exact replication"
  ),
  control_egocentric = c(
    33,
    4,
    4,
    7
  ),
  control_sample_size = c(
   101,
    33,
    10,
    53
  ),
  power_egocentric = c(
    48,
    8,
    4,
    11
  ),
  power_sample_size = c(
```

```
    105,
    24,
    12,
    56
  ),
  setting = as.factor(
    c(
      "Online",
     "In-Person",
      "Online",
      "In-Person"
    )
  )
)

# Meta-analysis, risk difference as effect size
estimate <- esci::meta_proportion(
  esci_meta_pdiff_two,
  power_egocentric,
  power_sample_size,
  studies
)

# Forest plot
myplot_forest <- esci::plot_meta(estimate)


# Meta-analysis, risk difference as effect size, moderator (setting)
estimate_moderator <- esci::meta_proportion(
  esci_meta_pdiff_two,
  power_egocentric,
  power_sample_size,
  studies,
  moderator = setting
)

# Forest plot
myplot_forest_moderator <- esci::plot_meta(estimate_moderator)
```

`meta_r` *Estimate meta-analytic Pearson's r across multiple studies with two continuous outcome variables.*

Description

`meta_r` is suitable for synthesizing across multiple studies that have measured a linear correlation (Pearson's r) from two continuous variables.

Usage

```
meta_r(
  data,
  rs,
  ns,
  labels = NULL,
  moderator = NULL,
  contrast = NULL,
  effect_label = "My effect",
  random_effects = TRUE,
  conf_level = 0.95
)
```

Arguments

`data`	A dataframe or tibble
`rs`	A collection of Pearson's r values, 1 per study, all between -1 and 1, inclusive
`ns`	A collection of study sample sizes, all integers > 2
`labels`	An optional collection of study labels
`moderator`	An optional factor to analyze as a categorical moderator, must have k > 2 per groups
`contrast`	An optional contrast to estimate between moderator levels; express as a vector of contrast weights with 1 weight per moderator level.
`effect_label`	Optional character giving a human-friendly name of the effect being synthesized
`random_effects`	TRUE for random effect model; FALSE for fixed effects
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.

Details

Once you generate an estimate with this function, you can visualize it with `plot_meta()`.

The meta-analytic effect size, confidence interval and heterogeneity estimates all come from `metafor::rma()`.

Value

An esci-estimate object; a list of data frames and properties. Returned tables include:

- **es_meta** - A data frame of meta-analytic effect sizes. If a moderator was defined, there is an additional row for each level of the moderator.
  - *effect_label* - Study label
  - *effect_size* - Effect size
  - *LL* - Lower bound of conf_level% confidence interval
  - *UL* - Upper bound of conf_level% confidence interval
  - *SE* - Expected standard error
  - *k* - Number of studies

  - *diamond_ratio* - ratio of random to fixed effects meta-analytic effect sizes
  - *diamond_ratio_LL* - lower bound of conf_level% confidence interval for diamond ratio
  - *diamond_ratio_UL* - upper bound of conf_level% confidence interval for diamond ratio
  - *I2* - I2 measure of heterogeneity
  - *I2_LL* - Lower bound of conf_level% confidence interval for I2
  - *I2_UL* - upper bound of conf_level% confidence interval for I2
  - *PI_LL* - lower bound of conf_level% of prediction interval
  - *PI_UL* - upper bound of conf_level% of prediction interval
  - *p* - p value for the meta-analytic effect size, based on null of exactly 0
  - *width - width of the effect-size confidence interval
  - *FE_effect_size* - effect size of the fixed-effects model (regardless of if fixed effects was selected
  - *RE_effect_size* - effect size of the random-effects model (regardless of if random effects was selected
  - *FE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio
  - *RE_CI_width* - width of the fixed-effects confidence interval, used to calculate diamond ratio

- **es_heterogeneity** - A data frame of of heterogeneity values and conf_level% CIs for the meta-analytic effect size. If a moderator was defined also reports heterogeneity estimates for each level of the moderator.
  - *effect_label* - study label
  - *moderator_variable_name* - if moderator passed, gives name of the moderator
  - *moderator_level* - 'Overall' and each level of moderator, if passed
  - *measure* - Name of the measure of heterogeneity
  - *estimate* - Value of the heterogeneity estimate
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
- **raw_data** - A data from with one row for each study that was passed
  - *label* - study label
  - *effect_size* - effect size
  - *weight* - study weight in the meta analysis
  - *sample_variance* - expected level of sampling variation
  - *SE* - expected standard error
  - *LL* - lower bound of conf_level% confidence interval
  - *UL* - upper bound of conf_level% confidence interval
  - *mean* - used to calculate study p value; this is the d value entered for the study
  - *sd* - use to calculate study p value; set to 1 for each study
  - *n* - study sample size
  - *p* - p value for the study, based on null of exactly 0

Examples

```
# Data: See Introduction to the New Statistics, first edition
esci_single_r <- data.frame(
  studies = c(
    'Violin, viola' ,
    'Strings' ,
    'Piano' ,
    'Piano' ,
    'Piano' ,
    'Piano' ,
    'Piano' ,
    'Piano' ,
    'Piano' ,
    'All' ,
    'Piano' ,
    'Piano' ,
    'Band' ,
    'Music majors' ,
    'Music majors' ,
    'All'
  ),
  rvalues = c(
    .67,
    .51,
    .4,
    .46,
    .47,
    .228,
    -.224,
    .104,
    .322,
    .231,
    .67,
    .41,
    .34,
    .31,
    .54,
    .583
  ),
  sample_size = c(
    109,
    55,
    19,
    30,
    19,
    52,
    24,
    52,
    16,
    97,
    57,
    107,
```

```
    178,
    64,
    19,
    135
  ),
  subsets = as.factor(
    c(
      'Strings' ,
      'Strings' ,
      'Piano' ,
      'Piano' ,
      'Piano' ,
      'Piano' ,
      'Piano' ,
      'Piano' ,
      'Piano' ,
      'Piano' ,
      'Piano' ,
      'Piano' ,
      'Strings' ,
      'Strings' ,
      'Strings' ,
      'Strings'
    )
 )
)

# Meta-analysis, random effects, no moderator
estimate <- esci::meta_r(
  esci_single_r,
  rvalues,
  sample_size,
  studies,
  random_effects = TRUE
)

# Forest plot
myplot_forest <- esci::plot_meta(estimate)


# Meta-analysis, random effects, moderator (subsets)
estimate_moderator <- esci::meta_r(
  esci_single_r,
  rvalues,
  sample_size,
  studies,
  subsets,
  random_effects = TRUE
)

# Forest plot
myplot_forest_moderator <- esci::plot_meta(estimate_moderator)
```

`overview` *Calculates descriptive statistics for a continuous variable*

Description

This function calculates basic descriptive statistics for a numerical variable. It can calculate an overall summary, or broken down by the levels of a grouping variable. Inputs can be summary data, vectors, or a data frame.

Usage

```
overview(
  data = NULL,
  outcome_variable = NULL,
  grouping_variable = NULL,
  means = NULL,
  sds = NULL,
  ns = NULL,
  grouping_variable_levels = NULL,
  outcome_variable_name = "My Outcome Variable",
  grouping_variable_name = NULL,
  conf_level = 0.95,
  assume_equal_variance = FALSE
)
```

Arguments

`data`	• for raw data, a data frame or tibble
`outcome_variable`	• for raw data, either a vector containing numerical data or the name of a data-frame column containing a factor
`grouping_variable`	• optional; for raw data either a vector containing a factor or the name of a data frame column containing a factor
`means`	For summary data - A vector of 1 or more numerical means
`sds`	For summary data - A vector of standard deviations, same length as means
`ns`	For summary data - A vector of sample sizes, same length as means
`grouping_variable_levels`	For summary data - An optional vector of group labels, same length as means. If not passed, auto-generated.
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My Outcome Variable'. Ignored if a data-frame is passed, this argument is ignored.

`grouping_variable_name`	Optional friendly name for the grouping variable. If a data frame is passed, this argument is ignored.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`assume_equal_variance`	Defaults to FALSE

Details

If equal variance is not assumed, each group is treated independently. In that case, the estimated mean, CI, and SE is from `statpsych::ci.mean1()`, and the estimated median, CI, and SE is from `statpsych::ci.median1()`. If equal variance is assumed, each group CI is calculated as with respect to all group data, using `statpsych::ci.lc.mean.bs()` and `statpsych::ci.lc.median.bs()`

Value

Returns a table of descriptive statistics

- **overview**
  - *outcome_variable_name* -
  - *grouping_variable_name* -
  - *grouping_variable_level* -
  - *mean* -
  - *mean_LL* -
  - *mean_UL* -
  - *median* -
  - *median_LL* -
  - *median_UL* -
  - *sd* -
  - *min* -
  - *max* -
  - *q1* -
  - *q3* -
  - *n* -
  - *missing* -
  - *df* -
  - *mean_SE* -
  - *median_SE* -

Examples

```
# example code
esci::overview(data_latimier_3groups, "Test%", "Group")
```

`overview_nominal` *Calculates descriptive statistics for a numerical variable*

Description

This function calculated basic descriptive statistics for a categorical/ nominal variable. Inputs can be summary data, vectors, or a data frame.

Usage

```
overview_nominal(
  data = NULL,
  outcome_variable = NULL,
  grouping_variable = NULL,
  cases = NULL,
  outcome_variable_levels = NULL,
  outcome_variable_name = "My Outcome Variable",
  grouping_variable_name = "My Grouping Variable",
  conf_level = 0.95,
  count_NA = FALSE
)
```

Arguments

`data`	• for raw data, a data frame or tibble
`outcome_variable`	• for raw data, either a vector containing factor data or the name of a data-frame column containing a factor
`grouping_variable`	• for raw data, either NULL (default), or the vector of a factor or a data-frame column containing a factor
`cases`	For summary data - A vector of 1 or more counts, integers>0
`outcome_variable_levels`	For summary data - An optional vector of group labels, same length as cases. If not passed, auto-generated.
`outcome_variable_name`	Optional friendly name for the outcome variable. Defaults to 'My Outcome Variable'. Ignored if a data-frame is passed, this argument is ignored.
`grouping_variable_name`	Optional friendly name for the grouping variable. Defaults to 'My Grouping Variable'. Ignored for summary data and for data frames – only used if vectors of data are passed.
`conf_level`	The confidence level for the confidence interval. Given in decimal form. Defaults to 0.95.
`count_NA`	Logical to count NAs (TRUE) in total N or not (FALSE)

Value

Returns a table of descriptive statistics

- **overview_nominal**
  - *outcome_variable_name* -
  - *outcome_variable_level* -
  - *cases* -
  - *n* -
  - *P* -
  - *P_LL* -
  - *P_UL* -
  - *P_SE* -
  - *P_adjusted* -
  - *ta_LL* -
  - *ta_UL* -

Examples

```
# example code
esci::overview_nominal(esci::data_latimier_3groups, "Group")
```

`plot_correlation` *Plot an estimated Pearson's r value*

Description

`plot_correlation` creates a ggplot2 plot suitable for visualizing an estimate correlation between two continuous variables (Pearson's *r*). This function can be passed an esci_estimate object generated by `estimate_r()`

Usage

```
plot_correlation(
  estimate,
  error_layout = c("halfeye", "eye", "gradient", "none"),
  error_scale = 0.3,
  error_normalize = c("groups", "all", "panels"),
  rope = c(NA, NA),
  ggtheme = NULL
)
```

Arguments

`estimate`	- An esci_estimate object generated by `estimate_r()`
`error_layout`	- Optional; One of 'halfeye', 'eye', 'gradient' or 'none' for how expected sampling error of the measure of central tendency should be displayed. **Caution - the displayed error distributions do not seem correct yet**
`error_scale`	- Optional real number > 0 specifying width of the expected sampling error visualization; default is 0.3
`error_normalize`	- Optional; One of 'groups' (default), 'all', or 'panels' specifying how width of expected sampling error distributions should be calculated.
`rope`	- Optional two-item vector specifying a region of practical equivalence (ROPE) to be highlighted on the plot. For a point null hypothesis, pass the same value (e.g. c(0, 0) to test a point null of exactly 0); for an interval null pass ascending values (e.g. c(-1, 1))
`ggtheme`	- Optional ggplot2 theme object to control overall styling; defaults to `ggplot2::theme_classic()`

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
# From raw data
data("data_thomason_1")

estimate_from_raw <- esci::estimate_r(
  esci::data_thomason_1,
  Pretest,
  Posttest
)

# To visualize the value of r
myplot_correlation <- esci::plot_correlation(estimate_from_raw)

# To visualize the data (scatterplot) and use regression to obtain Y' from X
myplot_scatter_from_raw <- esci::plot_scatter(estimate_from_raw, predict_from_x = 10)

# To evaluate a hypothesis (interval null from -0.1 to 0.1):
```

```
res_htest_from_raw <- esci::test_correlation(
  estimate_from_raw,
  rope = c(-0.1, 0.1)
)


# From summary data
estimate_from_summary <- esci::estimate_r(r = 0.536, n = 50)

# To visualize the value of r
myplot_correlation_from_summary <- esci::plot_correlation(estimate_from_summary)

# To evaluate a hypothesis (interval null from -0.1 to 0.1):
res_htest_from_summary <- esci::test_correlation(
  estimate_from_summary,
  rope = c(-0.1, 0.1)
)
```

`plot_describe` *Plot a histogram or dotplot of an estimated magnitude with raw data*

Description

`plot_describe` Takes an estimate produced from estimate_magnitude and produces a dotplot or histogram. It can mark various descriptive statistics on the plot, including mean, median, sd, quartiles, and z lines. If a percentile is passed, it color-codes data based on if it is above or below that percentile.

Usage

```
plot_describe(
  estimate,
  type = c("histogram", "dotplot"),
  mark_mean = FALSE,
  mark_median = FALSE,
  mark_sd = FALSE,
  mark_quartiles = FALSE,
  mark_z_lines = FALSE,
  mark_percentile = NULL,
  histogram_bins = 12,
  ylim = c(0, NA),
  ybreaks = NULL,
  xlim = c(NA, NA),
  xbreaks = NULL,
  fill_regular = "#008DF9",
  fill_highlighted = "#E20134",
```

```
  color = "black",
  marker_size = 5,
  ggtheme = NULL
)
```

Arguments

`estimate`	A esci_estimate object with raw data an es_mean
`type`	histogram or dotplot
`mark_mean`	should mean be marked?
`mark_median`	should median be marked?
`mark_sd`	should mean be marked?
`mark_quartiles`	should mean be marked?
`mark_z_lines`	should z lines be marked?
`mark_percentile`	a percentile (0 to 1) to be marked
`histogram_bins`	number of bins if a histogram
`ylim`	2-length numeric vector
`ybreaks`	numeric >= 1
`xlim`	2-length numeric vector
`xbreaks`	numeric >= 1
`fill_regular`	color for
`fill_highlighted`	color for
`color`	outline color
`marker_size`	Size of markers
`ggtheme`	theme to apply, if any

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
# example code
# Generate an estimate on a single continuous variable
estimate <- esci::estimate_magnitude(esci::data_latimier_3groups, `Test%`)

# Now describe the result, with a histogram
myplot_hist <- plot_describe(estimate)

# Same, but as a dotplot and mark the mean
myplot_dots <- plot_describe(estimate, type = "dotplot", mark_mean = TRUE)
```

`plot_interaction` *Plot the interaction from a 2x2 design*

Description

`plot_interaction` helps visualize the interaction from a 2x2 design. It plots the 2 simple effects for the first factor and can also help visualize the CIs on those simple effects. It is the comparison between those simple effects that represents an interaction (the difference in the difference). You can pass esci-estimate objects generated `estimate_mdiff_2x2_between()` or `estimate_mdiff_2x2_mixed()`. This function returns a ggplot2 object.

Usage

```
plot_interaction(
  estimate,
  effect_size = c("mean", "median"),
  show_CI = FALSE,
  ggtheme = NULL,
  line_count = 100,
  line_alpha = 0.02
)
```

Arguments

`estimate`	A esci_estimate object with raw data an es_mdiff_2x2_ function
`effect_size`	Optional; one of 'mean' or 'median' to determine the measure of central tendency plotted. Note that median is only available if the estimate was generated from raw data. Defaults to 'mean'
`show_CI`	Optional logical; set to TRUE to visualize the confidence intervals on each simple effect; defaults to FALSE
`ggtheme`	Optional ggplot2 theme object to specify the visual style of the plot. Defaults to `ggplot2::theme_classic()`

`line_count`	Optional integer > 0 to specify the number of lines used to visualize the simple-effect confidence intervals; defaults to 100
`line_alpha`	Optional numeric between 0 and 1 to specify the alpha (transparency) of the confidence interval lines; defaults to 0.02

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
data("data_videogameaggression")

estimates_from_raw <- esci::estimate_mdiff_2x2_between(
  esci::data_videogameaggression,
  Agression,
  Violence,
  Difficulty
)

# To visualize the estimated mean difference for the interaction
myplot_from_raw <- esci::plot_mdiff(
  estimates_from_raw$interaction,
  effect_size = "median"
)

# To conduct a hypothesis test on the mean difference
res_htest_from_raw <- esci::test_mdiff(
  estimates_from_raw$interaction,
  effect_size = "median"
)


# From summary data
means <- c(1.5, 1.14, 1.38, 2.22)
sds <- c(1.38, .96,1.5, 1.68)
ns <- c(26, 26, 25, 26)
grouping_variable_A_levels <- c("Evening", "Morning")
grouping_variable_B_levels <- c("Sleep", "No Sleep")

estimates_from_summary <- esci::estimate_mdiff_2x2_between(
```

```
    means = means,
    sds = sds,
    ns = ns,
    grouping_variable_A_levels = grouping_variable_A_levels,
    grouping_variable_B_levels = grouping_variable_B_levels,
    grouping_variable_A_name = "Testing Time",
    grouping_variable_B_name = "Rest",
    outcome_variable_name = "False Memory Score",
    assume_equal_variance = TRUE
  )

  # To visualize the estimated mean difference for the interaction
  plot_mdiff_interaction <- esci::plot_mdiff(
    estimates_from_summary$interaction,
    effect_size = "mean"
  )

  # To visualize the interaction as a line plot
  plot_interaction_line <- esci::plot_interaction(estimates_from_summary)

  # Same but with fan effect representing each simple-effect CI
  plot_interaction_line_CI <- esci::plot_interaction(
    estimates_from_summary,
    show_CI = TRUE
  )

  # To conduct a hypothesis test on the mean difference
  res_htest_from_raw <- esci::test_mdiff(
    estimates_from_summary$interaction,
    effect_size = "mean"
  )
```

`plot_magnitude` *Plot the mean or median for a continuous variable*

Description

`plot_magnitude` creates a ggplot2 plot suitable for visualizing the results of a study with one group and one or more continuous outcome variables. It can highlight either the mean or median of each outcome variable. This function can be passed an esci_estimate object generated by `estimate_magnitude()`

Usage

```
plot_magnitude(
  estimate,
  effect_size = c("mean", "median"),
  data_layout = c("random", "swarm", "none"),
```

```
  data_spread = 0.25,
  error_layout = c("halfeye", "eye", "gradient", "none"),
  error_scale = 0.3,
  error_nudge = 0.35,
  error_normalize = c("groups", "all", "panels"),
  rope = c(NA, NA),
  ggtheme = NULL
)
```

Arguments

`estimate`	• An esci_estimate object generated by `estimate_magnitude()`
`effect_size`	• Optional; One of 'mean' (default) or 'median'; specifies which measure of central tendency to highlight; note medians are only available if the esci_estimate object was generated from raw data
`data_layout`	• Optional; One of 'random' (default), 'swarm', or 'none' for how raw data (if available) will be displayed
`data_spread`	• Optional real number > 0 specifying width raw data (if available) should take on the graph; default is 0.25; default spacing between two groups on the graph is 1
`error_layout`	• Optional; One of 'halfeye', 'eye', 'gradient' or 'none' for how expected sampling error of the measure of central tendency should be displayed. Currently, only applies if 'mean' is selected as measure of central tendency
`error_scale`	• Optional real number > 0 specifying width of the expected sampling error visualization; default is 0.3
`error_nudge`	• Optional amount by which error distribution should be offset; default is 0.35
`error_normalize`	• Optional; One of 'groups' (default), 'all', or 'panels' specifying how width of expected sampling error distributions should be calculated.
`rope`	• Optional two-item vector specifying a region of practical equivalence (ROPE) to be highlighted on the plot. For a point null hypothesis, pass the same value (e.g. c(0, 0) to test a point null of exactly 0); for an interval null pass ascending values (e.g. c(-1, 1))
`ggtheme`	• Optional ggplot2 theme object to control overall styling; defaults to `ggplot2::theme_classic()`

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
# From raw data
data("data_penlaptop1")

estimate_from_raw <- esci::estimate_magnitude(
  data = data_penlaptop1[data_penlaptop1$condition == "Pen", ],
    outcome_variable = transcription
)

# To visualize the estimate
myplot_from_raw <- esci::plot_magnitude(
  estimate_from_raw,
  effect_size = "median"
)


# From summary data
mymean <- 24.5
mysd <- 3.65
myn <- 40

estimate_from_summary <- esci::estimate_magnitude(
  mean = mymean,
  sd = mysd,
  n = myn
)

# To visualize the estimate
myplot_from_summary <- esci::plot_magnitude(
  estimate_from_summary,
  effect_size = "mean"
)
```

`plot_mdiff` *Plots for comparing continuous outcome variables between conditions*

Description

`plot_mdiff` helps visualize comparisons of a continuous outcome variable between conditions. It can plot raw data (if available) for each condition, the mean or median (raw data only) for each condition, and it emphasizes a 1-df comparison among conditions, plotting the estimated difference and its confidence interval with a difference axis. You can pass esci-estimate objects generated by `estimate_mdiff_one()`, `estimate_mdiff_two()`, `estimate_mdiff_paired()`, `estimate_mdiff_ind_contrast()`, `estimate_mdiff_2x2_between()`, and `estimate_mdiff_2x2_mixed()`. This function returns a ggplot2 object.

Usage

```
plot_mdiff(
  estimate,
  effect_size = c("mean", "median"),
  data_layout = c("random", "swarm", "none"),
  data_spread = 0.15,
  error_layout = c("halfeye", "eye", "gradient", "none"),
  error_scale = 0.3,
  error_nudge = 0.4,
  error_normalize = c("groups", "all", "panels"),
  difference_axis_units = c("raw", "sd"),
  difference_axis_breaks = 5,
  difference_axis_space = 1,
  simple_contrast_labels = TRUE,
  ylim = c(NA, NA),
  ybreaks = 5,
  rope = c(NA, NA),
  rope_units = c("raw", "sd"),
  ggtheme = NULL
)
```

Arguments

`estimate`	An esci-estimate object generated by an estimate_mdiff_ function
`effect_size`	Optional; one of 'mean' or 'median' to determine the measure of central tendency plotted. Note that median is only available if the estimate was generated from raw data. Defaults to 'mean'
`data_layout`	Optional; one of 'random', 'swarm', or 'none' to determine how raw data (if available) will be displayed. Defaults to 'random'
`data_spread`	Optional numeric determining width raw data will use in each condition. Defaults to 0.15 (relative to 1 unit per condition)
`error_layout`	Optional; one of 'halfeye', 'eye', 'gradient' or 'none' to determine how expected error distribution will be displayed for each estimated parameter. Defaults to 'halfeye'. Currently does not apply if 'median' is selected as effect size, in which case a simple error bar will be used
`error_scale`	Optional numeric determining width of the expected error distribution. Defaults to 0.3
`error_nudge`	Optional numeric determining degree to which measures of central tendency will be shifted to the right of the raw data; defaults to 0.4
`error_normalize`	Optional; one of 'groups', 'all', or 'panels' to determine how width of the expected error distributions will be normalized. Defaults to 'groups'. See documentation in ggdist

`difference_axis_units`	Optional; one of 'raw' or 'sd' to determine if markings on the difference axis will be in raw-score units or in standard-deviation units. For 'sd' the standard deviation of the mean difference is used, and this is true even if 'median' is selected as the effect size
`difference_axis_breaks`	Optional numeric > 1 of suggested number of breaks for the difference axis. Defaults to 5
`difference_axis_space`	Optional numeric > 0 to indicate spacing to the difference axis. Defaults to 1
`simple_contrast_labels`	Optional logical to determine if contrasts are given simple labels ('Reference', 'Comparison', 'Difference') or more descriptive labels based on the contrast specified.
`ylim`	Optional 2-item vector specifying y-axis limits. Defaults to c(NA NA); Use NA to specify auto-limit.
`ybreaks`	Optional numeric > 2 for suggested number of y-axis breaks; defaults to 5
`rope`	Optional 2-item vector with item 2 >= item 1. Use to specify a range of values to use to visualize a hypothesis test. If both values are the same, a point-null hypothesis test will be visualized. If item2 > item1 an interval-null hypothesis test will be visualized. Defaults to c(NA, NA), which is to not visualize a hypothesis test
`rope_units`	Optional; one of 'raw' or 'sd' to indicate units of the rope passed. Defaults to 'raw'
`ggtheme`	Optional ggplot2 theme object to specify the visual style of the plot. Defaults to `ggplot2::theme_classic()`

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
# From raw data
data("data_penlaptop1")

estimate_from_raw <- esci::estimate_mdiff_two(
  data = data_penlaptop1,
```

```
  outcome_variable = transcription,
  grouping_variable = condition,
  switch_comparison_order = TRUE,
  assume_equal_variance = TRUE
)

# To visualize the estimated median difference (raw data only)
myplot_from_raw <- esci::plot_mdiff(
  estimate_from_raw,
  effect_size = "median"
)

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_mdiff(
  estimate_from_raw,
  effect_size = "median",
  rope = c(-2, 2)
)


# From summary data
estimate_from_summary <- esci::estimate_mdiff_two(
  comparison_mean = 12.09,
  comparison_sd = 5.52,
  comparison_n = 103,
  reference_mean = 6.88,
  reference_sd = 4.22,
  reference_n = 48,
  grouping_variable_levels = c("Ref-Laptop", "Comp-Pen"),
  outcome_variable_name = "% Transcription",
  grouping_variable_name = "Note-taking type",
  assume_equal_variance = TRUE
)

# To visualize the estimated mean difference
myplot <- esci::plot_mdiff(
  estimate_from_summary,
  effect_size = "mean"
)

# To conduct a hypothesis test
res_htest_from_summary <- esci::test_mdiff(
  estimate_from_summary,
  effect_size = "mean",
  rope = c(-2, 2)
)
```

`plot_meta` *Generates a forest plot displaying results of a meta-analysis*

Description

‘plot_meta’ returns a ggplot2 object visualizing the results of a meta-analysis, showing each study effect size and CI, the overall effect size and CI as a diamond, effect sizes estimated at each moderator level (if defined), and (optionally) prediction intervals for subsequent studies. This function requires as input an esci_estimate object generated by an esci meta-analysis function: `meta_any()`, `meta_d1()`, `meta_d2()`, `meta_mdiff_two()`, `meta_mean()`, `meta_pdiff_two()`, `meta_proportion()`, and `meta_r()`.

Usage

```
plot_meta(
  estimate,
  mark_zero = TRUE,
  include_PIs = FALSE,
  report_CIs = FALSE,
  explain_DR = FALSE,
  meta_diamond_height = 0.35,
  ggtheme = ggplot2::theme_classic()
)
```

Arguments

`estimate`	• an esci_estimate object generated by an esci meta_ function
`mark_zero`	• Boolean; defaults to TRUE to include a dotted line indicated no effect (effect_size = 0)
`include_PIs`	• Boolean; defaults to FALSE; set to TRUE to include prediction intervals for the overall effect and each moderator level (if defined)
`report_CIs`	• Boolean; defaults to FALSE; set to TRUE to include printed representation of each study effect size and CI along the right- hand of the figure
`explain_DR`	• Boolean; defaults to FALSE; set to TRUE if no moderator is defined to show both the RE and FE effect sizes to represent how the diamond ration measure of effect-size heterogeneity is calculated
`meta_diamond_height`	• Optional real number > 0 to indicate height that each meta-analytic diamond should be drawn; defaults to 0.35
`ggtheme`	• Optional ggplot2 theme object to control overall styling; defaults to `ggplot2::theme_classic()`

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
# Data set -- see Introduction to the New Statistics, 2nd edition
data("data_mccabemichael_brain")

# Meta-analysis: random effects, no moderator
estimate <- esci::meta_mdiff_two(
  data = esci::data_mccabemichael_brain,
  comparison_means = "M Brain",
  comparison_sds = "s Brain",
  comparison_ns = "n Brain",
  reference_means = "M No Brain",
  reference_sds = "s No Brain",
  reference_ns = "n No Brain",
  labels = "Study name",
  effect_label = "Brain Photo Rating - No Brain Photo Rating",
  assume_equal_variance = TRUE,
  random_effects = TRUE
)
# Forest plot
myplot_forest <- esci::plot_meta(estimate)


# Meta-analysis: random effects, moderator
estimate_moderator <- esci::meta_mdiff_two(
  data = esci::data_mccabemichael_brain,
  comparison_means = "M Brain",
  comparison_sds = "s Brain",
  comparison_ns = "n Brain",
  reference_means = "M No Brain",
  reference_sds = "s No Brain",
  reference_ns = "n No Brain",
  labels = "Study name",
  moderator = "Research group",
  effect_label = "Brain Photo Rating - No Brain Photo Rating",
  assume_equal_variance = TRUE,
  random_effects = TRUE
)
# Forest plot
myplot_forest_moderator <- esci::plot_meta(estimate_moderator)


# Meta-analysis: random effects, moderator, output d_s
estimate_moderator_d <- esci::meta_mdiff_two(
  data = esci::data_mccabemichael_brain,
  comparison_means = "M Brain",
  comparison_sds = "s Brain",
  comparison_ns = "n Brain",
  reference_means = "M No Brain",
```

```
    reference_sds = "s No Brain",
    reference_ns = "n No Brain",
    labels = "Study name",
    moderator = "Research group",
    effect_label = "Brain Photo Rating - No Brain Photo Rating",
    assume_equal_variance = TRUE,
    random_effects = TRUE
  )
  # Forest plot
  myplot_forest_moderator_d <- esci::plot_meta(estimate_moderator_d)
```

`plot_pdiff` *Plots for comparing categorical outcome variables between conditions*

Description

`plot_pdiff` helps visualize comparisons of a categorical outcome variable between conditions. It plots proportions of cases for each level of grouping variable and emphasizes a 1-df comparison among conditions, plotting the estimated difference and its confidence interval with a difference axis. You can pass esci-estimate objects generated by `estimate_pdiff_one()`, `estimate_pdiff_two()`, `estimate_pdiff_paired()`, `estimate_pdiff_ind_contrast()` This function returns a ggplot2 object.

Usage

```
plot_pdiff(
  estimate,
  error_layout = c("halfeye", "eye", "gradient", "none"),
  error_scale = 0.3,
  error_normalize = c("groups", "all", "panels"),
  difference_axis_breaks = 5,
  difference_axis_space = 1,
  simple_contrast_labels = TRUE,
  ylim = c(NA, NA),
  ybreaks = 5,
  rope = c(NA, NA),
  ggtheme = NULL
)
```

Arguments

`estimate`	An esci-estimate object generated by an estimate_pdiff_ function
`error_layout`	Optional; one of 'halfeye', 'eye', 'gradient' or 'none' to determine how expected error distribution will be displayed for each estimated parameter. Defaults to 'halfeye'. Currently does not apply if 'median' is selected as effect size, in which case a simple error bar will be used

`error_scale`	Optional numeric determining width of the expected error distribution. Defaults to 0.3
`error_normalize`	Optional; one of 'groups', 'all', or 'panels' to determine how width of the expected error distributions will be normalized. Defaults to 'groups'. See documentation in ggdist
`difference_axis_breaks`	Optional numeric > 1 of suggested number of breaks for the difference axis. Defaults to 5
`difference_axis_space`	Optional numeric > 0 to indicate spacing to the difference axis. Defaults to 1
`simple_contrast_labels`	Optional logical to determine if contrasts are given simple labels ('Reference', 'Comparison', 'Difference') or more descriptive labels based on the contrast specified.
`ylim`	Optional 2-item vector specifying y-axis limits. Defaults to c(NA NA); Use NA to specify auto-limit.
`ybreaks`	Optional numeric > 2 for suggested number of y-axis breaks; defaults to 5
`rope`	Optional 2-item vector with item 2 >= item 1. Use to specify a range of values to use to visualize a hypothesis test. If both values are the same, a point-null hypothesis test will be visualized. If item2 > item1 an interval-null hypothesis test will be visualized. Defaults to c(NA, NA), which is to not visualize a hypothesis test
`ggtheme`	Optional ggplot2 theme object to specify the visual style of the plot. Defaults to `ggplot2::theme_classic()`

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
data("data_campus_involvement")

estimate_from_raw <- esci::estimate_pdiff_two(
  esci::data_campus_involvement,
  CommuterStatus,
  Gender
```

```
)

# To visualize the estimate
myplot_from_raw <- esci::plot_pdiff(estimate_from_raw)

# To conduct a hypothesis test
res_htest_from_raw <- esci::test_pdiff(estimate_from_raw)


# From summary_data
estimate_from_summary <- esci::estimate_pdiff_two(
  comparison_cases = 10,
  comparison_n = 20,
  reference_cases = 78,
  reference_n = 252,
  grouping_variable_levels = c("Original", "Replication"),
  conf_level = 0.95
)

# To visualize the estimate
myplot_from_summary <- esci::plot_pdiff(estimate_from_summary)

#' # To conduct a hypothesis test
res_htest_from_summary <- esci::test_pdiff(estimate_from_summary)
```

`plot_proportion` *Plot an estimated proportion*

Description

`plot_proportion` creates a ggplot2 plot suitable for visualizing an estimated proportion from a categorical variable. This function can be passed an esci_estimate object generated by `estimate_proportion()`

Usage

```
plot_proportion(
  estimate,
  error_layout = c("halfeye", "eye", "gradient", "none"),
  error_scale = 0.3,
  error_normalize = c("groups", "all", "panels"),
  rope = c(NA, NA),
  plot_possible = FALSE,
  ggtheme = NULL
)
```

Arguments

`estimate`	- An esci_estimate object generated by `estimate_proportion()`
`error_layout`	- Optional; One of 'halfeye', 'eye', 'gradient' or 'none' for how expected sampling error of the measure of central tendency should be displayed. **Caution - the displayed error distributions do not seem correct yet**
`error_scale`	- Optional real number > 0 specifying width of the expected sampling error visualization; default is 0.3
`error_normalize`	- Optional; One of 'groups' (default), 'all', or 'panels' specifying how width of expected sampling error distributions should be calculated.
`rope`	- Optional two-item vector specifying a region of practical equivalence (ROPE) to be highlighted on the plot. For a point null hypothesis, pass the same value (e.g. c(0, 0) to test a point null of exactly 0); for an interval null pass ascending values (e.g. c(-1, 1))
`plot_possible`	- Boolean; defaults to FALSE; TRUE to plot lines at each discrete proportion possible given the sample size (e.g for a proportion with 10 total cases, would draw lines at 0, .1, .2, etc.)
`ggtheme`	- Optional ggplot2 theme object to control overall styling; defaults to `ggplot2::theme_classic()`

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
# From raw data
data("data_campus_involvement")

estimate_from_raw <- esci::estimate_proportion(
  esci::data_campus_involvement,
  CommuterStatus
)


# To visualize the estimate
myplot_from_raw <- esci::plot_proportion(estimate_from_raw)
```

```
# From summary data
estimate_from_summary <- esci::estimate_proportion(
  cases = c(8, 22-8),
  outcome_variable_levels = c("Affected", "Not Affected")
)

# To visualize the estimate
myplot_from_summary<- esci::plot_proportion(estimate_from_summary)
```

`plot_rdiff` *Plots for comparing Pearson* r *values between conditions*

Description

`plot_rdiff` helps visualize comparisons of Pearson's r estimates between conditions. It plots the Pearson's r value for each level of a grouping variable and emphasizes a 1-df comparison among conditions, plotting the estimated difference and its confidence interval with a difference axis. You can pass esci-estimate objects generated by `estimate_rdiff_two()`. This function returns a ggplot2 object.

Usage

```
plot_rdiff(
  estimate,
  error_layout = c("halfeye", "eye", "gradient", "none"),
  error_scale = 0.3,
  error_normalize = c("groups", "all", "panels"),
  difference_axis_breaks = 5,
  simple_contrast_labels = TRUE,
  ylim = c(NA, NA),
  ybreaks = 5,
  rope = c(NA, NA),
  ggtheme = NULL
)
```

Arguments

`estimate`	An esci-estimate object generated by an estimate_pdiff_ function
`error_layout`	Optional; one of 'halfeye', 'eye', 'gradient' or 'none' to determine how expected error distribution will be displayed for each estimated parameter. Defaults to 'halfeye'. Currently does not apply if 'median' is selected as effect size, in which case a simple error bar will be used
`error_scale`	Optional numeric determining width of the expected error distribution. Defaults to 0.3

`error_normalize`	Optional; one of 'groups', 'all', or 'panels' to determine how width of the expected error distributions will be normalized. Defaults to 'groups'. See documentation in ggdist
`difference_axis_breaks`	Optional numeric > 1 of suggested number of breaks for the difference axis. Defaults to 5
`simple_contrast_labels`	Optional logical to determine if contrasts are given simple labels ('Reference', 'Comparison', 'Difference') or more descriptive labels based on the contrast specified.
`ylim`	Optional 2-item vector specifying y-axis limits. Defaults to c(NA NA); Use NA to specify auto-limit.
`ybreaks`	Optional numeric > 2 for suggested number of y-axis breaks; defaults to 5
`rope`	Optional 2-item vector with item 2 >= item 1. Use to specify a range of values to use to visualize a hypothesis test. If both values are the same, a point-null hypothesis test will be visualized. If item2 > item1 an interval-null hypothesis test will be visualized. Defaults to c(NA, NA), which is to not visualize a hypothesis test
`ggtheme`	Optional ggplot2 theme object to specify the visual style of the plot. Defaults to `ggplot2::theme_classic()`

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
# From raw data
data("data_campus_involvement")

estimate_from_raw <- esci::estimate_rdiff_two(
  esci::data_campus_involvement,
  GPA,
  SWB,
  Gender
)
```

```
# To visualize the difference in r
myplot_from_raw <- esci::plot_rdiff(estimate_from_raw)

# To visualize the data (scatterplot) by group
myplot_scatter <- esci::plot_scatter(estimate_from_raw)

# To evaluate a hypothesis (by default: point null of exaclty 0):
res_htest_from_raw <- esci::test_rdiff(
  estimate_from_raw
)


# From summary data
estimate <- esci::estimate_rdiff_two(
  comparison_r = .53,
  comparison_n = 45,
  reference_r = .41,
  reference_n = 59,
  grouping_variable_levels = c("Females", "Males"),
  x_variable_name = "Satisfaction with life",
  y_variable_name = "Body satisfaction",
  grouping_variable_name = "Gender",
  conf_level = .95
)

myplot_from_summary <- esci::plot_rdiff(estimate)

# To evaluate a hypothesis (interval null from -0.1 to 0.1):
res_htest_from_summary <- esci::test_rdiff(
  estimate,
  rope = c(-0.1, 0.1)
)
```

`plot_scatter` *Generates a scatter plot of data for two continuous variables*

Description

`plot_scatter` returns a ggplot2 object of data from two continuous variables. Can indicate regression line and its confidence interval,prediction intervals regression residuals and more. This function requires as input an esci_estimate object generated by `estimate_r()`

Usage

```
plot_scatter(
  estimate,
  show_line = FALSE,
  show_line_CI = FALSE,
```

```
  show_PI = FALSE,
  show_residuals = FALSE,
  show_mean_lines = FALSE,
  show_r = FALSE,
  predict_from_x = NULL,
  plot_as_z = FALSE,
  ggtheme = ggplot2::theme_classic()
)
```

Arguments

`estimate`	• an esci_estimate object generated by `estimate_r()`
`show_line`	• Boolean; defaults to FALSE; set to TRUE to show the regression line
`show_line_CI`	• Boolean; defaults to FALSE; set to TRUE to show the confidence interval on the regression line
`show_PI`	• Boolean; defaults to FALSE; set to TRUE to show prediction intervals
`show_residuals`	• Boolean; defaults to FALSE; set to TRUE to show residuals of prediction
`show_mean_lines`	• Boolean; defaults to FALSE; set to TRUE to plot lines showing the mean of each variable
`show_r`	• Boolean; defaults to FALSE; set to TRUE to print the *r* value and its CI on the plot
`predict_from_x`	• Optional real number in the range of the x variable for the plot; Defaults to NULL; if passed, the graph shows the predicted Y' for this x value
`plot_as_z`	• Boolean; defaults to FALSE; set to TRUE to convert x and y scores to z scores prior to plotting
`ggtheme`	• Optional ggplot2 theme object to control overall styling; defaults to `ggplot2::theme_classic()`

Details

This function was developed primarily for student use within jamovi when learning along with the text book Introduction to the New Statistics, 2nd edition (Cumming & Calin-Jageman, 2024).

Expect breaking changes as this function is improved for general use. Work still do be done includes:

- Revise to avoid deprecated ggplot features
- Revise for consistent ability to control aesthetics and consistent layer names

Value

Returns a ggplot object

Examples

```
# From raw data
data("data_thomason_1")

estimate_from_raw <- esci::estimate_r(
  esci::data_thomason_1,
  Pretest,
  Posttest
)

# To visualize the value of r
myplot_correlation <- esci::plot_correlation(estimate_from_raw)

# To visualize the data (scatterplot) and use regression to obtain Y' from X
myplot_scatter_from_raw <- esci::plot_scatter(estimate_from_raw, predict_from_x = 10)

# To evaluate a hypothesis (interval null from -0.1 to 0.1):
res_htest_from_raw <- esci::test_correlation(
  estimate_from_raw,
  rope = c(-0.1, 0.1)
)


# From summary data
estimate_from_summary <- esci::estimate_r(r = 0.536, n = 50)

# To visualize the value of r
myplot_correlation_from_summary <- esci::plot_correlation(estimate_from_summary)

# To evaluate a hypothesis (interval null from -0.1 to 0.1):
res_htest_from_summary <- esci::test_correlation(
  estimate_from_summary,
  rope = c(-0.1, 0.1)
)
```

`print.esci_estimate` *Print an esci_estimate*

Description

Pretties up the printing of a complex esci_estimate object.

Usage

```
## S3 method for class 'esci_estimate'
print(x, ..., verbose = FALSE)
```

Arguments

`x`	• object to print; must be of class esci_estimate
`...`	S3 signature for generic plot function.
`verbose`	• optional logical print all details; defaults to false

`test_correlation` *Test a hypothesis about the strength of a Pearson's* r *correlation*

Description

`test_correlation` is suitable for testing a hypothesis about a the strength of correlation between two continuous variables (designs in which Pearson's *r* is a suitable measure of correlation).

Usage

```
test_correlation(estimate, rope = c(0, 0), output_html = FALSE)
```

Arguments

`estimate`	• An esci_estimate object generated by the estimate_r function
`rope`	• A two-element vector defining the Region of Practical Equivalence (ROPE). Specify c(0, 0) to test a point null of exactly 0. Specify any two ascending values between -1 and 1 to test an interval null (e.g. c(.25, .45) to test the hypothesis that Pearson's *r* in the population (rho) is between .25 and .45).
`output_html`	• TRUE to return results in HTML; FALSE (default) to return standard output

Details

This function can be passed an esci_estimate object generated by `estimate_r()`.

It can test hypotheses about a specific value for the difference (a point null) or about a range of values (an interval null)

Value

Returns a list with 1-2 data frames

- **point_null** - always returned
    - *test_type* - 'Nil hypothesis test', meaning a test against H0 = 0
    - *outcome_variable_name* - Name of the outcome variable
    - *effect* - Label for the effect being tested
    - *null_words* - Express the null in words
    - *confidence* - Confidence level, integer (95 for 95%, etc.)
    - *LL* - Lower boundary of the confidence% CI for the effect
    - *UL* - Upper boundary of the confidence% CI for the effect

  - *CI* - Character representation of the CI for the effect
  - *CI_compare* - Text description of relation between CI and null
  - *t* - If applicable, t value for hypothesis test
  - *df* - If applicable, degrees of freedom for hypothesis test
  - *p* - If applicable, p value for hypothesis test
  - *p_result* - Text representation of p value obtained
  - *null_decision* - Text represention of the decision for the null
  - *conclusion* - Text representation of conclusion to draw
  - *significant* - TRUE/FALSE if significant at alpha = 1-CI
- **interval_null** - returned only if an interval null is specified
  - *test_type* - 'Practical significance test', meaning a test against an interval null
  - *outcome_variable_name* -
  - *effect* - Name of the outcome variable
  - *rope* - Test representation of null interval
  - *confidence* - Confidence level, integer (95 for 95%, etc.)
  - *CI* - Character representation of the CI for the effect
  - *rope_compare* - Text description of relation between CI and null interval
  - *p_result* - Text representation of p value obtained
  - *conclusion* - Text representation of conclusion to draw
  - *significant* - TRUE/FALSE if significant at alpha = 1-CI

Examples

```
# example code
estimate <- esci::estimate_r(r = 0.536, n = 50)

# Test against a point null of exactly 0
test_correlation(estimate)

# Test against an interval null (-0.1, 0.1)
test_correlation(estimate, rope = c(-0.1, 0.1))
```

`test_mdiff` *Test a hypothesis about a difference in a continuous outcome variable.*

Description

`test_mdiff` is suitable for conducting a testing a hypothesis about the magnitude of difference between two conditions for a continuous outcome variable. It can test hypotheses about differences in means or medians for both independent and paired designs.

Usage

```
test_mdiff(
  estimate,
  effect_size = c("mean", "median"),
  rope = c(0, 0),
  rope_units = c("raw", "sd"),
  output_html = FALSE
)
```

Arguments

`estimate`	• An esci_estimate object generated by an estimate_mdiff_ function
`effect_size`	• One of 'mean' or 'median'. The effect size selected must be available in the esci_estimate object; medians are only available when the estimate was generated from raw data.
`rope`	• A two-element vector defining the Region of Practical Equivalence (ROPE). Specify c(0, 0) to test a point null of exactly 0. Specify any two ascending values to test an interval null (e.g. c(-1, 1) to test the hypothesis tha the difference is between -1 and 1).
`rope_units`	• One of 'raw' (default) or 'sd', specifies the units of the ROPE. If 'sd' is specified, the rope is defined in standard deviation units (e.g. c(-1, 1) is taken as between -1 and 1 *standard deviations* from 0). When sd is used, the ROPE is converted to raw scores and then the test is conducted on raw scores.
`output_html`	• TRUE to return results in HTML; FALSE (default) to return standard output

Details

This function can be passed an esci_estimate object generated by `estimate_mdiff_one()`, `estimate_mdiff_two()`, `estimate_mdiff_paired()`, or `estimate_mdiff_ind_contrast()`.

It can test hypotheses about a specific value for the difference (a point null) or about a range of values (an interval null)

Value

Returns a list with 1-2 data frames

- **point_null** - always returned
  - *test_type* - 'Nil hypothesis test', meaning a test against H0 = 0
  - *outcome_variable_name* - Name of the outcome variable
  - *effect* - Label for the effect being tested
  - *null_words* - Express the null in words
  - *confidence* - Confidence level, integer (95 for 95%, etc.)
  - *LL* - Lower boundary of the confidence% CI for the effect
  - *UL* - Upper boundary of the confidence% CI for the effect
  - *CI* - Character representation of the CI for the effect

  - *CI_compare* - Text description of relation between CI and null
  - *t* - If applicable, t value for hypothesis test
  - *df* - If applicable, degrees of freedom for hypothesis test
  - *p* - If applicable, p value for hypothesis test
  - *p_result* - Text representation of p value obtained
  - *null_decision* - Text represention of the decision for the null
  - *conclusion* - Text representation of conclusion to draw
  - *significant* - TRUE/FALSE if significant at alpha = 1-CI
- **interval_null** - returned only if an interval null is specified
  - *test_type* - 'Practical significance test', meaning a test against an interval null
  - *outcome_variable_name* -
  - *effect* - Name of the outcome variable
  - *rope* - Test representation of null interval
  - *confidence* - Confidence level, integer (95 for 95%, etc.)
  - *CI* - Character representation of the CI for the effect
  - *rope_compare* - Text description of relation between CI and null interval
  - *p_result* - Text representation of p value obtained
  - *conclusion* - Text representation of conclusion to draw
  - *significant* - TRUE/FALSE if significant at alpha = 1-CI

Examples

```
# example code
data("data_penlaptop1")

estimate <- esci::estimate_mdiff_two(
  data = data_penlaptop1,
  outcome_variable = transcription,
  grouping_variable = condition,
  switch_comparison_order = TRUE,
  assume_equal_variance = TRUE
)

# Test mean difference against point null of 0
esci::test_mdiff(
  estimate,
  effect_size = "mean"
)

# Test median difference against point null of 0
#  Note that t, df, p return NA because test is completed
#  by interval.
esci::test_mdiff(
  estimate,
  effect_size = "median"
)

# Test mean difference against interval null of -10 to 10
```

```
esci::test_mdiff(
  estimate,
  effect_size = "mean",
  rope = c(-10, 10)
)

# Test mean difference against interval null of d (-0.20, 0.20) d = 0.2 is often
# thought of as a small effect, so this test examines if the effect is
# negligible (clearly between negligble and small), substantive (clearly more
# than small), or unclear. The d boundaries provided are converted to raw scores
# and then the CI of the observed effect is compared to the raw-score boundaries
esci::test_mdiff(
  estimate,
  effect_size = "mean",
  rope = c(-0.2, 0.2),
  rope_units = "sd"
)
```

`test_pdiff` *Test a hypothesis about a difference in proportion*

Description

`test_pdiff` is suitable for testing a hypothesis about a difference in proportions between two conditions with a categorical outcome variable. It can test hypotheses for both independent and paired designs.

Usage

```
test_pdiff(estimate, rope = c(0, 0), output_html = FALSE)
```

Arguments

`estimate`	• An esci_estimate object generated by an estimate_pdiff_ function
`rope`	• A two-element vector defining the Region of Practical Equivalence (ROPE). Specify c(0, 0) to test a point null of exactly 0. Specify any two ascending values between -1 and 1 to test an interval null (e.g. c(-.25, .25) to test the hypothesis that the difference in proportion is between -.25 and .25).
`output_html`	• TRUE to return results in HTML; FALSE (default) to return standard output

Details

This function can be passed an esci_estimate object generated by `estimate_pdiff_one()`, `estimate_pdiff_two()`, `estimate_pdiff_paired()`, or `estimate_pdiff_ind_contrast()`.

It can test hypotheses about a specific value for the difference (a point null) or about a range of values (an interval null)

Value

Returns a list with 1-2 data frames

- **point_null** - always returned
    - *test_type* - 'Nil hypothesis test', meaning a test against H0 = 0
    - *outcome_variable_name* - Name of the outcome variable
    - *effect* - Label for the effect being tested
    - *null_words* - Express the null in words
    - *confidence* - Confidence level, integer (95 for 95%, etc.)
    - *LL* - Lower boundary of the confidence% CI for the effect
    - *UL* - Upper boundary of the confidence% CI for the effect
    - *CI* - Character representation of the CI for the effect
    - *CI_compare* - Text description of relation between CI and null
    - *t* - If applicable, t value for hypothesis test
    - *df* - If applicable, degrees of freedom for hypothesis test
    - *p* - If applicable, p value for hypothesis test
    - *p_result* - Text representation of p value obtained
    - *null_decision* - Text represention of the decision for the null
    - *conclusion* - Text representation of conclusion to draw
    - *significant* - TRUE/FALSE if significant at alpha = 1-CI
- **interval_null** - returned only if an interval null is specified
    - *test_type* - 'Practical significance test', meaning a test against an interval null
    - *outcome_variable_name* -
    - *effect* - Name of the outcome variable
    - *rope* - Test representation of null interval
    - *confidence* - Confidence level, integer (95 for 95%, etc.)
    - *CI* - Character representation of the CI for the effect
    - *rope_compare* - Text description of relation between CI and null interval
    - *p_result* - Text representation of p value obtained
    - *conclusion* - Text representation of conclusion to draw
    - *significant* - TRUE/FALSE if significant at alpha = 1-CI

Examples

```
estimate <- estimate_pdiff_two(
  comparison_cases = 10,
  comparison_n = 20,
  reference_cases = 78,
  reference_n = 252,
  grouping_variable_levels = c("Original", "Replication"),
  conf_level = 0.95
)

# Test against null of exactly
test_pdiff(estimate)
```

```
# Test against null of (-0.1, 0.1)
test_pdiff(estimate, rope = c(-0.1, 0.1))
```

`test_rdiff` *Test a hypothesis about a difference in correlation strength*

Description

`test_rdiff` is suitable for testing a hypothesis about a difference in correlation (*r*) between two conditions. At the moment, it can only test hypotheses for independent-group designs.

Usage

```
test_rdiff(estimate, rope = c(0, 0), output_html = FALSE)
```

Arguments

`estimate`	• An esci_estimate object generated by an estimate_rdiff_ function
`rope`	• A two-element vector defining the Region of Practical Equivalence (ROPE). Specify c(0, 0) to test a point null of exactly 0. Specify any two ascending values between -1 and 1 to test an interval null (e.g. c(-.25, .25) to test the hypothesis that the difference in correlation is between -.25 and .25).
`output_html`	• TRUE to return results in HTML; FALSE (default) to return standard output

Details

This function can be passed an esci_estimate object generated by `estimate_rdiff_two()`.

It can test hypotheses about a specific value for the difference (a point null) or about a range of values (an interval null)

Value

Returns a list with 1-2 data frames

- **point_null** - always returned
    - *test_type* - 'Nil hypothesis test', meaning a test against H0 = 0
    - *outcome_variable_name* - Name of the outcome variable
    - *effect* - Label for the effect being tested
    - *null_words* - Express the null in words
    - *confidence* - Confidence level, integer (95 for 95%, etc.)
    - *LL* - Lower boundary of the confidence% CI for the effect
    - *UL* - Upper boundary of the confidence% CI for the effect
    - *CI* - Character representation of the CI for the effect
    - *CI_compare* - Text description of relation between CI and null

  - *t* - If applicable, t value for hypothesis test
  - *df* - If applicable, degrees of freedom for hypothesis test
  - *p* - If applicable, p value for hypothesis test
  - *p_result* - Text representation of p value obtained
  - *null_decision* - Text represention of the decision for the null
  - *conclusion* - Text representation of conclusion to draw
  - *significant* - TRUE/FALSE if significant at alpha = 1-CI
- **interval_null** - returned only if an interval null is specified
  - *test_type* - 'Practical significance test', meaning a test against an interval null
  - *outcome_variable_name* -
  - *effect* - Name of the outcome variable
  - *rope* - Test representation of null interval
  - *confidence* - Confidence level, integer (95 for 95%, etc.)
  - *CI* - Character representation of the CI for the effect
  - *rope_compare* - Text description of relation between CI and null interval
  - *p_result* - Text representation of p value obtained
  - *conclusion* - Text representation of conclusion to draw
  - *significant* - TRUE/FALSE if significant at alpha = 1-CI

Examples

```
# example code
estimate <- esci::estimate_rdiff_two(
  comparison_r = .53,
  comparison_n = 45,
  reference_r = .41,
  reference_n = 59,
  grouping_variable_levels = c("Females", "Males"),
  x_variable_name = "Satisfaction with life",
  y_variable_name = "Body satisfaction",
  grouping_variable_name = "Gender",
  conf_level = .95
)
test_rdiff(estimate)
```

Index